# Structure in practical model error bounds 

## Citation for published version (APA):

Ariaans, L. J. J. M. (1997). Structure in practical model error bounds. [Phd Thesis 1 (Research TU/e / Graduation TU/e), Electrical Engineering]. Technische Universiteit Eindhoven. https://doi.org/10.6100/IR474943

## DOI:

10.6100/IR474943

## Document status and date:

Published: 01/01/1997

## Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

## Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.
Link to publication


## General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25 fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

## Take down policy

If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

## Structure in Practical Model Error Bounds

Leon Ariaans



# Structure in Practical Model Error Bounds 

## PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de Rector Magnificus, prof. dr. M. Rem, voor een commissie aangewezen door het College van

Dekanen in het openbaar te verdedigen op woensdag 26 februari 1997 om 16.00 uur
door

Leonardus Johannes Jacobus Marie Ariaans
geboren te Thorn

Dit proefschrift is goedgekeurd door de promotoren:
prof.dr.ir. A.C.P.M. Backx
en
prof.dr.ir. P.P.J. van den Bosch
Copromotor: dr. S. Weiland

## CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Ariaans, Leonardus J.J.M.
Structure in practical model error bounds / by
Leonardus J.J.M. Ariaans.
Eindhoven : Technische Universiteit Eindhoven, 1997.
Proefschrift. - ISBN 90-386-0240-5
NUGI 832
Trefw.: systeemidentificatie / onzekerheid / robuuste regelsystemen. Subject headings: control system analysis / modelling / ancertain systems:

Woe to the author who always wants to teach! The secret of being a bore is to tell everything.

- Voltaire, De la Nature de l'Homme (1737).

English translation taken from

- Donald E. Knuth, The TEXbook (1984).


## Voorwoord

Voor de lezer is dit proefschrift nog maar net begonnen, maar voor mij zit het er (bijna) op. "Het" slaat dan in eerste instantie op het schrijven van dit proefschrift, maar ook mijn tijd bij de vakgroep Meten en Regelen, eerst als student, later als promovendus, zit er nu op. Terugkijkend op deze tijd wil ik een aantal mensen bedanken.

Allereerst prof. Ton Backx, die mij deze baan heeft aangeboden voor ik ernaar kon vragen. Ik waardeer de vrijheid die hij mij tijdens mijn onderzoek gelaten heeft zeer, evenals zijn bijdrage aan dat onderzoek, die op verscheidene essentiële plaatsen in dit proefschrift aanwijsbaar is. Dit geldt zeker ook voor dr. Ad Damen en dr. Siep Weiland. Hun kritische maar altijd constructieve opmerkingen zijn een grote inspiratiebron voor mij geweest, temeer daar zij de materie elk op hun eigen wijze benaderen.

Vervolgens wil ik prof. Paul van den Bosch bedanken, niet alleen voor het doorlezen van de eerste versies van dit proefschrift, maar ook voor het vertrouwen dat hij de afgelopen tijd in mij getoond heeft. Dank ook aan prof. Okko Bosgra, voor het proeflezen van mijn proefschrift on do plezicrige discussies die daaruit voortgevloeid zijn.

De bereidheid van Jobert Ludlage om zijn drukke bezigheden bij, tegenwoordig, Aspentech te onderbreken om te voorkomen dat mijn onderzoek al te ver van de praktijk wegdriftte stel ik zeer op prijs.

Een bijzonder woord richt ik tot mijn kamergenoot gedurende vrijwel mijn hele promotieonderzoek, Jozef Mazák: you made me realise that life is full of disappointments but that it is better to be an optimist proven wrong than a pessimist proven right.

Uitdrukkelijke waardering verdienen ook Wim Beckers en Udo Bartzke voor het altijd snel oplossen van de schaarse problemen op computergebied. De mogelijkheden die zij mij hebben geboden om mijn (computer)werkplek naar eigen inzicht in te richten beschouw ik als een belangrijk privilege.

Mijn (oud)-collega's Mark Keulers, Heinz Falkus, Milan Hanajík, Par Dunias, Robert Jan Gorter, Yvo Boers, Maurice Heemels en Liu Hong mogen hier niet onvermeld blijven. Zij vormen een wezenlijke factor in het plezier waarmee ik mijn werk en ook andere activiteiten bij de vakgroep verricht heb. Dit geldt ook voor Dik de Bruin, die mij in de paar weken dat hij mijn kamergenoot was meer afwezig dan aanwezig vond, voor Paul Borghouts, die mijn taken in de feestcommissie verlichtte en mij vaak hielp het weekend op gepaste wijze te beginnen, en voor alle overige (oud)-leden van de vakgroep die nog niet met name genoemd zijn.
"Mijn" afstudeerders Winston Langeslag en Natal van Riel wil ik bedanken voor de bijdrage die zij - bewust en onbewust - aan mijn onderzoek hebben geleverd en voor de wijze waarop zij mijn inzicht in sommige aspecten van systeemidentificatie en regeltechniek in het algemeen hebben aangescherpt.

Als laatste bedank ik pap, mam en oos Jos. Sinds zij aan de wieg van dit alles hebben gestaan, zijn zij mijn steun en toeverlaat gebleven. Zonder deze veilige thuishaven was dit proefschrift niet mogelijk geweest.

Eindhoven, 1 januari 1997.

## Abstract

In many processes, control is applied to change the dynamic behaviour of the process such that the process performs better in some sense. Modern techniques for control design are, in general, based on a model of the process. If the model uncertainty, i.e. the difference between the model and the true process, can be bounded and the control design can take this bounded uncertainty into account, the resulting control is said to be robust. This thesis is concerned with the problem of finding bounds on the model uncertainty from experimental data and prior knowledge for application in robust control design.

Practical application of model uncertainty bounds obtained by current techniques for uncertainty bounding is hampered by the fact that these bounds often turn out to be unrealistically large. It is analysed which effects (should) contribute to a model uncertainty bound. These effects can be divided into three categories: (a) models used for robust control are taken to be linear, time-invariant and of low order, while the underlying process is not; (b) experimental data of a process gives only an incomplete and uncertain account of the process behaviour due to finite data length, finite sampling frequency and unknown external factors as noise and disturbances; (c) some knowledge that is available of the process can be expressed only approximately or not at all in the model and/or the uncertainty bounds. This leads to conservatism. The practical motivation for application of robust control is mainly to cope with the influence of category (a). Current model uncertainty bounds take typically the influence of categories (b) and (c) into account.

To study the interplay between different factors involved in a model uncertainty bounding procedure, a general framework has been developed in this thesis. Contrary to other frameworks in which identification procedures are "embedded," this framework takes great care not to put unrealistic restrictions on the process and/or its noise. Some of the properties that all uncertainty bounding algorithms (should) have in common are investigated. Furthermore the relation that should exist between noise, disturbances and simplifications such as time-invariance and linearity on the one hand and model uncertainty bounds on the other is clarified.

Based on the observations made above, an algorithm is proposed that splits model uncertainty for MIMO systems in so-called structured and unstructured parts. The structured part is bounded in a detailed way and is intended to capture the changes in process dynamics that occur if the process is operated in several operating points. In the unstructured part, all other sources of model uncertainty are lumped together and are bounded in a much less detailed way. If the structured part describes the dominating factors in the uncertainty of
a process, the remaining unstructured part will be much smaller than would be the case without a separate description of the dominating factors. In this way the combination of structured and unstructured model uncertainty bounds can give a better, less conservative description of the total model uncertainty. At the same time, the uncertainty bounds represent more aspects of model uncertainty that are relevant for robust control than most nther current bounds do.

The dominating contributions to model uncertainty can be estimated from data sets obtained in different operating points. It is shown how these contributions can be excluded from the unstructured model uncertainty bound. Deterministic assumptions are made for the noise and disturbances acting on the process. Several extensions to the basic algorithm have been developed, allowing the use of more prior knowledge in the uncertainty bounding or in the estimation of the structured error components.

The algorithm has been implemented in a $\mathrm{C}++$ program. After careful consideration of implementation in Matlab, it was decided that Matlab provides insufficient support to implement the algorithm with the full flexibility that is required. A generally applicable library was developed to support the implementation. From simple objects like vectors and sets of integers, more complex data structures are developed, such as signals, model sets, identification sessions, etc. These tools were developed and used to implement the algorithm, but they have their value in the field of computer aided system identification in their own right.

The algorithm is tested by means of a number of simulation examples. It is also applied to practical data taken from an asynchronons machine. From the results it can be concluded, that the algorithm can indeed reduce the unstructured model uncertainty significantly by splitting off a limited number of structured error components that were determined by the algorithm. It can also find an approximate description for error structures that it can not represent exactly, leading again to a significant decrease of the unstructured error.

Both the structured and the unstructured error bounds are very sensitive to the way in which the effect of undermodelling is bounded. Also very high quality data is needed to distinguish structured contributions to the error from unstructured ones. It seems that the signal to noise ratio should be of the order 40 dB or better for successful application of the algorithm in its present form. As the noise exceeds this level for the asynchronous motor, using separate structured error components hardly reduces the remaining unstructured error in this example. This is mainly due to the deterministic (worst-case) assumptions on the noise and disturbances. It is recommended to investigate in future research the use of stochastic assumptions on the noise. This seems certainly possible and may result in smaller, but soft, error bounds, which, however, seem better applicable for realistic signal to noise ratios.

## Contents

Voorwoord ..... iv
Abstract ..... vii
1 Introduction ..... 5
1.1 Background and goals ..... 5
1.2 Overview of the contents ..... 8
1.3 Notation and terminology ..... 10
2 Model uncertainty and conservatism ..... 15
2.1 Introduction ..... 15
2.2 Origins of model uncertainty ..... 19
2.3 Different types of model error bounds ..... 28
2.4 Quality aspects of uncertainty bounds ..... 33
2.5 Summary ..... 34
3 A formal foundation for model uncertainty bounding ..... 35
3.1 Introduction ..... 35
3.2 Formalism ..... 35
3.3 Discussion ..... 62
3.4 Summary ..... 70
4 An algorithm for structured and unstructured error bounds ..... 71
4.1 Overview of contents ..... 71
4.2 Basic ideas ..... 72
4.3 Model parametrisation and uncertainty description ..... 75
4.4 Experimental data and prior knowledge ..... 80
4.5 Outline of the algorithm ..... 84
4.6 Review of systems and models ..... 87
4.7 Step 1-4: obtaining a MIMO basis ..... 88
4.8 Step 5: estimating auxiliary models ..... 91
4.9 Step 6 \& 7: estimating the error structure ..... 101
4.10 Step 8: translating data and prior knowledge ..... 106
4.11 Step 9: bounding the local structured error ..... 113
4.12 Step 10: bounding the local unstructured error ..... 116
4.13 Step 11: combining the local results ..... 140
4.14 Influence of the choice of the basis ..... 142
4.15 Summary ..... 143
5 Extensions to the basic algorithm ..... 149
5.1 Introduction ..... 149
5.2 Estimating initial conditions ..... 150
5.3 Regularisation of Least Squares estimates ..... 158
5.4 Prior knowledge of static gains ..... 163
5.5 Prior knowledge of complex gains for arbitrary frequencies ..... 164
5.6 Weighting of parameters ..... 172
5.7 Other ways to determine the error structure ..... 186
5.8 Unstable systems and closed loop experiments ..... 191
6 Examples and case studies ..... 199
6.1 Overview ..... 199
6.2 A simple SISO simulation example ..... 200
6.3 A more complex SISO simulation example ..... 206
6.4 A MIMO simulation example ..... 219
6.5 Case study: Asynchronous motor ..... 229
6.6 Conclusions ..... 241
7 Software components for CASID ..... 243
7.1 Introduction ..... 243
7.2 Object-oriented aspects of C++ ..... 249
7.3 Matrices and related data structures ..... 252
7.4 Building on vectors and matrices ..... 265
7.5 Keeping all data of an identification session ..... 272
7.6 User interface ..... 273
7.7 Performance ..... 274
8 Conclusions and recommendations ..... 277
8.1 Conclusions ..... 277
8.2 Recommendations ..... 280
A Overview of notation ..... 285
A. 1 Latin symbols ..... 285
A. 2 Greek symbols ..... 288
A. 3 Other symbols ..... 289
A. 4 Other notations ..... 289
A. 5 Acronyms and abbreviations ..... 289
B System based orthonormal basis functions ..... 291
B. 1 Generating a basis from a general system ..... 292
B. 2 Properties of system based orthonormal basis functions ..... 294
B. 3 Transients of a generalised FIR model ..... 296
C Application of the formal framework ..... 303
Bibliography ..... 309
Samenvatting ..... 315
Curriculum vitae ..... 317

## 1

## Introduction

1.1 Background and goals
1.3 Notation and terminology
1.2 Overview of the contents

### 1.1 Background and goals

The word "model" has different interpretations. If something is shaped after a model, the model represents something that reality is supposed to resemble as closely as possible. In other contexts, models try to describe aspects of reality in a simplified form. Scale models of a car used in wind tunnel experiments are examples of this. In this case the model should resemble the geometry of the car as closely as possible.

Mathematical models of the behaviour of processes usually fall in the latter category. They are used to gain a better understanding of the process dynamics, to formulate expectations for its future behaviour or to derive ways to interact with the process to obtain a more desirable behaviour of it. These usages are normally referred to as analysis, prediction and control.

Control strategies are developed for processes to make them behave more accurately, faster and/or more reliably. In industrial processes, control may further result in better safety, less demand of raw material and energy, less environmental load, higher quality of the resulting product or fewer off-spec products, in short: more economic operation of the process. Consequently control is applied heavily in aerospace, process industry, chemical industry, consumer electronics and so on.

Modern control is almost exclusively based on mathematical models of the process. Instead of designing a controller for the process, a controller is designed based on the model of the process. Often it is assumed that the model is identical to the process. This is called the certainty equivalence principle. Because the process is not identical to the plant, a controller that performs well on the model may exhibit poor performance on the actual process or even destabilise that process. To cope with this problem, control design strategies were developed (and still are) that can recognise that the model is only an
approximation of the process. If a bound on the mismatch between the process and the model can be given and if a controller can be designed that performs well for all processes whose mismatch with the model does not exceed that bound, the resulting controller is called robust. Aiming for control robustness is further motivated by the practical observation that the behaviour of a process changes for different operating conditions, under the influence of external factors, aging and wear.

White-box models for a process are derived by describing the physical laws governing the system. Black-box models on the other hand are derived from experimental data of the process; the physical mechanisms of the process are basically not taken into account explicitly, i.e. the process is treated as a black box. Reliable methods exist for the derivation of black-box models from data obtained in properly designed experiments. In the last few years these methods were supplemented by methods to derive a bound on the mismatch between these models and the actual process. This mismatch will further be called the model error or model uncertainty. The process model and the uncertainty bound together with the specifications for the controlled process behaviour form all required ingredients for the formulation of a robust control design problem. Having techniques available that subsequently solve such a problem, it may seem that nothing stops us from applying robust control in practice.

Alas, robustness is achieved at the expense of performance. If one can rely on the correctness of a model, in other words if the model uncertainty is small, a controller can be designed that pushes performance of the model to its limits. However, the larger the uncertainty becomes, the more cautious the control design has to be. A controller that improves performance for one process may decrease it for another. To guarantee a certain level of performance for all processes that fall within the uncertainty bounds, performance for the model has to be sacrificed and probably also for processes close to the model in some sense.

Unfortunately it has turned out, that the uncertainty bounds that are found by the aforementioned techniques are often so big in practice, that the degradation of performance is unacceptable. Moreover, based on practical experience and physical insight it is often felt that the uncertainty bounds are unrealistically large. Apparently the uncertainty bounds are often taken to be unnecessarily large and consequently degrade performance substantially. This is referred to as conservatism and the resulting bounds are said to be conservative. Conservatism has been recognised in the field of uncertainty bounding as a serious problem which deserves further attention.

The problem area of this thesis is in line with the previous observations. A preliminary statement of the goal of this thesis is:

Reduce the conservatism in model error bounds that are derived from experimental data and prior knowledge.

Prior knowledge is mentioned explicitly as a means to bound model errors: if prior knowledge may be used to decide that model error bounds are unrealistically large, this knowledge might as well be used to reduce these bounds.

The preliminary formulation of the goal of this thesis has been our compass during the research reported in this thesis. However, it is such a broad statement that it deserves some further refinement.

If one wants to develop tighter bounds for model uncertainty, a clear understanding of what should be contained by the uncertainty bounds and what can be excluded from them is a prerequisite. Moreover it is felt that in the design of all uncertainty bounding techniques some choices are made that actually go beyond that specific technique. These choices reflect a general statement about certain aspects involved in the derivation of uncertainty bounds from experimental data and prior knowledge. Different, contradicting statements can be distilled from existing uncertainty bounding techniques. A consideration of such issues in a context that is not confined to a particular model error bounding technique is therefore desirable.

The first part of a more refined statement of the goal of this thesis is:

Provide a fundamental analysis of the factors that lead to model uncertainty in the context of black-box identification for robust control design. Moreover, analyse the steps that are involved in deriving error bounds from experimental data and prior knowledge. Provide a juctification for the choices that are generally made in the design of such a technique.

The second part follows naturally from the first part:

Based on the results of the aforementioned analysis, adapt existing techniques for model error bounding for application in robust control design or develop new ones such that they fit better to the requirements of robust control design in general and are less conservative in particular.

A short preview of the methodology that was developed to achieve this second part of the goal is given in the overview of the contents in the next section.

It has been explicitly mentioned that the intended application for the uncertainty bounds is robust control design. This does not mean that the results are irrelevant for other fields of application. However, generalisation of the interpretation of results presented in the subsequent chapters to these fields is considered outside the scope of this thesis.

### 1.2 Overview of the contents

The first part of the stated goal of this thesis is addressed in chapters 2 and 3. Chapter 2 provides a general consideration of factors contributing to model uncertainty. Recognising which factors make the uncertainty bounds larger than they need to be is the first step towards deriving uncertainty bounds that are as tight as possible. For practically useful uncertainty bounds, tightness of the bound is not the only quality criterion. Other aspects determining the quality of an uncertainty bounding procedure are considered in this chapter as well, so that a well balanced compromise between all these aspects can be made.

A framework for model uncertainty bounding is given in chapter 3. After a presentation of the framework, the relation between different factors involved in model uncertainty bounding is investigated. Because the framework is not restricted to a particular technique for uncertainty bounding, the conclusions that can be drawn from this study apply to many current and future techniques for uncertainty bounding. Moreover, the framework takes great care not to impose unrealistic assumptions such as linearity or time-invariance on the true process.

Chapters 4,5 and 6 form the main contribution of this thesis. These chapters provide a detailed presentation of a new algorithm for model uncertainty bounding. This algorithm splits the model uncertainty in so-called structured and unstructured parts. The structured parts are bounded in a very detailed way. The model uncertainty that is not accounted for by the structured parts is lumped together in the unstructured part and bounded in a much less detailed way. Provided the structured parts represent the dominating sources of model uncertainty, this separation leads to a tightly bounded representation of the model uncertainty. The algorithm is developed for MIMO systems.

Part of the machinery used in the algorithm is due to Hakvoort (1994, chapter 4). In chapter 4 no modifications are made to these procedures. Chapter 5 proposes some improvements and also extends other parts of the algorithm. These extensions are relegated to a different chapter to avoid unnecessary distraction from the main ideas by technical details in chapter 4. Nevertheless chapter 4 is full of concepts and symbols. An example developed along with the theory throughout the chapter tries to help the reader in getting accustomed to these concepts. At the beginning of each section is also a summary of the algorithm indicating what part of the algorithm is being considered in that section.

Figure 1.1 shows the benefit that may be obtained from using separate structured error components. In this simple example, a process is operated in two operating points. The Nyquist plots of the transfer functions in these two operating points are plotted using continuous lines and are marked with x's. Between these two Nyquist plots is the Nyquist plot of the nominal model for


Figure 1.1: Potential benefit of using structured errors
both working points. The light shaded areas give the uncertainty for a number of frequencies if no structured error components are used. The dark shaded areas give the unstructured uncertainty that remains if only one structured error component is bounded separately. The size of the structured error component for these frequencies is indicated by dashed lines.

This example is discussed in more detail in chapter 6. A more complex SISO simulation example and a MIMO simulation example are presented next in this chapter. In the last example presented in this chapter the algorithm is applied to experimental data obtained from an asynchronous motor.

Chapter 7 presents a side-product of the algorithm. The algorithm was not implemented in Matlab but in a C++ program. To support this implementation, a library of "classes" was developed. With these classes the programmer is no longer forced to code his algorithms with matrices and vectors, he can also use objects with a higher level of abstraction, such as signals, model sets, identification sessions, etc. An argumentation for this choice as well as an overview of the design of these classes is given in this chapter.

In chapter 8 conclusions are drawn and recommendations for future research are given.

The main text of the thesis can be separated in three parts. The first part consists of chapters 2 and 3 , the second of chapters 4,5 and 6 and the third of chapter 7. These parts can be read more or less independently of each other, although the first part gives background information that was important for the design of the algorithm in the second part. Moreover, the complexity of the
second part gives some further justification for the choices made in the third.
In the remainder of this introduction, some of the notation and terminology used in this thesis is introduced. The notation is fairly standard, but some conventions are used that can not be considered standard practice. A list of most symbols used in this thesis is given in appendix $A$.

### 1.3 Notation and terminology

### 1.3.1 Sets and spaces

Throughout this thesis, $\mathbb{N}$ will be used to denote the set of non-negative integers, including $0 . \mathbb{Z}$ is the set of all integers. $\mathbb{R}$ denotes the set of real numbers and $\mathbb{C}$ the set of complex numbers. $\mathscr{Z}_{+}$and $\mathbb{R}_{+}$are the subsets of $\mathbb{Z}$ and $\mathbb{R}$ consisting of positive values only. $\mathbb{Z}_{-}$and $\mathbb{R}_{-}$are the equivalent subsets of negative values.

For $z \in \mathbb{C}, \operatorname{Re} z$ denotes the real part of $z, \operatorname{Im} z$ is used for the imaginary part. The absolute value or modulus of $z$ is denoted $|z|$. For $z \in \mathbb{C} \backslash\{0\}$ the argument of $z$ is represented by $\arg z$. For $z=0$ the argument of $z$ is undetermined. For $z \neq 0$, the argument of $z$ is determined modulo $2 \pi . z^{*}$ is used for the complex conjugate of $z$.

For $\mathbb{F} \in\{\mathbb{R}, \mathbb{C}\}$ and $n \in \mathbb{Z}_{+}, \mathbb{F}^{n}$ denotes the field of vectors with $n$ elements taking their values in $\mathbb{F}$. $e_{i}^{n}$ is the $i$ th unit vector in $\mathbb{F}^{n}$. The span of $a, b, c$ with $a, b, c \in \mathbb{F}^{n}$ is denoted $\langle a, b, c\rangle . \mathbb{F}^{m \times n}$ is the field of $m \times n$ matrices with entries in $\mathbb{F}$. We identify $\mathbb{F}^{n}$ with $F^{n \times 1}$, so vectors are column vectors. The identity matrix $I_{n} \in \mathbb{R}^{n \times n}$ is given by

$$
\left[e_{1}^{n}, \ldots, e_{n}^{n}\right]
$$

For $M \in \mathbb{F}^{m \times n}, M_{i *} \in \mathbb{F}^{1 \times n}$ denotes the $i$ th row of $M, M_{* j} \in \mathbb{F}^{m}$ denotes the $j$ th column. A matrix $M \in \mathbb{F}^{m \times n}$ is said to be tall if $m \geq n$ and said to be fat if $n \geq m$. Note that square matrices are both tall and fat.

For $n \in \mathbb{Z}$, the bold symbol $n$ is a shorthand notation for the set $\{1, \ldots, n\}$. For $n<1, \boldsymbol{n}$ is the empty set. This convention will also be used for other symbols. For any set $\mathcal{S}, 2^{\mathcal{S}}$ denotes the power set of $\mathcal{S}$, i.e. the set of all subsets of $\mathcal{S}$.

Let $\left\{X^{i}\right\}_{i=1}^{n}$ be a sequence of $n$ matrices with equal number of columns. Then

$$
\underset{i \in \boldsymbol{n}}{\operatorname{stack} X^{i}}:=\left[\begin{array}{c}
X^{1} \\
\vdots \\
X^{n}
\end{array}\right]
$$

the operator "stack" stacks the matrices in the sequence on which it operates on top of each other, where the first matrices in the sequence end up top-most.

Conversely, if $\left\{X^{i}\right\}_{i=1}^{n}$ is a sequence of $n$ matrices with equal number of rows, then

$$
\operatorname{sbs}_{i \in n} X^{i}:=\left[\begin{array}{lll}
X^{1} & \ldots & X^{n}
\end{array}\right]
$$

"sbs" puts the elements in the set on which it operates side-by-side in a matrix. For any sequence of matrices $\left\{X^{i}\right\}_{i=1}^{n}$,

$$
\operatorname{diag}_{i \in \boldsymbol{n}} X^{i}:=\left[\begin{array}{lll}
X^{1} & & \\
& \ddots & \\
& & X^{n}
\end{array}\right]
$$

is the block diagonal matrix with the $X^{i}$ on its diagonal.
For $a, b \in \mathbb{R}^{q}$, the inner product of $a$ and $b$ is denoted $(a \mid b)$ and equals

$$
(a \mid b):=\sum_{k=1}^{q} a_{k} b_{k}
$$

The transpose of a matrix $A$ is denoted $A^{T} . A^{*}$ denotes the complex conjugate transpose. For real matrices $A, A^{T}=A^{*}$.

For a linear operator $P$, the notation $\operatorname{img} P$ is used for the image space of $P$. For a matrix $P$ the image space is taken equal to the span of the columns of $P$. With a matrix $P \in \mathbb{F}^{q \times p}$ is, as usual, associated an operator $P^{\prime}, P^{\prime}$ : $\mathbb{F}^{p} \rightarrow \mathbb{F}^{q}, x \mapsto P x$. It holds img $P \equiv \operatorname{img} P^{\prime}$.

For $P \in \mathbb{R}^{q \times p}$ or $P \in \mathbb{C}^{q \times p}$, rank $P$ shall denote the rank of the matrix $P$.

### 1.3.2 Signals, systems and norms

The symbols $\ell_{2}, \mathcal{L}_{2}, h_{2}$ and $\mathcal{H}_{2}$ have their usual meaning:
$\ell_{2}^{q \times p}$ will be used for the space of functions defined on $\mathbb{Z}$, taking values in $\mathbb{R}^{q \times p}$ that are square summable:

$$
\ell_{2}^{q \times p}:=\left\{f \mid f: \mathbb{Z} \rightarrow \mathbb{R}^{q \times p}, t \mapsto f(t) ; \quad\left(\sum_{\iota \in \mathbb{Z}} \sum_{i=1}^{p} f_{* i}^{T}(t) f_{* i}(t)\right)<\infty\right\}
$$

These functions will be referred to as time-domain signals or $\ell_{2}$-sequences. $\mathcal{L}_{2}^{q \times p}$ is the space of functions defined on $\mathbb{C}$ taking values in $\mathbb{C}^{q \times p}$ that are square integrable on the unit circle:

$$
\mathcal{L}_{2}^{q \times p}:=\left\{F \mid F: \mathbb{C} \rightarrow \mathbb{C}^{q \times p}, z \mapsto F(z) ; \quad \oint_{|z|=1} \operatorname{trace} F^{*}(z) F(z) d z<\infty\right\}
$$

These functions will be called frequency-domain functions. To emphasize this, signals $f \in \ell_{2}^{q \times p}$ will in general be denoted $f(t)$ and signals $F \in \mathcal{L}_{2}^{q \times p}$ will be denoted $F(z)$. Also, symbols representing elements of $\ell_{2}$ will in general be lower case letters and elements of $\mathcal{L}_{2}$ will be represented by capital letters.

Remark 1.1 It is important to note, that $t$ and $z$ are indeterminates, unless explicitly stated otherwise. This means $t$ and $z$ do not represent a single real or complex value, but a whole set of values. This enables us to use the convention to denote signals $f \in \ell_{9}^{q \times p}$ as $f(t)$ and signals $F \in \mathcal{L}_{2}^{q \times p}$ as $F(z): f(t)$ denotes in general the signal $f$, whereas, say, $f\left(t^{\prime}\right)$ denotes the vauue of the signal $f$ at time $t=t^{\prime}$. As an example of an explicit exception, in $\sum_{t \in \mathbb{Z}} f(t)$ the notation $f(t)$ does refer to the value of $f$ at time $t$, because it is clear from " $\sum_{t \in \mathbb{Z}}$ ", that $t$ is not indeterminate in this case.

With each element $f(t)$ of $\ell_{2}^{q \times p}$ is associated an element $F(z)$ of $\mathcal{L}_{2}^{q \times p}$ through the Z-transform:

$$
F(z)=\sum_{t \in \mathbb{Z}} f(t) z^{-t}
$$

For such a pair $(f(t), F(z)), F(z)$ is called the Z-transform of $f(t), f(t)$ is the inverse Z-transform of $F(z)$. The values $f\left(t^{\prime}\right), t^{\prime} \in \mathbb{Z}$ are the Laurent series expansion coefficients of $F(z)$ around $z=0$.
$\mathcal{H}_{2}^{q \times p}$ is the restriction of $\mathcal{L}_{2}^{q \times p}$ to those functions that are analytic outside the unit disc, including at $\infty$. $h_{2}^{q \times p}$ is the set of inverse Z-transforms of $\mathcal{H}_{2}^{q \times p}$. For every $f(t) \in h_{2}^{q \times p}$ it holds that $f\left(t^{\prime}\right)=0$ for $t^{\prime}<0$.
$\zeta$ will denote the forward-shift operator or left-shift operator:

$$
\zeta: \ell_{2}^{q \times p} \rightarrow \ell_{2}^{q \times p}, f(t) \mapsto f(t+1)
$$

We can associate with every $f(t) \in h_{2}^{q \times p}$ an operator $F(\zeta)$ :

$$
\begin{equation*}
F(\zeta): \ell_{2}^{p} \rightarrow \ell_{2}^{q}, u(t) \mapsto y(t):=\sum_{\tau \in \mathbb{N}} f(\tau)\left[\zeta^{-\tau} u\right](t) \tag{1.1}
\end{equation*}
$$

If $f(t) \in h_{2}^{q \times p}, F(z)$ will be used to denote its Z-transform and $F(\zeta)$ to denote the operator associated to it through (1.1). Often this will not be mentioned explicitly. Moreover, we will not be very strict in distinguishing the three entities. They are seen more as different representations for the same thing. $f(t)$ will be called the impulse response of $F(z)$ or $F(\zeta)$ and $F(z)$ is called the transfer function associated with $F(\zeta)$ and $f(t)$.

Similarly to the previous section, for all $\mathcal{F} \in\left\{\ell_{2}, \mathcal{L}_{2}, h_{2}, \mathcal{H}_{2}\right\}, \mathcal{F}^{q}$ is used as a shorthand notation for $\mathcal{F}^{q \times 1}$ and $\mathcal{F}$ for $\mathcal{F}^{1}$.

With $\ell_{2}^{q \times p}$ is associated the norm $\|\cdot\|_{\ell_{2}}$ :

$$
\forall f(t) \in \ell_{2}^{q \times p} \quad\|f(t)\|_{\ell_{2}}:=\sqrt{\sum_{t \in \mathbb{Z}} \sum_{i=1}^{p} f_{* i}^{T}(t) f_{* i}(t)}
$$

The norm $\|.\|_{h_{2}}$ is identical. As $f\left(t^{\prime}\right)=0$ for $f(t) \in h_{2}^{q \times p}$ and $t^{\prime}<0$ the norm $\|\cdot\|_{h_{2}}$ reduces to

$$
\forall f(t) \in h_{2}^{q \times p} \quad\|f(t)\|_{h_{2}}:=\sqrt{\sum_{t \in \mathbb{N}} \sum_{i=1}^{p} f_{* i}^{T}(t) f_{* i}(t)}
$$

The norms for $\mathcal{L}_{2}^{q \times p}$ and $\mathcal{H}_{2}^{q \times p}$ are obtained by integrating over the unit circle as follows:

$$
\begin{aligned}
\forall F(z) \in \mathcal{L}_{2}^{q \times p} \quad\|F(z)\|_{\mathcal{L}_{2}} & :=\frac{1}{2 \pi} \oint_{|z|=1} \operatorname{trace} F^{*}(z) F(z) \frac{1}{z} d z \\
& =\frac{1}{2 \pi} \int_{-\pi}^{\pi} \operatorname{trace} F^{*}\left(e^{j \omega}\right) F\left(e^{j \omega}\right) d \omega
\end{aligned}
$$

and

$$
\begin{aligned}
\forall F(z) \in \mathcal{H}_{2}^{q \times p} \quad\|F(z)\|_{\mathcal{H}_{2}} & :=\frac{1}{2 \pi} \oint_{|z|=1} \operatorname{trace} F^{*}(z) F(z) \frac{1}{z} d z \\
& =\frac{1}{2 \pi} \int_{-\pi}^{\pi} \operatorname{trace} F^{*}\left(e^{j \omega}\right) F\left(e^{j \omega}\right) d \omega
\end{aligned}
$$

The $\ell_{1}$-norm of $f(t) \in\left(\mathbb{R}^{q \times p}\right)^{\mathbb{N}}$ is defined as

$$
\|f(t)\|_{\ell_{1}}:=\max _{i \in \boldsymbol{q}} \sum_{j \in \boldsymbol{p}} \sum_{t \in \mathbb{N}}\left|f_{i j}(t)\right| .
$$

Note that not every $f(t) \in\left(\mathbb{R}^{q \times p}\right)^{\mathbb{N}}$ has a well-defined (finite) $\ell_{1}$-norm. In this thesis the notation $\|f(t)\|_{\ell_{1}}$ will implicitly assume that the norm is well-defined. The $\ell_{\infty}$-norm of $f(t) \in\left(\mathbb{R}^{p}\right)^{Z}$ is defined as

$$
\|f(t)\|_{\ell_{\infty}}:=\max _{i \in p} \max _{t \in \mathbb{Z}}\left|f_{i}(t)\right|
$$

Finally, the $\mathcal{H}_{\infty}$-norm for $F(z) \in \mathcal{H}_{\infty}^{q \times p}$ is defined as

$$
\forall F(z) \in \mathcal{H}_{\infty}^{q \times p}, \quad\|F(z)\|_{\mathcal{H}_{\infty}}=\sup _{\omega \in[0, \pi]} \sigma_{\max }\left(F\left(e^{j \omega}\right)\right)
$$

where $\sigma_{\max }(M)$ is the largest singular value of $M \in \mathbb{R}^{q \times p}$.
If the time domain signals $f(t) \in \ell_{2}$ are to be interpreted as "real life" signals, one should consider the time $t$ to be normalised to the length of the sampling interval. This means that the frequency domain signals $F\left(e^{j \omega}\right) \in \mathcal{L}_{2}$ are expressed in normalised frequency: $\omega=\pi$ corresponds to half the sampling frequency. This normalisation is also implied if it is claimed that a system has large time constants. "Large" means in this statement "large compared to the sampling time."

The subset of those $f(t) \in h_{2}^{q \times p}$ that satisfy

$$
\exists N \in \mathbb{N}, \forall t^{\prime}>N, \quad f\left(t^{\prime}\right)=0
$$

will be called the set of finite impulse response (FIR) models. Strictly speaking, if $f_{1}(t), f_{2}(t) \in h_{2}^{q \times p}$ are not FIR models, then their impulse responses are infinitely long and one can not say that the impulse response of one is longer or shorter than that of the other. However, with length of the impulse response will somewhat ambiguously be meant the smallest $N$ such that

$$
\begin{equation*}
\sqrt{\sum_{t=0}^{N} \sum_{i=1}^{p} f_{* i}^{T}(t) f_{* i}(t)} \geq p\|f(t)\|_{h_{2}} \tag{1.2}
\end{equation*}
$$

where $p \in(0,1)$ is some sensibly chosen value. The ambiguity in the description lies in the fact that $p$ is not fixed. With this interpretation of length of impulse response we can say that the impulse response of $f_{1}(t)$ is longer or shorter than that of $f_{2}(t)$. If $f_{1}(t)$ has a shorter impulse response than $f_{2}(t)$, we will call $f_{1}(t)$ faster than $f_{2}(t) . F_{1}(z)$ and $F_{1}(\zeta)$ are called faster than $F_{2}(z)$ and $F_{2}(\zeta)$ resp. if $f_{1}(t)$ is faster than $f_{2}(t)$.

As a consequence of this, the statement that the impulse response of some $f(t) \in h_{2}^{q \times p}$ is longer than $N$ samples, $N \in \mathbb{N}$, should be interpreted such, that

$$
\sqrt{\sum_{i=0}^{N} \sum_{i=1}^{p} f_{* i}^{T}(t) f_{* i}(t)}<p\|f(t)\|_{h_{2}}
$$

where $p$ is as in the previous paragraph.
A system $F(\zeta) \in \mathcal{L}_{2}^{q \times p}$ is called stable if a bounded sequence $u(t)$ maps to a bounded sequence $[F(\zeta) u](t)$. This is often referred to as BIBO stability.
$\delta(t)$ denotes the Kronecker delta function.

$$
\delta: \mathbb{Z} \rightarrow \mathbb{R}, t \mapsto \begin{cases}1 & t=0 \\ 0 & t \neq 0\end{cases}
$$

We will also use $\delta(t)$ and $\delta(t-n), n \in \mathbb{Z}$ to denote the restriction of $\delta(t)$ and $\delta(t-n)$ to $\mathbb{N}$. With this interpretation, $\delta(t), \delta(t-n) \in h_{2}$. However, usage will be clear from the context.

Further notation will be introduced throughout the text. For a quick reminder of the meaning of a specific symbol the reader is referred to appendix $A$, which contains a list of most symbols used in this thesis, with a short description for each symbol and, where appropriate, a reference to the page on which the symbol is explained or introduced.

## 2

## Model uncertainty and conservatism

2.1 Introduction<br>2.2 Origins of model uncertainty<br>2.3 Different types of model error bounds

### 2.4 Quality aspects of uncertainty bounds <br> 2.5 Summary

### 2.1 Introduction

In the late eighties, early nineties, important progress was made in the field of robust control design. A solution to the "standard" $\mathcal{H}_{\infty}$ problem was presented in (Doyle et al., 1988; Doyle et al., 1989). These results were subsequently refined by the introduction of the structured singular value $\mu$, leading to socalled $\mu$-analysis and $\mu$-synthesis, (Packard, 1988; Packard and Doyle, 1993).

Roughly speaking these results mean that in the configuration of figure 2.1 a controller $K$ can be designed for a model $G$ that is perturbed by an unknown but bounded model uncertainty $\Delta$. The techniques can be applied such, that


Figure 2.1: Generalised plant, controller and uncertainty interconnection in robust control design
a certain performance level can be guaranteed for any $\Delta$ such that

$$
\begin{equation*}
\|\Delta\|_{\mathcal{H}_{\infty}}<d_{\Delta} \tag{2.1}
\end{equation*}
$$

(Some assumptions are made here. The most important ones are, first of all, that the model $G$ is linear, time-invariant and of finite McMillan degree. Further, it is required that a linear, time-invariant controller $K$ exists that stabilises the interconnection of $G$ and $\Delta$ for all $\Delta$ satisfying (2.1). Finally. performance is assumed to be measured in terms of the $\mathcal{H}_{\infty}$-norm of the operator depicted in figure 2.1, mapping $\ell_{2}$ signals $w$ to $\ell_{2}$ signals $z$.)

Maintaining performance even if the true plant deviates from its nominal model, normally referred to as performance robustness, is to be considered a desirable property: in many practical situations physical insight may lead to the conclusion that under different circumstances the controller $K$ will "face" a different plant $G^{\prime}$, where $G^{\prime}$ is the interconnection of $G$ and some value of $\Delta$ saticfying (2.1). Think for cxample of a controller for a chemical plant. It would be attractive if one can take explicitly into account in the control design that the plant reacts differently in summer, when the atmospheric temperature is relatively high, than in winter. Other changes in the process behaviour may be induced by degradation of catalysts, wear and aging of the plant. Many more influences can be thought of that a controller should be robust against, for this particular example as well as for other types of plants. This issue will be further investigated in the remainder of this chapter.

Following the results in the field of robust control design, the system identification community picked up the ball and set off to develop techniques to estimate an appropriate value for $d_{\Delta}$ in (2.1) or other bounds on the model uncertainty that can be handled by $\mathcal{H}_{\infty}$ control design or $\mu$-synthesis. In (Wahlberg and Ljung, 1992) four main streams were distinguished in this research:

1. Identification in $\mathcal{H}_{\infty}$, as developed in (Helmicki et al., 1989; Helmicki et al., 1990a; Helmicki et al., 1990b; Helmicki et al., 1991; Mäkilä and Partington, 1992; Mäkilä et al., 1994; Mäkilä et al., 1995; Partington, 1991; Partington, 1992)
2. Methods based on the traditional, statistical approach, complemented by an estimate of the bias error. (Goodwin and Salgado, 1989; Goodwin et al., 1992; Kosut, 1988).
3. Many publications have appeared in the field of set membership algorithms, which could better have been called parameter set membership algorithms, see (Gutman, 1988; Kosut et al., 1992; Walter and PietLahanier, 1990) and the references therein.
4. Research not so much concerned with uncertainty bounding but rather with improving the link between identification and control design was
reported in (Schrama and Van den Hof, 1992; Schrama, 1992; Gevers, 1991; Gevers, 1993; Van den Hof and Schrama, 1995).

Although the aim of this chapter is not to give a complete review of literature on the topic, other developments will be mentioned in the remainder of this chapter.

Even though the missing requirement for robust control design, a quantitative bound on the model uncertainty, was more or less fulfilled by these developments, the case was not yet solved and closed. Combination of the estimated uncertainty bounds and the robust control design techniques often gave disappointing results. To achieve sufficient robustness, some nominal performance has to be sacrificed. The performance degradation that was observed in practice was often considered unacceptably large. Because the control design techniques yield the "optimal" controller, or at least optimal to within a specified margin, the most obvious way to try to improve this situation is to enhance the estimation of the uncertainty bounds to yield tighter and/or more detailed bounds. This assumes implicitly that the bounds estimated so far are not as tight as possible, in other words that the estimated bounds are conservative.

Clearly, much effort was and is spent to link the development of robust control design with the development of model uncertainty estimation. The link that is developed resembles to some extent a tunnel that is built from two ends. Unfortunately the two teams of construction workers that are supposed to meet in the middle might very well miss each other or at best have two mis-aligned tunnel halves. This is illustrated by the following, fictitious conversation between a model builder and a control designer that is taken from (Wahlberg and Ljung, 1992), in which the model builder defends his uncertainty bounds against the scepticism of the control designer:
MB: Here is my model statement: A nominal Nyquist plot with confidence bounds.
CDE: Fine, but can you guarantee that the true Nyquist plot is not outside these bounds.

MB: Well, all I can say is that there is no evidence in the data of such a risk. CDE: But I want to be positive.
MB: One can never be $100 \%$ positive about statements about the real world. You can increase your confidence, though, by collecting more observations under varying conditions and see if my model is still unfalsified with respect to these data.

CDE: I see here that you have assumed that the disturbances are random. I don't believe in that nonsense. Disturbances can be very deterministic. Then the model discrepancy could be a lot worse.
MB: Well, all I can say is that there is no evidence in the particular data set we worked with that we need to be so pessimistic.

The model builder does a good job in this conversation pointing out that the uncertainty bounds are an extrapolation or generalisation of a limited number of observations, but that we can be more or less confident about this generalisation. In fact, this is characteristic for any natural scientific research.

Up to this point, everything seems to be developing according to plan. Unfortunately, the conversation could have proceeded as follows.

CDE: It is my experience that the control performance varies with the quality of the raw material we use in our plant. What does the uncertainty bound say about that?

MB: There is no evidence in our data set that such an effect ever occurs.
CDE: I was already afraid you were going to say that. Do your uncertainty bounds apply if we are making a product with different specifications? Or if a rain shower hits the reactor?
MB: Sorry, I was only allowed to do a limited number of experimental runs. But don't worry, collecting extra data to establish the influence of these effects would only decrease (at least not increase) the size of my uncertainty bounds.

CDE: I am glad to hear that, because, quite frankly, the bounds I have now are so big that I can say goodbye to any performance in my robustly designed controller.

Note that this part of the conversation is not taken from (Wahlberg and Ljung, 1992) any more so that the complete responsibility for it lies with the author of this thesis.

Three important issues come up in this second part of the conversation:

- If the model builder and the control designer are talking about model uncertainty, they are thinking of different causes for this uncertainty.
- If one wants to make statements about a plant in different operating conditions, data and prior knowledge about the plant should be collected for all (relevant) operating conditions. Although this is in itself a trivial observation, it may imply that the experiments that were sufficient to derive good nominal models for a plant are insufficient to base tight, accurate uncertainty bounds upon.
- While experience has shown in many practical cases that linear timeinvariant models are a sufficiently good approximation of real plants, one has to keep firmly in mind during the uncertainty bounding that the true process is not linear time-invariant. A data set taken from the plant in one operating condition can not be used to reduce the uncertainty in the plant's behaviour in another operating condition. Instead, the former data set reveals uncertainty that is not present in the latter data set and vice versa. The model builder is confused about this in his last remark.

The issues raised above suggest to take a closer look at model uncertainty as such. The remainder of this chapter is concerned with this. First it will be investigated what effects (should) contribute to the model error bound. Then some fundamental decisions often made implicitly in a model error bounding algorithm are discussed. Finally some general quality aspects of model uncertainty bounds are reviewed.

To finish the tunnel metaphor discussed earlier: if two tunnel halves do not meet in the middle, it is not necessary to close both halves and start all over again. A small extra stretch is mostly sufficient to connect both ends.

### 2.2 Origins of model uncertainty

In this thesis "model error" and "model uncertainty" are used for the same concepts. Three categories of contributions to model error bounds will be distinguished:

- Undermodelling
- Strict sense uncertainty
- Wide sense conservatism

Strict sense uncertainty refers to a subset of all systems that can be represented exactly by the framework used for identification and uncertainty bounding, namely the subset of those systems that can not be falsified by the data or prior knowledge. If a plant is known to be a first order process with, say a DC gain and a pole location within certain known intervals, one could say that there is only model uncertainty and no model error if this plant is to be modelled by a first order transfer. In this case a subset of the set of first order transfers can be formulated having the properties that one of its members coincides with the real process and that for none of its members it can be concluded from data or prior knowledge that it is not the member coinciding with the real process. This subset represents in this example the strict sense uncertainty.

Strict sense conservatism refers to those models that

1. are consistent with the nominal model $(\mathrm{s})$ and the uncertainty bound,
2. can be falsified by the data and prior knowledge and
3. have a negative effect on the performance of the robustly designed controller.

Wide sense conservatism refers to those models that satisfy only 1 . and 2 . The extra requirement 3 . distinguishes strict sense conservatism from wide sense conservatism. For model uncertainty bounding for other purposes than robust
control design another requirement than the one mentioned under 3 . will be appropriate.

Undermodelling means that the true system can not be represented by a model in the model set. If one has an exactly known second order plant that is modelled by a first order model, then, strictly speaking, there is undermodelling (and possibly conservatism, depending on how this undermodelling is bounded), but no uncertainty: as the true plant is known, there is only one dynamical system that can not be falsified by the prior knowledge.

Examples of these will be given below.

### 2.2.1 Undermodelling

At the risk of stating the obvious, the most important cause for a model error is that the model used to describe the process is a gross simplification of that process. Nevertheless one may consider this issue a little further to get an impression of what kind of model errors can be expected.

The techniques for robust control design mentioned earlier require that the nominal model $G$ in figure 2.1 on page 15 is a linear time-invariant model. They do not require that $\Delta$ is linear time-invariant; $\Delta$ is only required to be an $\mathcal{H}_{\infty}$-norm bounded operator. This means that in the configuration of figure 2.1 the following implication holds for certain $d_{v}, d_{x} \in \mathbb{R}_{+}$:

$$
\|x\|<d_{x} \Longrightarrow\|v\|<d_{v}
$$

If $\Delta$ is not linear time-invariant, the interconnection of $G$ and $\Delta$ is not necessarily linear time-invariant either. Contrary to this generality allowed by $\mathcal{H}_{\infty^{-}}$ control design, most identification techniques rely on the fact that, at least in an operating point, the plant can be approximated sufficiently well by a linear time-invariant model. This also applies to the algorithm discussed in chapter 4 and 5. Allowing non-linear models $G$ and/or non-linear uncertainty blocks $\Delta$ would mean giving up most of the tools used in identification and model error bounding. Nevertheless, the restriction to use linear time-invariant models is a self-imposed restriction.

Given this restriction, one can distinguish the following aspects of undermodelling.

1) Non-linear relation between $u$ and $y$

Everyone doing identification with linear models is probably well aware of the fact that the true process is not linear. This may manifest itself in different ways. Most clearly present in an experimental data set is the non-linear response to the inputs whose response is to be modelled: a superposition of excitations does not give a response that is the corresponding superposition of responses, scaling an excitation does not lead to an accordingly scaled response. In chapter 3 a heuristic way to deal with, or rather limit, the effect of this phenomenon will be discussed.
2) Operating points

It is also well known, that a process behaves different in different operating points. In this context an operating point is interpreted as a set of operating conditions, such as ambient temperature, nominal values of inputs, type of raw material used, etc. Another effect of non-linearities is that unmodelled inputs may change the dynamics between the inputs and outputs that are modelled. For example, most physical processes change their behaviour if they are operated at different temperatures. In chemical plants, where temperature is an important parameter, this is clear, but the effect can also occur in electronic or mechanical devices. These differences in process behaviour are similar to those induced by a changing operating point. They are more difficult to deal with, because these changes can not be influenced, or only partially, and often the initial change that caused the shift in process behaviour is not directly measured.
3) Time variance

Not only is the linear part in linear, time-invariant models an oversimplification of reality. Processes are also time-varying, due to the effects of aging and wear. A catalyst in a chemical plant generally gets polluted and consequently less active during its life-span. A driving belt in a mechanical construction may stretch after extensive periods of heavy load, which will translate to a change in the dynamics of the transmission at hand. These effects are difficult to deal with using robust control if their quantitative effect is not reproducible or predictable: one can not do experiments to determine how a plant will respond in two years time, short of the cases in which another, similar but older plant is available for experimentation. One can try and gather sufficient information about the changes that can be expected from experience with other, similar plants. If the rate of change is not too high and the costs of experimentation, identification and controller design are not too high either, one could be better off re-identifying the plant and retuning the controller after a period of time.
4) Limited dynamic range

Forgetting for a moment about non-linearities and time variances, there is still plenty of room for undermodelling. In practice, experiments can cover only a limited dynamic range: the fastest response that can be identified is overbounded by hall the sampling frequency. The slowest dynamics that one can "see" are limited by the length of the experiments. During the experiment design, sampling frequency and experiment length are chosen such, that the relevant dynamics of the plant are covered by the data. Nevertheless the effects of "irrelevant" dynamics may be visible in the model error bounds.

If the effect of this kind of undermodelling can not be separated from the effects that are considered relevant, this visibility may be stronger than expected: if for example some high frequent behaviour of the plant is omitted from the transfer function of the model, this may be visible in the uncertainty of every sample of the impulse response of the plant. To interchange the role of time and frequency domain in this example, one may think of a large delay with a small associated gain that one chooses to omit from the process description. Although this simplification has a strongly localised effect in a time domain description of the plant, it will increase the uncertainty in the frequency domain for each and every frequency. A small simplification can consequently lead to a considerable increase in model uncertainty if the uncertainty description can not express the simplification properly.
5) Distributed and high-order dynamics

Not only may the dynamics of the true process have a wider range than covered by the experiments and consequently by the resulting model. It is also not uncommon that the actual process has distributed dynamics. The model is in general only of finite order. Consequently reactions in chemical plants are usually assumed to take place under homogeneity conditions. In flexible structures, only a few modes are considered and these modes can only be described approximately. Much more examples can be thought of where this simplification occurs.

It is my conviction, that the kind of model uncertainty discussed in this section is exactly what a robust control design wants to be robust against. However, model uncertainty bounding algorithms have concentrated on bounding the effect of the causes discussed in the next two sections. - This is not meant to imply that all causes of model uncertainty that will be discussed subsequently should be ignored in robust control design, although it would be nice if some of them actually could.

### 2.2.2 Strict sense uncertainty

The conclusions that can be drawn from experimental data concerning the true process behaviour is principally restricted by at least two effects.

- Firstly, the experimental data is finite. For many identification methods, asymptotic consistency results may be derived under "weak" or "reasonable" conditions - remarkably enough never called "realistic" conditions. These results state that the identified model tends to the true process if the experimental data length tends to infinity. Turning this statement around it says, that the identified model may not coincide with the true process for finite experimental data. Loosely speaking, some effects do
not "average out" on a finite data set that would on an infinitely long data set.

Another illustration of the influence of finite data can be found by considering the discrete Fourier transform (DFT). From any textbook on the subject it is known that the DFT may be interpreted as the transform of a periodic continuation of the finite length signal. Choosing a different continuation of the signal would result in different Fourier coefficients.

- The second cause of strict sense uncertainty is the possible corruption of the experimental data by noise, disturbances, responses to unmodelled inputs, etc. As already mentioned, under suitable conditions these effects may vanish if the length of the data set goes to infinity.

Remark 2.1 An obvious way to reduce the effect of finite data length is to extend the experiment duration. This is limited in practice by the allowed experimentation time, the computational load involved in estimating the model, etc. Also the process or the characteristics of its disturbances may change in very long experiments.

Input design for the identification experiment may play an important role in shaping the influence of noise: one may expect an accurate model in frequency regions and, in case of MIMO or MISO processes, for input directions in which the input is strongly exciting. A less accurate model can be expected in regions where the input is hardly exciting and the experimental data consequently barely contains information about the process response.

One may try to improve the signal to noise ratio in the experimental data by increasing the input amplitude. This is limited by actuator and operating constraints. Also one can expect a larger excitation of non-linearities for larger input amplitudes.

In conclusion, the effect of finite data length and noisy measurements may possibly be reduced yet never eliminated.

### 2.2.3 Wide sense conservatism

Contrary to the contributions discussed in the two previous sections, there is no fundamental impossibility to eliminate the effects of conservatism: these effects can be reduced or even removed by designing better model error bounding algorithms.

A common cause of conservatism is, that one has more knowledge about the noise that is affecting the process than can be handled by the algorithm. In parameter set estimation techniques for example, the true output $y_{t r}(t)$ of the process is generally assumed to be corrupted by a noise signal $\epsilon(t)$ with a known $\ell_{\infty}$ bound:

$$
\begin{equation*}
y(t)=y_{t r}(t)+\epsilon(t), \quad \max _{t}|\epsilon(t)|<\bar{e} \tag{2.2}
\end{equation*}
$$

A more detailed statement on the noise can be made by making $\bar{e}$ a function of time. This can be handled by parameter set estimation techniques as well, but this still does not address the unattractive property of the characterisation (2.2) that the noise can behave very wildly: at one sample instant $\epsilon(t)$ may be equal to $\bar{e}$, at the next it may be $-\bar{e}$ and it can keep alternating between these two extremes. It is very unlikely that the true noise and disturbance signals really show this behaviour. Neglecting for a moment the problem of quantifying this information, there remains the problem of incorporating this knowledge into the model error bounding algorithm.

This example has many variants. Think for example of algorithms that require bounds on the noise in frequency domain. These can not handle (2.2) as such. To determine the largest possible deviation in frequency domain following from (2.2) for a certain frequency $\omega$, one would have to consider the power in a sinusoidal signal $\epsilon(t)$ of frequency $\omega$ and amplitude $\bar{e}$. For another frequency $\omega$ the procedure has to be repeated. Considering only these maximal deviations in frequency domain is a very conservative representation of (2.2): it can be concluded from (2.2) that if for a certain frequency $\omega$ the deviation is equal to its maximal value, then the deviation for all other frequencies is necessarily equal to zero!

Another problem with the noise bound (2.2) is that it assumes that the noise and disturbances enter the process at the output. It is likely that some of the noise enters the process at the inputs, so that it is filtered by the dynamics that are to be modelled. Much more difficult is noise that enters the process through unmodelled inputs. The noise can then be assumed to be filtered by some of the dynamics that are to be modelled, but one may be unable to say a priori by which part of the dynamics.

One may also have information concerning the process transfer itself that can not be used by an error bounding algorithm. It is conceivable that one knows for a practical process the range of the frequency for which a resonance peak occurs. If the uncertainty structure that is used is an additive error with an $\mathcal{H}_{\infty}$-norm bound on the error,

$$
G_{t r}(\zeta)=G(\zeta)+\Delta(\zeta), \quad\|\Delta(\zeta)\|_{H_{\infty}}<d_{\Delta}
$$

the reader may verify that this information can only be represented in an approximate way. Figure 2.2 shows this for an example. In this Bode amplitude plot, the continuous lines correspond to the transfers having the largest and the smallest value of the resonance frequencies. The dashed line shows the frequency characteristic of the chosen nominal model. This has been chosen to minimise the conservatism used by an additive error description, while being consistent itself with the prior knowledge. The shaded area can be expressed as a frequency dependent bound on the amplitude of the additive error $\Delta$. It is the smallest such area that contains all transfers consistent with prior knowledge. Nevertheless this area contains much more transfers than only those consistent


Figure 2.2: Approximate description of varying resonance frequency
with the prior knowledge that only the resonance frequency may vary.
As another example of a case in which the true undermodelling/uncertainty does not fit the measure that is used to quantify the uncertainty, think of the $\mathcal{H}_{\infty}$-norm bound (2.1) for a SISO uncertainty block $\Delta$. If the true uncertainty in $\Delta(j \omega)$ can be represented by the darkest shaded circle in figure 2.3 for a certain value of $\omega$, the amplitude-based $\mathcal{H}_{\infty}$-norm bound can represent this at best by the outer-bounding light shaded circle. One may wonder to what


Figure 2.3: Conservatism in an amplitude bound
extent the situation shown in figure 2.3 is realistic. This may be illustrated by means of results obtained for the water vessel process shown schematically in figure 2.4. This example has been taken from (Van den Boom, 1993). The process consists of four vessels $V_{1}, \ldots, V_{4}$ into which water can flow at the top and from which water flows through a restriction at the bottom. The outflow of a vessel is roughly proportional to the water level in the vessel. The response from in-flow to water level is therefore roughly a first order response. The inputs to the total process are the water flows into the two top-most vessels. The outputs to be controlled are the water levels in the two bottom vessels. In (Van den Boom, 1993) the model uncertainty was characterised by means of the maximal $\mathcal{H}_{\infty}$-norm of an uncertainty block $\Delta$ related to the additive error on a coprime factorisation of the nominal model of the process. It is beyond the scope of this example to explain the uncertainty bounding procedure in detail. The interested reader is referred to (Van den Boom, 1993). As a result of the worst-case approach followed in this technique, the largest possible absolute value for $\Delta(j \omega)$ was calculated that was still consistent with the experimental data and the prior knowledge. The prior knowledge consisted mainly of a bound on the noise in the measured input and output signals in frequency domain and a stabilisation assumption. It would be interesting to compare this worst-case amplitude of $\Delta(j \omega)$ with the best-case amplitude, i.e. the smallest possible absolute value of $\Delta(j \omega)$ that is consistent with the data and the prior knowledge.

In figure 2.5 both the maximal (worst-case) and minimal (best-case) amplitude of $\Delta(j \omega)$ have been plotted for a number of frequencies. It is interesting to see, that the best-case errors can have values significantly larger than zero. For all frequencies where this is the case, an effect similar to that of figure 2.3 occurs: the worst-case amplitude corresponds for each frequency $\omega^{\prime}$ to a disk in the complex plane that is known to contain $\Delta\left(j \omega^{\prime}\right)$. The best-case amplitude corresponds to a disk in complex plane that is known not to contain $\Delta\left(j \omega^{\prime}\right)$. Nevertheless this "best-case disk" is completely contained by the "worst-case disk."

All examples discussed so far in this section are cases in which one has extra prior knowledge that can not be used by the algorithm. One may distinguish here the case that one has well-defined quantitative information that can not be used and the case that one has only vague, qualitative knowledge that can not be represented accurately, let alone be used by a formal algorithm. Ignoring the latter kind of knowledge is perhaps not so much a shortcoming of the model error bounding algorithm: one can not expect to derive well-defined, accurate noise bounds from knowledge that can not be stated accurately and unambiguously.

Another cause of conservatism lies in the concessions that have to be done in practical algorithms to the computability of the bound. A tight bound is often over-bounded by a bound that is less accurate, but easier to compute.


Figure 2.4: Water vessel process taken from (Van den Boom, 1993)


Figure 2.5: Best- and worst-case errors for the water vessel process

The final source of conservatism that is mentioned here is the data preprocessing that precedes the identification and error bounding steps. Detrending, linearising etc. have as objective to remove unwanted effects from the data. Unfortunately this normally also distorts or removes information that one would actually like to keep. Nevertheless, deciding not to preprocess the data because of this is probably a cure that is much worse than the disease.

### 2.3 Different types of model error bounds

In this section some choices are discussed that are made by the designer of an algorithm for model uncertainty bounding, even though he/she may not be aware of this.

### 2.3.1 Hard bounds or soft bounds

The issue whether model uncertainty bounds should be "hard", i.e. deterministic, guaranteed to be correct, or "soft", i.e. probabilistic, correct up to a certain confidence level, has received much attention. Despite the discussion whether soft bounds can be reconciled with the philosophy of robust control design (more specifically $\mathcal{H}_{\infty}$ design and $\mu$-synthesis) or only hard bounds are compatible with the worst-case approach taken in these techniques, the difference between hard and soft bounds is not as sharp as it may seem. If one takes the size of a $99.5 \%$ confidence interval as a hard bound on the noise level in the experimental data and uses this subsequently in a "hard bounding" technique, are the resulting noise bounds hard or soft? If one decides that these bounds must be soft, then the issue can be closed, because in practical measurements there is always some random noise and the only bounds we can ever get are soft bounds. So, in order to discuss this problem a little further, let us agree that the bounds have to be considered "hard." How is the $0.5 \%$ possibility that the noise bounds are too small to be put in agreement with the earlier claims that hard guarantees were required?

Still, it would go too far to claim that the whole problem is a non-issue. One should be aware of the kind of assumptions made to properly interpret the results of an algorithm. It is probably not possible to pronounce one approach as correct and the other is incorrect.

In (De Vries, 1994) it is argued that one should use a probabilistic measure on the uncertainty due to noise, because this leaves the possibility open that the influence of noise "averages out." This addresses the conservatism that was discussed in section 2.2 .3 based upon equation (2.2) on page 23. It is further stated that undermodelling should be bounded in a deterministic way, as undermodelling does not average out over time and can have a worst-case nature indeed.

In (Goodwin and Salgado, 1989; Goodwin et al., 1992) the stochastics is taken one step further and both the noise and the undermodelling are taken to be samples from a random process. In the first reference the prior knowledge consists of the cross-covariance function in frequency domain of the additive error. In the second it consists of an exponentially decaying upper bound on the variance of the impulse response parameters of the additive error. "Additive error" is taken here with respect to some unknown optimal model. This prior information is then mapped in frequency domain into an a posteriori estimate of the expectation of the square of the additive error with respect to the nominal model.

Another approach to ease the worst-case nature of many error bounds could be to attach a probability distribution to the bound. To our knowledge, this would be a new approach. It is therefore interesting to investigate this idea a little further. Restricting attention for a moment only to the influence of noise, a probability or likelihood can be specified for the differences between the measured signals ( $u_{m}, y_{m}$ ) and all candidate "true", undisturbed signals $\left(u_{t r}, y_{t r}\right)$ where $\left(u_{t r}, y_{t r}\right)$ are taken from some set $U_{t r} \times Y_{t r}$. A mapping $\mathcal{M}$ is then to be formulated, mapping a candidate undisturbed signal $\left(u_{t r}, y_{t r}\right)$ to a measure $d$ on the model uncertainty:

$$
\mathcal{M}: U_{t r} \times Y_{t r} \rightarrow \mathbb{R}_{+}, \quad\left(u_{t r}, y_{t r}\right) \mapsto d
$$

Any hard error bounding technique can be used for $\mathcal{M}$. Note that in this simplified case no prior knowledge is involved in $\mathcal{M}$. Define the "inverse mapping"

$$
\overleftarrow{\mathcal{M}}: \mathbb{R}_{+} \rightarrow 2^{U_{t r} \times Y_{t r}}, \quad d \mapsto\left\{\left(u_{t r}, y_{t r}\right) \in U_{t r} \times Y_{t r} \mid \mathcal{M}\left(u_{t r}, y_{t r}\right) \leq d\right\}
$$

Let $f_{U Y}\left(u_{t r}, y_{t r}\right)$ be the probability density that was attached to the candidate undisturbed signals $\left(u_{t r}, y_{t r}\right)$. The probability distribution $F_{d}(d)$ of the bound $d$ is then given by

$$
\begin{equation*}
F_{d}: \mathbb{R}_{+} \rightarrow[0,1], \quad d \mapsto \int_{\overleftarrow{M}(d)} f_{U Y}(u, y) d(u, y) \tag{2.3}
\end{equation*}
$$

Although the idea is simple, application of it seems almost impossible. Let us assume for simplicity that an explicit expression is available for $\overline{\mathcal{M}}(d)$, a very optimistic assumption indeed. It would still be unlikely that an analytic solution exists for the integral in (2.3). Reverting to numerical integration provides no solution, as, even in case of a SISO system, the integral is of dimension $2 N$ where $N$ is the number of data points in the experimental data.

Remark 2.2 In the remainder of this section attention will seem to be restricted to hard bounds. This is by no means meant to imply that hard bounds are to be preferred over soft bound, it just allows for simpler expression of the statements to be made. Nevertheless all statements can also be reinterpreted in a soft-bounding sense and should apply as such as well.

### 2.3.2 Process in the model set or not

An issue attracting little discussion but dividing the model uncertainty bounding approaches clearly in two different camps is the question whether the true process can be considered to be in the model set or not. In this section we will assume that the model set consists of parametric models.

Considering model sets consisting of restricted complexity models and assuming that the true process can be described exactly by one of those models provides a means to extrapolate information obtained during finite experiments to statements holding true for all time instances. Using this approach it is possible to say something about what experimental results would have looked like if the experiment had spanned a longer period. The assumption that the process is in the model set provides explicit conditions on the way in which time-extrapolation can occur. In frequency domain both interpolation between the sampled frequency-points and extrapolation to higher frequencies occurs. This is important to realise, because one needs to have an alternative for this if one refuses to assume that the process is in the model set: even though one does not assume that the process is in the model set, there must be some way of precluding the possibility that the process would start responding fiercely to the excitations after the experiment interval. For frequency domain data, there must be some way of ensuring that the frequency response of the system is in some sense smooth between the frequency samples. This can be achieved in several ways.

A common way is to assume an exponentially decaying bound on the impulse response parameters of the true system. Similar to this is the assumption for discrete time systems that the poles of the true system are within a circle of radius $\rho$, where $\rho$ is known and less than one. Another method, used in (De Vries, 1994; Hakvoort, 1994) and chapter 4 and 5 of this thesis, is to assume an exponentially decaying bound on the generalised impulse response parameters of the system. For more information on the latter approach the reader is referred to the above references and to chapter 4 of this thesis.

Remark 2.3 The smoothness assumption above can be thought of as the assumption that the true process is in some set. This set is a subset of the set of linear time-invariant processes. This subset will be encountered again in chapter 3.

In parameter set estimation techniques, one basically assumes that the process is in the model set, see (Walter and Piet-Lahanier, 1990) and the references therein. To some extent this has been generalised in (Milanese and Elia, 1993; Livstone and Dahleh, 1995; Vicino and Zappa, 1996). These references allow for an unstructured additive model uncertainty. However, it is assumed that this uncertainty is norm-bounded a-priori. The possible error this extra uncertainty induces in the estimated model parameters is then bounded
by the proposed algorithms. The problem of obtaining this prior knowledge, which is exactly what the problem of model uncertainty bounding is all about, is not addressed by these references.

If one assumes that the process is in the model set, all of the undermodelling effects listed under 1) to 5) in section 2.2 .1 , page 20 onwards, are ignored completely. If one recognises that the process is more complex than any model, the effects listed under 4) and 5) can be accounted for. In the context of robust control design, this is definitely to be preferred.

Remark 2.4 The effect of 2) "working points", page 21, can be accounted for by using separate data sets for a range of operating points and operating conditions. This range should cover the range of conditions that can be expected after the implementation of the controller that is still to be designed. One may then distinguish local uncertainty bounds, valid in one operating point, and global uncertainty bounds, valid for all operating points. The global bounds are simply obtained by taking bounds containing all local bounds. This idea is applied in the algorithm of chapter 4.

This approach is valid only in cases where it can be relied upon that all of a particular data set was obtained in a single operating point under roughly constant operating conditions. If this is not the case, one is, possibly unknowingly, trying to reduce uncertainty of a model for one operating point by using data for another operating point.

### 2.3.3 Best case or worst case

At a first glance it seems clear that robust control design methods need a specification of an uncertainty description that, provided the the prior knowledge and assumptions are valid, guarantees to at least model the true process. In other words, uncertainty bounding should find the smallest possible bound that we can trust to be large enough to contain a description for the real plant. This leads to worst-case bounds: given a set that is known to contain the true process, the bound that is large enough to contain all processes in this set should be used.

Nevertheless the opposite approach is encountered in the literature as well. In (Zhou and Kimura, 1993; Zhou and Kimura, 1994) the smallest bound that can not be falsified by data and prior knowledge is searched: the data and prior knowledge together define implicitly a set that is known to contain the true process. The bound on the uncertainty derived by Zhou and Kimura is chosen such that at least one process in that set satisfies the bound. The stronger statement that all processes in the set satisfy the bound can not be guaranteed.

In connection with robust control design this seems a suspicious approach, even though the references consider noise-free data only. There is also another
strange effect: the more data is obtained from the process, the greater our "tools" to falsify an error bound. The error bound is a non-decreasing function of the amount of data that we have available!

Remark 2.5 The effect that having more data increases the error bound was encountered earlier: if data is obtained in another operating point, the model error bound can not be reduced, but is possibly increased. However, increasing the bound has the benefit of extending the faith we can have in the error bound to an extra operating point of the process.

Given this explanation, there might be a similar explanation for the case at hand. This explanation would be that a longer data set increases the validity of the error bound to a longer time span. This would imply that the error bound is only valid for the duration of the experiment. Clearly, this is unacceptable.

### 2.3.4 Bound for one process or any process

The bounds on the worst-case model error derived in (Gu, 1994; Gu and Khargonekar, 1991; Helmicki et al., 1989; Helmicki et al., 1990a; Helmicki et al., 1990b; Helmicki et al., 1991) are the worst possible bounds that one may observe if the true system is any system in a restricted set. (This set is the kind of set discussed in section 2.3 .2 if one does not want to assume that the true process is in the model set.) This means in particular, that the bounds are valid not only for the process for which we want to have a model but also for all other processes in that set. This in turn implies, that these particular bounds can not depend on the experimental data, because these are necessarily taken from only one member in the set. This differs from most other bounds, in which the data potentially does contribute to a reduction of the model uncertainty.

Having a bound that applies for all possible processes is useful if one wants to compare the quality of different algorithms. If algorithm A gives a tighter bound than algorithm B for a certain process but algorithm B outperforms algorithm A on another process, one can not say that either is better than the other. If the bounds produced by the algorithms are not tied to particular processes, the algorithms can be compared. If one is "only" interested in a bound on the uncertainty for one particular process, namely the process that is to be controlled, one need not take such a worst-case approach. Nevertheless it may be attractive for reasons of computability to do this anyway.

Remark 2.6 Only the bounds on the model uncertainty do not depend on the data in the above references. The identified model does depend on the data of course. In this sense the data reduces the model uncertainty quite significantly!

### 2.4 Guality aspects of uncertainty bounds

In this section aspects concerning the practical applicability, or quality, of an error bounding technique will be discussed. These should be interpreted as "desirable properties." It is not the case that a technique having all of the properties mentioned here is necessarily a good technique and a technique lacking some of them is useless in practice.

The probably hardest requirement for practical application is that the algorithm provides a computable bound. Whether a bound is to be considered computable or not is to some extent up to personal taste. Bounds for which simply no algorithm to compute them are known, are obviously not computable. But, while one person may quite happily spend a week's worth of computing power on the calculation of a bound, someone else may find this already unacceptable and consider the bound uncomputable. Apart from the subjective element in this decision it also depends strongly on the state of the art in (affordable) computing power, which has been developing rapidly during the last decades.

In the context of uncertainty bounding for robust control design, it is imperative that the statement about the model uncertainty can be used by techniques for robust control design. This is related to, but not identical to aiming for bounds that are in line with control objectives. The link between identification and control has been studied for example in (Gevers, 1991; Gevers, 1993; Schrama and Van den Hof, 1992; Schrama, 1992).

It should definitely be considered a practical advantage of an error bound if it can incorporate the kind of prior knowledge that one has in a certain situation. The match between what kind of knowledge is practically available and what can be used by error bounding techniques would benefit from a wide discussion between practicists and theorists.

Also of much practical value is a bound that is interpretable in physical terms. An interpretable bound can be confronted with one's expectation for the model uncertainty. If the obtained bound is in line with this expectation, the faith in the bound is increased, even in the presence of the simplifying assumptions that were made in the derivation of the bound. If, on the other hand, the bound deviates significantly from what was expected, this is a sign that things need to be carefully checked. It may be that the error bound is conservative, but it could also be that the nominal model, the input design, the experimental conditions etc. are the cause of the unexpected results. In summary, an interpretable bound can be validated, or invalidated, based on arguments that are not necessarily restricted to the theoretical framework into which identification and error bounding have been cast.

Contrary to what the name suggests, prior knowledge consists mostly of assumptions that are made about the process. These assumptions are hopefully very sensible indeed, but they remain assumptions, not real knowledge. If the
error bounds turn out to be determined mainly by the prior "knowledge," one should realise that these assumptions have far-reaching consequences. If, on the other hand, certain aspects of prior knowledge did not contribute to a reduction of the uncertainty bounds at all, it may be worthwhile to investigate whether that particular specification was not too cautious. These considerations are only possible if it is known what actually determined the size of the uncertainty bounds. The ability of an uncertainty bounding algorithm to provide this information is the final desirable property we like to mention.

### 2.5 Summary

There seems to be a different interpretation of the concept "model uncertainty" in the areas of control design and model error bounding. The three main categories of contributions to a model error bound are undermodelling, finite data and noise and conservatism. For a robust control design it is desirable that as many of these aspects of undermodelling as possible are bounded in a reliable way by the eventual error bounds.

Model error bounds can be decreased by reducing conservatism. Important causes of conservatism are the inability to use all of the prior knowledge that is available and the inability to express all of the a posteriori knowledge about the model uncertainty in the uncertainty bound.

## 3

# A formal foundation for model uncertainty bounding 

3.1 Introduction<br>3.3 Discussion<br>3.2 Formalism<br>3.4 Summary

### 3.1 Introduction

In chapter 2 model uncertainty was considered in an informal way. In this chapter a general formal framework is presented for process identification and model uncertainty bounding. The aim of this formalism is to describe at an abstract level the concepts involved in process identification and uncertainty bounding. This provides a framework in which one can think about model uncertainty bounding without being tied to a particular algorithm.

The formalism is intended to be complete in the sense that every possibly relevant aspect of the process behaviour is covered by elements of the formalism: if the true system can not be fully represented by models in the chosen model class, which is always the case in practice, the formalism should provide enough alternatives to represent the mismatching parts by. As was already mentioned, the framework should not be restricted to a particular class of existing or prospective algorithms. Due to this generality, most terms in this formalism are rather abstract: for different algorithms the abstract notions may be given different forms. For clarity an example is developed along with the formalism.

### 3.2 Formalism

To represent systems and models, the behavioural representations described in (Willems, 1986) will be used. In this representation, dynamical systems are considered sets of vector valued signals. More precisely, a dynamical system is considered to be a triple ( $T, W, B$ ), where $W$ is called the signal alphabet, $T$ the time set and $B \subset W^{\mathbb{T}}$ the behaviour of the system. The behaviour $B$ is
the set of all signals $w(t)$, defined for $t$ in the time set $\mathbb{T}$, that the system may exhibit.

Example: Consider the example of a two phase asynchronous motor. For a further discussion of this kind of device, see section 6.5 on page 229 . Of the many aspects of this device, we may choose to limit attention to the voltages $U_{1}$ and $U_{2}$ over the stator windings, the currents $I_{1}$ and $I_{2}$ through these windings, the mechanical torque $T_{m}$ exercised by the motor at its axis and the angular velocity $\omega$ of the axis. If we abstract from the units in which these quantities are to be expressed, we can say that the signal alphabet is equal to $\mathbb{R}^{6}$ : for $t^{\prime} \in \mathbb{T}$ we have that $w\left(t^{\prime}\right):=\left(U_{1}\left(t^{\prime}\right), U_{2}\left(t^{\prime}\right), I_{1}\left(t^{\prime}\right), I_{2}\left(t^{\prime}\right), T_{m}\left(t^{\prime}\right), \omega\left(t^{\prime}\right)\right) \in R^{6}$.

If the system is considered in discrete time domain, $\mathbb{T}=\mathbb{Z}$ or $\mathbb{T}=\mathbb{Z}_{+}$. If it is considered in continuous time, then $\mathbb{T}=\mathbb{R}$ or $\mathbb{T}=\mathbb{R}_{+}$. Notwithstanding the name "time set" for $\mathbb{T}$ there is nothing prohibiting that the engine is considered in frequency domain. This gives $\mathbb{T}=\mathbb{C}$ and $W=\mathbb{C}^{6}$. Variants of this where $\mathbb{T}$ is a subset of $\mathbb{C}$, such as the imaginary axis, the unit circle or sampled subsets of these, can easily be found.

This is already a gross abstraction of the physical machine. The temperature of the windings is not considered, nor the price of the engine, its manufacturer or the colour of its case, to name only a few things.

To consider the behaviour of a system $B$ on a subset $\mathbb{T}^{\prime}$ of $\mathbb{T}$, the notation $\left.B\right|_{T^{+}}$will be used. This is formally defined as

$$
\left.B\right|_{\mathbb{T}^{\prime}}:=\left\{w^{\prime} \in \mathbb{W}^{\mathbb{T}^{\prime}} \mid \exists w \in B, \forall t^{\prime} \in \mathbb{T}^{\prime} \quad w^{\prime}\left(t^{\prime}\right)=w\left(t^{\prime}\right)\right\}
$$

It is assumed that for a certain behaviour $B$ a partitioning of the signals $w(t) \in B$ into inputs and outputs is both possible and known. This will be denoted $w(t)=(u(t), y(t))$ where $u(t)$ consists of the input signals and $y(t)$ of the output signals. The same partitioning of $w$ into $u$ and $y$ will be used for all behaviours in the formalism. This partitioning is used at one point in the formalism, but the formalism does not really rely on it.

There are two reasons to suffer the loss of generality induced by distinguishing between inputs and outputs. Firstly, it makes the formalism easier accessible for those used to thinking in terms of inputs and outputs, which is, despite the work of Willems and his followers, in my opinion the large majority of the identification and control community. For those trained in not thinking in terms of inputs and outputs the generalisation to a more symmetric formulation should be straightforward. Secondly, in all practical cases where the signals can be partitioned into actuator signals and sensor signals, which constitutes, again at least in my view, by far the majority of cases, the partitioning into inputs and outputs is trivial. Therefore the loss of generality is not considered very significant.

Another departure of convention with respect to the behavioural framework is that behaviours will not be denoted $B$ or $\mathcal{B}$ in this formalism. If a symbol describes a process, model or any other dynamical system then it is a behaviour, this is not further stressed by denoting it $B$. The variation in notation that is made possible by dropping this convention is used to denote different kinds of systems or models by different symbols. In the examples, also other representations of models will be used.

An overview of the formalism is now given in the symbolism (3.1) below. At this point it will not make much sense yet, but throughout the remainder of this section, the various symbols in this framework will be introduced. It is expected that the reader will often be referring back to this overview. Therefore it is repeated at the top of every left page in this chapter; this is meant as an aid to the reader and not as a vain attempt to make the formalism seem more important than it is.

$$
\begin{align*}
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{\text {id }} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{\text {id }} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
D \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathrm{e}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

Before explaining all the symbols in this formalism in detail, it is worthwhile to mention a few conventions here. A behaviour of a system, process, model, etc. is denoted by a normal Roman capital letter, such as $S, M$, etc. Capital Roman script letters, such as $\mathcal{S}, \mathcal{P}$ and $\mathfrak{Q}$, are used for sets of behaviours. Behaviours being sets of trajectories, these script capital letters denote sets of sets of signal trajectories.

The items marked by a ' $\square$ ' involve concepts that the formalism can not generate but that are required in later steps of the formalism. An example of this is the symbol $W$, that will turn out to represent the experimental data used in identification. In the formalism, experimental data can not be derived from other, known quantities. Nevertheless several steps in the formalism will require that experimental data is available.

In the remainder of this section the various symbols will be explained. A discussion of a symbol is marked by a box containing that symbol and a ' $D$ ' sign. This makes it possible to see at a glance what symbol is being explained in a certain part of the text or where to find an explanation for a certain symbol.

$$
\begin{align*}
& \rightarrow\left\{\begin{array}{l}
\mathcal{M}_{i d} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
D \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

$S \triangleright$ The particular system under study is denoted $S . S$ is the behaviour of that "true" system, so it is the set of all signals $w(t)$ that the true system can exhibit. It is implicitly assumed, that the signal alphabet $\mathbb{W}$ and the time set $\mathbb{T}$ are known. Unless otherwise noted, this signal alphabet and this time set will also be used for all other behaviours in this formalism.
In practical situations we may have access to (part of) the scalar valued signals in $w(t)$ for some $w(t) \in S$, where $t \in \mathbb{T}^{\prime} \subset \mathbb{T}$ and $\mathbb{T}^{\prime}$ is some bounded subset of $\mathbb{T}$. Accessing these signals may contaminate the "true signals," for instance by introducing measurement noise. Also other external influences may be present in a (partial) observation of $w(t)$. These contaminations are assumed to be present in the signals $w(t) \in S$. With this interpretation $S$ represents somewhat more than only the true system including all its imperfections; it also includes the imperfections of the measurement equipment.
The set $S$ is not accessible for measurements. At best we can try to verify whether some $w_{1}(t)$ is a member of $S$. As soon as we have carried out an experiment for $t \in \mathbb{T}^{\prime}$, it is impossible to do another experiment for $t \in \mathbb{T}^{t}$. Apart from this, we can not do experiments for all $t \in \mathbb{T}$ if $T$ is unbounded, which will generally be the case. This implies that even elements of $S$ are in practice not fully accessible for measurements.
In advance of the introduction of the concept of stabilisability, which will be formalised on page 41 , it is already stated here that the system $S$ is stabilisable.
S] $\triangleright$ is the set of all processes of the same kind as $S$. This is the place where one can put any prior knowledge about the system $S$ that is available by virtue of the fact that it applies for any system of that kind.

Example (cont.): If $S$ is the behaviour of the motor, then $S$ may be (a subset of) the set of dissipative systems. Indeed, the law of preservation of energy
states, that all energy that has somehow entered the motor should have either been stored in the motor or have left the system. In practice one meets the problem here of accounting for all losses that occur in the system. A first attempt at this might be

$$
\begin{array}{r}
\int_{-\infty}^{t} U_{1}(t) I_{1}(t)+U_{2}(t) I_{2}(t) d t=\sum_{i} \frac{1}{2} L_{i} I_{i}^{2}(t)+\sum_{j} \frac{1}{2} C_{j} V_{j}^{2}(t)+ \\
\sum_{i_{1}} \sum_{i_{2} \neq i_{1}} \frac{1}{2} M_{i_{1}, i_{2}} I_{i_{1}}(t) I_{i_{2}}(t)+\sum_{k} \int_{-\infty}^{t} R_{k} I_{k}^{2}(t) d t+ \\
\frac{1}{2} J \omega^{2}(t)+\int_{-\infty}^{t} T_{m}(t) \omega(t) d t+\text { other effects } \tag{3.2}
\end{array}
$$

Skipping the sign conventions that are implicitly assumed here, the meaning of the new symbols is as follows. $U_{1} I_{1}$ is the power that flows to the motor through the first stator winding, $U_{2} I_{2}$ is the same for the other stator winding. $L_{i}$ for $i$ in some countable set is the set of all inductances that can be distinguished in the motor. $I_{i}$ is the electrical current flowing through inductance $L_{i} . M_{i_{1}, i_{2}}$ is the cross-conductance between two "ports" $i_{1}$ and $i_{2}$. The currents flowing into these ports are $I_{i_{1}}$ and $I_{i_{2}}$ respectively. Note that every pair of ports occurs twice in the summation, once as the pair $\left(i_{1}, i_{2}\right)$ and once as the pair $\left(i_{2}, i_{1}\right)$.
$C_{j}$ for $j$ in another countable set is the set of all lumped capacitances in the motor. $V_{j}$ is the voltage over capacitance $C_{j} . R_{k}$ is a set of resistances, through which currents $I_{k}$ flow. They account for the electrical losses in the motor.
$T_{m} \omega$ is the mechanical power exercised by the motor on its axis: $T_{m}$ is the torque that the motor exercises on an external load and $\omega$ is the angular velocity of the axis. $\frac{1}{2} J \omega^{2}(t)$ is the kinetic energy in the turning parts of the motor. $J$ is the inertia of those parts of the motor that participate in the rotating motion.

With "other effects" all other losses and buffers of energy are meant. This includes the mechanical losses due to friction, electro-magnetic radiation, edy current losses, etc. If the motor jumps from its position during operation, it will have a kinetic energy and, if it happens to jump up, also a potential energy, etc.

Most effects are hard to quantify exactly, but most of them can be bounded. A rough but simple conclusion to be drawn from (3.2) is

$$
\int_{-\infty}^{t} U_{1}(t) I_{1}(t)+U_{2}(t) I_{2}(t) d t \geq \int_{-\infty}^{t} T_{m}(t) \omega(t) d t
$$

If $\mathbb{T}=\mathbb{R}$ and $\mathbb{T}$ is indeed a time set in the literal meaning of the word, then this inequality can already be used to exclude certain behaviours from

$$
\begin{align*}
& \left.\rightarrow \underset{\text { D requirements }}{\{ } \begin{array}{c}
\mathcal{M}_{\mathrm{id}} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{\mathrm{id}} "+" \Delta \Delta \\
d(\Delta) \leq d_{\Delta} \\
D
\end{array}\right\} \rightarrow C_{0} \in \mathrm{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

S. For behaviours consisting of one-sided sequences, discrete time sequences or frequency domain sequences, additional work is involved before the law of preservation of energy can be used to determine membership of $S$ for a certain $S^{t} \subset W^{T}$. This is not further pursued here, because the purpose of this example was only to give an idea of the kind of knowledge that can be represented by the set S .
$\delta$ (cont.) $\triangleright$ The set $\mathcal{S}$ is certainly not a well-defined set! How broad or how narrow the set $\mathcal{S}$ is depends highly on the knowledge one has of the kind of device of which $S$ is a particular realisation. If there is no knowledge that can be used to decide that a certain behaviour can not possibly represent a device similar to the true system $S$, then we have the extreme case that

$$
\mathcal{S}=2^{W^{T}}
$$

$\mathcal{S}$ is the set of all subsets of $W^{T T}$.
$W$ D Some measurements have been taken of the system $S$ on a time set $\mathbb{T}^{\prime} \subset \mathbb{T}$. These measurements are represented by a set $W \subset W^{\mathbb{T}^{\prime}}$. Thus

$$
\left.W \subset S\right|_{\mathbb{T}^{\prime}}
$$

Example (cont.): Assume that the asynchronous motor is considered in discrete time, so that $\mathbb{T}=\mathbb{Z}$. Assume further that measurements of all elements of $w$ were taken from $t^{\prime}=T_{s 1}$ to $t^{\prime}=T_{e 1} \geq T_{s 1}$ and from $t^{\prime}=T_{s 2}$ to $t^{\prime}=T_{e 2} \geq T_{s 2}$. Assuming for simplicity that $T_{e 1}<T_{s 2}$, it holds

$$
T^{t}=\mathbb{Z} \cap\left(\left[T_{s 1}, T_{e 1}\right] \cup\left[T_{s 2}, T_{e 2}\right]\right)
$$

and $W$ consists of a single trajectory $w=\left(U_{1}, U_{2}, I_{1}, I_{2}, T_{m}, \omega\right)$, specifying a measured value for $U_{1}\left(t^{\prime}\right), U_{2}\left(t^{\prime}\right), I_{1}\left(t^{\prime}\right), I_{2}\left(t^{\prime}\right), T_{m}\left(t^{\prime}\right)$ and $\omega\left(t^{\prime}\right)$ for every $t^{\prime} \in \mathbb{T}^{\prime}$.

From this example it may be clear that $W$ contains in general only one trajectory, even if multiple data sets were obtained from the system $S$. If $W$ contains multiple elements, that implies that multiple measurements of the system have been taken simultaneously.

## Modelling and uncertainty bounding aspects

$C_{\text {stab }} \triangleright$ It is assumed that a stabilising controller for the system is both known and connected to the system during the experiments in which $W$ was obtained. For stable systems this may reduce to a situation in which no controller is present.
The interconnection of the system $S$ and a controller $C_{\text {sta }}$ will not be formalised. The scheme of figure 3.1 specifies it informally. The input $u$ of the process $S$ is partitioned into a "free" part $u_{1}$ and a part $u_{2}$ connected to the controller output. The output is partitioned similarly into a part used by the controller and a part not used by the controller. An extra set of free signals $r$ is introduced. In general these signals represent the reference trajectories one wants the system $S$ to follow in some sense. Both $u_{1}$ and $r$ may be empty. The signal $\xi_{23}$ represents the disturbance and measurement noise effects that are part of $S$. The notation will become clear later in this chapter. Although these signals are considered in this formalism to be part of $S$, they may be external signals to the system from a physical point of view. In case both $u_{1}$ and $r$ are empty, the closed loop system has turned into an autonomous system.
The controller $C_{s t a b}$ is said to stabilise $S$ if

$$
\left(\left\|u_{1}\right\|<\infty \wedge\|r\|<\infty \wedge\left\|\xi_{23}\right\|<\infty\right) \Longrightarrow\left(\left\|u_{2}\right\|<\infty \wedge\|y\|<\infty\right)
$$

This is generally referred to as bounded-input, bounded-output (BIBO) stability. $S$ is said to be stabilisable if a controller $C_{s t a b}$ exists that sta-


Figure 3.1: Interconnection of $S$ and $C_{\text {Stab }}$

$$
\begin{align*}
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{s e t} \\
\mathcal{M}_{i d} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
\square \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

bilises $S$. For the sake of generality it is not specified what norm is actually meant with $\|$.$\| .$

It can now be explained why the noise and disturbances $\xi_{23}$ had to be drawn explicitly in figure 3.1. BIBO stability requires that the output remains bounded if the inputs remain bounded. However, it may very well be the case that the outputs become unbounded if the external inputs to the system that were absorbed in $S$, i.e. the noise and disturbance influences represented by $\xi_{23}$, become unbounded. If this happens, we do not want this to imply that the system is unstable.
Note that it is not assumed that $S$ is a normed set. $S$ is intended to represent the "true" physical system and it is therefore not considered desirable to restrict the signal trajectories in $S$ to those trajectories that have a finite value for $\|w\|,\|u\|$ and/or $\|y\|$.

In the remainder of this chapter it will be assumed that the reference signal $r$ is bounded:

$$
\begin{equation*}
\|r\|<\infty \tag{3.4}
\end{equation*}
$$

$S$ (cont.) $\triangleright$ Another property of the systems in $\mathcal{S}$ has been introduced implicitly in the discussion of $C_{s t a b}$ : all systems in $\mathcal{S}$ are stabilisable. This corresponds to the assumption that the true process is stabilisable. This is reasonable to require. If the true system is not stabilisable, not much benefit can be derived from applying identification, uncertainty bounding and robust control to it anyway.
$Q \triangleright$ The set $Q$ consists of a class of systems that is present in every algorithm for model uncertainty bounding but for which there is nevertheless not a general name. Usually, it is the class of systems to which the true system is assumed to belong, such as the class of linear, time-invariant systems. In
any case, it is a class of systems in which the outputs $y$ can be completely explained from the inputs $u$, possibly up to transient effects if $T$ is a onesided time set. Because systems are represented in this formalism by their behaviour, $Q$ is a set of behaviours.
Contrary to the system $S$, the signals of the systems in $Q$ do not contain measurement noise or external disturbances. (A refinement could be made such that $Q$ contains also some information considering the noise acting on the system, such as a filtering indicating the colouring that is present in the noise. For convenience this is not carried out here.) If $Q$ contains linear systems, non-linear side-effects are neither part of the systems in $Q$.
In section 2.3 .2 it was pointed out that one needs to assume more about the true process than merely that it is linear, time-invariant if one wants to stand a chance of, say, extrapolating data from a finite, discrete time interval to an $\mathcal{H}_{\infty}$-bound on the additive model error. It is exactly this kind of knowledge that is represented by the set $Q$.
As the outputs can be explained from the inputs, there exists a mapping from the inputs, and possibly from initial conditions, to the outputs. This mapping is assumed to be a causal one, i.e. for a certain $t_{1} \in \mathbb{T}, y\left(t_{1}\right)$ does not depend on $u\left(t_{2}\right)$ for any $t_{2}>t_{1}$.
More requirements on $Q$ will follow.
Example (cont.): For the asynchronous machine it may be known that the response of the current through the windings to a pulse on the voltage over the windings falls off at a certain exponential rate. For $Q$ one might therefore choose the set of linear, time-invariant processes having at least this decay rate in their impulse responses.

Note that this is not meant to imply that the true system is a linear, timeinvariant process; see the following discussion on $\xi$.
$\xi \triangleright$ The signal $\xi$ accounts for everything in the true system $S$ that can not be accounted for by a system $Q \in Q$. Every $w \in S$ is decomposed into a component $w_{Q}$, that can be accounted for by a certain $Q \in Q$, and a component $\xi$ as follows

$$
\begin{equation*}
\forall w=(u, y) \in S \quad w=w_{Q}+\xi, \quad w_{Q} \in Q \in Q \tag{3.5}
\end{equation*}
$$

The signal $\xi$ acts, possibly, on both inputs and outputs. Neither $w_{Q}$ nor $Q$ are unique in (3.5), but some choices for $w_{Q}$ and/or $Q$ are more sensible for our purposes than others. Before considering this, figure 3.2 may already clarify to some extent the role of $\xi$. $\xi_{u 1}$ and $\xi_{y 1}$ are generally considered disturbances and $\xi_{u 2}$ and $\xi_{y 2}$ are normally regarded as measurement noise. In this formalism, both $u_{1}$ and $u_{3}$ may be considered the process input $u$. In case $u_{1}$ is used as the process input, the signals that a human operator

$$
\begin{align*}
& \left.\rightarrow \underset{\substack{ \\
\mathcal{M}_{\text {id }} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d} "+" \Delta \Delta \\
d(\Delta) \leq d_{\Delta} \\
\text { requirements }}}{ }\right\} \rightarrow C_{0} \in \mathrm{e}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

or a controller specifies for the plant is taken as "the" input. If $u_{3}$ is considered the process input, the disturbances in $\xi_{u 1}$ are eliminated from the signal $u$, but this goes at the expense of introducing measurement noise $\xi_{u 2}$. If the formalism is applied in an open loop situation, one may choose whether to use $u_{1}$ or $u_{3}$. However, in a closed loop situation one can only use $u_{1}$, because it was assumed in the discussions on stability that the process input $u$ was the signal that was generated by the controller, i.e. $u_{1}$. At the output we do not have this choice. The only process output that is in general available is the measured output $y_{3}$.

Attention is now turned to two problems: which $w_{\varrho} \in Q$ is to be used in (3.5) and which $Q \in Q$ ? There is a simple solution to the first problem:

$$
\begin{equation*}
\forall w \in S \quad w_{Q}=\arg \min _{\tilde{w} \in Q}\|w-\tilde{w}\| \tag{3.6}
\end{equation*}
$$

In words: for every trajectory in the true system $S$, find that trajectory in $Q$, called $w_{Q}$, that fits best to $w$. The misfit between a $w \in S$ and a $\tilde{w} \in Q$ is quantified here by $\|w-\tilde{w}\|$ for some norm. For simplicity the problems that the minimum may not exist or may not be unique are ignored here.


Figure 3.2: Possible interpretations for $u$ and $y$

A first attempt at a solution to the other problem could be to take $Q$ equal to

$$
\begin{equation*}
Q=\arg \min _{\hat{Q} \in Q} \max _{w \in S} \min _{\tilde{w} \in \mathscr{Q}}\|w-\tilde{w}\| . \tag{3.7}
\end{equation*}
$$

This takes a worst-case approach. It is not a feasible solution for two reasons: first of all, not for every $w \in S$ the expression $\|w\|$ may be well defined. This problem also applies to (3.6), but it was no attention given there as the solution that will be developed below for (3.7) will be applicable. Even if $\|w\|$ were well-defined for every $w \in S$, the expression $\|w-\tilde{w}\|$ can be expected to grow out of bounds if $\|w\|$ grows out of bounds. The choice of $Q$ is completely dominated by the fit that can be realised for large $w$. This may not be so much of a problem for linear $S$, but for non-linear $S$ it certainly is.

In order to resolve this problem, the assumptions that are made implicitly if a "real" system is approximated by a linear, time-invariant model need to be made more explicit. If a system is to be approximated by a linear, time-invariant model, this means that a linear, time-invariant description of the system has to be found, that gives a good account of the process behaviour locally.

Locally means first of all locally in time: there once was a time that the system to be approximated did not exist yet and there will be another time in which it does not exist any more or at least does not behave any more in the way we expect it to now. Therefore, the norm used in (3.6) and (3.7) should be such that there is little or no weighing of mismatches between $w$ and $\tilde{w}$ that occur far in the past and far in the future. What "far in the future" means depends on the period of time that the model is intended to be valid.

Weighing mismatches between $\tilde{w}\left(t^{\prime}\right)$ and $w\left(t^{\prime}\right)$ not at all in $\|w-\tilde{w}\|$ (equation (3.6)) for certain values of $t^{\prime}$ may turn the norm $\|w-\tilde{w}\|$ into a semi-norm. The notation $\|\cdot\|$ may then be somewhat misleading, but this is not considered a big problem.

Locally also means locally in amplitude range. The model should be a good representation of the system in the amplitude range where the system is expected to be used. As this formalism is about model uncertainty bounding for robust control design, the relevant amplitude ranges are those that will occur in the eventual closed loop situation.

Let the set $R \subset W^{T}$ be such that it contains only trajectories $w$ for which the amplitudes of inputs and outputs are within the range discussed above. Then instead of (3.7) a second attempt at picking a sensibly optimal $Q \in Q$

$$
\begin{align*}
& \left.\begin{array}{cc}
S \in \mathcal{S} \rightarrow & \triangleright W \\
& \triangleright C_{\text {stab }} \\
& \triangleright \forall u,\left(\|u\| \leq d_{u} \Longrightarrow\|\xi\| \leq d_{\xi}\right) \\
\triangleright \mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{s}}
\end{array}\right\} \rightarrow \mathcal{P \subset Q} \begin{array}{l} 
\\
\\
\end{array}  \tag{3.1}\\
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{s e t} \\
\mathcal{M}_{i d} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
\triangleright \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathrm{e}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ?
\end{align*}
$$

could be

$$
\begin{equation*}
Q=\arg \min _{\tilde{Q} \in \mathbb{Q}} \max _{w \in S \cap R} \min _{\tilde{w} \in \hat{Q}}\|w-\tilde{w}\| \tag{3.8}
\end{equation*}
$$

Further refinements can be made to the selection of $Q$, but the general idea is probably sufficiently clear by now. The optimal $Q$ will be referred to as the manifestation of $S$ in $Q$. In chapters 4 and $5 Q$ will be a subset of the set of linear, time-invariant processes. (See appendix C for a further discussion of the links between this formalism and the algorithm of chapters 4 and 5 .) Then $Q$ will be referred to as a linear, time-invariant manifestation of the process, or just the linear manifestation, which is shorter but less accurate.
Given the fact that the behaviours in 2 consist of trajectories in which the outputs $y$ can be explained from the input $u$ by relationships of a certain form, $\xi$ has to account for three effects:

1. The relation that is present between $u$ and $y$ may not be exactly of the form specified by elements in 0 .
2. The outputs $y$ will not be completely explicable by the limited number of signals that happened to be included in $u$. Other inputs to the system will have their influence on the outputs as well. This includes those inputs that are normally referred to as process disturbances.
3. The measurements of $y$ and possibly of $u$ may not represent the true value of these signals. The actual measurements can be contaminated by measurement noise.

The distinction between 2. (disturbances) and 3. (measurement noise) is not so relevant as far as identification is concerned, as in general no attempt is made during identification to distinguish between the two. However, for control design the difference is important. A controller is supposed to react to process disturbances but should be as insensitive to measurement noise as possible. Extra knowledge is then required to make a distinction between the two effects.

Example (cont.): Considering the inputs $U_{1}$ and $U_{2}$ of the asynchronous machine and its outputs $I_{1}$ and $I_{2}$, it is known that for large currents $I_{1}$ and $I_{2}$ a saturation effect occurs in the magnetic material inside the machine. This has its influence on the relation between $U_{1}$ and $U_{2}$ on the one hand and $I_{1}$ and $I_{2}$ on the other. If $Q$ is taken to be a set of linear models, this effect can not be described by any $Q \in \mathcal{Q}$. This is an example of the effects mentioned under 1 . that $\xi$ has to account for.

Suppose that the inputs $U_{1}$ and $U_{2}$ are specified in open loop by a feedforward controller or signal generator implemented in some computing equipment. It is conceivable, although maybe not very likely in practice, that through electro-magnetic coupling extra stray signals are picked up by the cables connecting the computing equipment to the motor. If the specified $U_{1}$ and $U_{2}$ are taken as the inputs to the process, the extra signals that are added to the inputs before they are really applied to the motor are to be considered exogenous inputs, or disturbances. This corresponds to an effect listed under 2. above.

The same effect may occur to the measurements of $I_{1}$ and $I_{2}$. Suppose these currents are converted to a voltage that is then lead to the computing equipment. If stray signals are picked up by the cables connecting the computing equipment to the current-to-voltage converter, these signals should be considered measurement noise and are an example of effect 3 .

The currents flowing through the motor windings have a limited range: if their amplitude gets too large, the windings will burn. This leads naturally to a range $R$ for the currents $I_{1}$ and $I_{2}$ in which the motor behaves more or less as it is intended to. Outside this range, the behaviour of the motor may be predictable, but definitely not conforming to a linear model that could have been derived for the motor.

Note that for the effects 2 . and 3 . the signal $\xi$ represents the effect of the exogenous inputs. This does not mean that $\xi$ contains these inputs directly. $\xi$ is additive to the inputs and outputs that are included in the description $S$. The exogenous inputs may very well enter the system physically at a location that does not coincide with the location of any of the signals in $w \in S$.

Up to this point no assumptions were made about the degree in which the outputs $y$ can be explained by the inputs $u$ for a $w=(u, y) \in S$. We could have taken $u$ to represent the prices of certain flowers at a market place and $y$ the number of employees called Piet in a company. The assumption that the outputs $y$ have to be explicable, at least to a certain extent, by the inputs $u$ is formalised next:
$\forall u,\left(\|u\| \leq d_{u} \Longrightarrow\|\xi\| \leq d_{\xi}\right) \triangleright$ Any feasible model $Q \in Q$ that specifies
a relationship between $u$ and $y$ has the property that the data can be decomposed as

$$
w=w_{Q}+\xi
$$

with $w_{Q} \in Q$. However, the output $y$ is only to a large extent explicable by $u$ if it holds that

$$
\begin{equation*}
\|\xi\| \leq d_{\xi} \ll\|w\| \tag{3.9}
\end{equation*}
$$

provided both inputs and outputs contribute in roughly equal amounts to $\|\xi\|$ and $\|w\|$. If these norms are completely dominated by the contribution of the inputs, the above statement is not true. In this case we can fit to a $w \in S$ a $\tilde{w} \in Q \in Q$ such that the inputs in $w$ and $\tilde{w}$ are equal but the outputs are completely different. This will nevertheless lead to $\|\xi\|=\|w-\tilde{w}\| \ll\|w\|$. A similar situation may occur if $\|w\|$ or $\|w-\tilde{w}\|$ is completely dominated by the outputs. Assuming that such domination by neither the inputs nor the outputs is the case, (3.9) has far-reaching implications.
As already stated, it means that $y$ can be explained to a large extent through $u$. (3.9) will therefore not apply if $u$ and $y$ are two signals that have nothing to do with each other. (3.9) implies moreover that the explanation of $y$ by $u$ takes the form of a relationship between $y$ and $u$ of the sort $Q$. Assuming once more that $Q$ consists of linear, time-invariant behaviours, (3.9) quantifies what it means that the non-linear system $S$ can be approximated by a linear, time-invariant process.
As was mentioned in the discussion leading to (3.8), it is in practice not reasonable to assume that (3.9) holds for every $w \in S$. What is actually meant is that $\xi$ should remain small if the inputs and outputs stay within a certain range. This latter requirement is intended to be implied by $\|u\| \leq d_{u}$. This will be explained further below.

The requirement

$$
\|u\| \leq d_{u} \quad \Longrightarrow \quad\|\xi\| \leq d_{\xi}
$$

links the theoretical descriptions in $Q$ with the physical reality in $S$. A few reality-checks seem therefore in order.

That $\|u\| \leq d_{u}$ can be expected to imply $\|\xi\| \leq d_{\xi} \ll\|w\|$ is not trivial. The argumentation for this consists of two steps. First it is considered whether there exists a range of input and output amplitudes, such that
a) $S$ can indeed be approximated sufficiently well by a system $Q \in \Omega$ so that in this range it holds $\|\xi\| \ll\|w\|$.
b) this range is large enough to contain the range of input- and output-amplitudes that can be expected in the system during the eventual closed-loop operation.

Second it will be discussed whether one can expect $\|u\| \leq d_{u}$ to be a sufficient condition for both inputs and outputs to stay in this range.

Starting with the first issue, is it possible to find a proper description $Q \in Q$ for the system in the input- and output-ranges of interest? Whether this can be achieved depends on the system $S$ and on Q. Practical experience shows for a lot of processes that the approximation of a real plant by a linear, timeinvariant description is good enough to get away with. However, practice can also provide examples in which this standard tool fails miserably. As far as the formalism is concerned, this is an open question. Note that the range of interest for the input and output amplitudes was formalised on page 45 by the set $R$.

Remark 3.1 In the previous parts with "ranges of interest" for inputs and outputs was more or less implicitly meant that the inputs and outputs should satisfy an upper bound. In practice it is also sensible to maintain a lower bound on the magnitude of $u$ : it is not uncommon that an actuator performs poorly for very small excitations or that the process simply does not respond to signals below a certain threshold. Dead zone, hysteresis and backlash are examples of such effects. These effect will not further be considered here.

Moving on to the second issue, is the condition $\|u\| \leq d_{u}$ sufficient to ensure that both inputs and outputs remain in a certain range and is the condition $\|u\| \leq d_{u}$ a reasonable one in the first place? For open-loop stable systems $S$ it is easy to see that the answer to both questions is "yes." Making sure that the answer is also affirmative for unstable systems $S$ is exactly what the stabilising controller $C_{\text {stab }}$ was introduced for. It may be assumed that the disturbances and the measurement noise are bounded. This implies for the signal $\xi_{23}$ in figure 3.1 on page 41

$$
\begin{equation*}
\left\|\xi_{23}\right\|<\infty . \tag{3.10}
\end{equation*}
$$

(See also equation (3.3) on page 41.) A bound on $u$ implies, together with assumptions (3.4) on page 42 and (3.10) above, indeed a bound on $y$. This can

$$
\begin{align*}
& \left.\begin{array}{cc}
S \in S \rightarrow & \Delta W \\
& \triangleright C_{\text {stab }} \\
& \Delta \forall u,\left(\|u\| \leq d_{u} \Longrightarrow\|\xi\| \leq d_{\xi}\right) \\
\triangleright \mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{p}}
\end{array}\right\} \rightarrow \begin{array}{l} 
\\
\end{array}  \tag{3.1}\\
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d}+" \Delta \Delta \\
d(\Delta) \leq d_{\Delta} \\
\triangleright \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ?
\end{align*}
$$

be seen as follows. From the fact that $C_{\text {stab }}$ stabilises $S$ follows by definition

$$
\begin{align*}
\left(\left\|u_{1}\right\|<\infty \wedge\|r\|<\infty \wedge\left\|\xi_{23}\right\|<\infty\right) \Longrightarrow & \\
& \left(\left\|u_{1}\right\|<\infty \wedge\left\|u_{2}\right\|<\infty \wedge\|y\|<\infty\right) \tag{3.11}
\end{align*}
$$

Given the assumptions that $r$ and $\xi_{23}$ are bounded, we have

$$
\left(\left\|u_{1}\right\|<\infty \Longleftrightarrow\|u\|<\infty\right) \Longrightarrow\|y\|<\infty
$$

$u_{1}$ is in practice a bounded signal for the same reason as $r$ is bounded: it is a variable specified by the user of the plant. It can now be concluded that bounding $u$ will bound $y$ and that it is reasonable to assume that $u$ can be bounded. This completes the discussion of the second issue.

This is the point in the formalism at which inputs are treated essentially different from outputs. One could replace the implication $\|u\| \leq d_{u} \Longrightarrow\|\xi\| \leq$ $d_{\xi}$ by the implication $\|w\| \leq d_{w} \Longrightarrow\|\xi\| \leq d_{\xi}$. However, keeping $u$ bounded is something that can be achieved simply by specifying bounded values for them, at least in the open-loop case. One can not specify that the outputs remain bounded. One can only apply an input and see what happens. In the closed-loop case it becomes more a matter of taste whether to require that $u$ is bounded or that $w$ is bounded: not all of $u$ can be considered free variables any more and one has to resort to arguments given above to show that all signals in $u$ remain bounded in practical circumstances. In exactly the same manner it could have been argued that all signals in $w$ remain bounded.

If the selection of inputs and outputs has been such, that the relationship between inputs and outputs is too weak for proper control, this will emerge here in the algorithm, even before the control design. In this case, choosing $d_{u}$ such that $u$ remains in the range of interest, a considerable part of $y$ will not be explicable by $u$, so $\xi$ will be a large signal and

$$
\|u\| \leq d_{u} \quad \Longrightarrow \quad\|\xi\| \leq d_{\xi} \ll\|w\|
$$

will not hold.
Consider once more equation (3.8) from page 46:

$$
\begin{equation*}
Q=\arg \min _{\bar{Q} \in Q} \max _{w \in S \cap R} \min _{\tilde{w} \in \tilde{Q}}\|w-\tilde{w}\| \tag{3.8}
\end{equation*}
$$

One can argue that this optimisation is a very hypothetical one. The set $S$ is not known and even if it were, it would be a difficult job to solve (3.8). However, things are not as bad as they may seem. The optimal $Q$ that is determined by. (3.8) is never used. It is only introduced to express that as much as possible of the process behaviour should be explained by a relation in $\mathcal{Q}$. In other words, the signal $\xi$ accounting for the fact that $S \notin Q$, for disturbances, measurement noise etc. should be made as small as possible. That this should be confined to a certain amplitude range for both inputs and outputs, expressed by the range $R$, is nothing new. During the input-design for an identification experiment the amplitudes of the inputs have to be chosen exactly such, that input- and output-amplitudes are in a practically relevant range. During the input design a value for $d_{t}$ is therefore already implicitly determined. All that remains to be determined is a bound on the effects 1 . to 3 . on page 46 . After a number of identification runs with the plant, one should be able to determine at least an upper bound for these effects.
$W$ (cont.) $\triangleright$ The trajectories in $W$ should satisfy

$$
\|u\| \leq d_{u}
$$

$\mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{p}} \triangleright$ Prior knowledge that is available of the system is represented by $\mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{p}}$. If it is known for example that the DC-gain of a certain SISO transfer in $S$ is equal to 1 , while all others are known to be $0, \mathcal{P}_{2}$ will consist of all behaviours in $Q$ that are consistent with this information. If it is further known that there is a resonance peak near a certain known frequency, $\mathcal{P}_{3}$ will contain all members of 2 that can be considered to be consistent with this prior knowledge, etc.

To decide whether an element of $Q$ is consistent with a certain type of prior knowledge about the system $S$, the influence of $\xi$ has to be accounted for because prior knowledge is usually stated in terms of the noise-free, disturbance-free system.

Now the experimental data and all prior knowledge in the formalism have been introduced. This information will subsequently be processed to identified models and uncertainty bounds.
$\mathcal{P} \triangleright$ Apart from $\mathscr{P}_{2}, \ldots, \mathscr{P}_{N_{\mathcal{P}}}$ we introduce $\mathcal{P}_{0} \subseteq \mathcal{Q}$ containing all elements of $Q$ consistent with $W$ and a noise level $\rho$. The noise level $\rho$ will be specified

$$
\begin{align*}
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
D \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

later. $Q \in Q$ is said to be consistent with $W$ and $\rho$ if and only if

$$
\forall w \in W, \exists \tilde{w} \in Q, \quad\|w-\tilde{w}\| \leq \rho
$$

Until it is specified what noise level is to be used, we take $\mathscr{P}_{0}=P_{0}(\rho)$ where $P_{0}$ is a function of an as yet unknown argument $\rho$ :

$$
\begin{equation*}
P_{0}: \mathbb{R}_{+} \rightarrow 2^{Q}, \quad \rho \mapsto\{Q \in \mathbb{Q} \mid \forall w \in W \exists \tilde{w} \in Q, \quad\|w-\tilde{w}\| \leq \rho\} \tag{3.12}
\end{equation*}
$$

The fact that the system is stabilised by $C_{s t a b}$ is represented by the subset $\mathcal{P}_{1}$ of $Q$ containing all systems in $Q$ that are indeed stabilised by $C_{\text {stab }}$. The role of $\xi$ in this will also be considered later.
All prior knowledge and experimental data about the system can now be summarised by stating that $Q$, the manifestation of $S$ in $S$ (see (3.8)), is a member of the subset

$$
\mathcal{P}:=\bigcap_{i=0}^{N_{\mathcal{P}}} \mathcal{P}_{i}
$$

of $Q$.
This set will be called the process uncertainty set. All elements of $\mathcal{P}$ are non-falsifiable by the experimental data or the prior knowledge.

The Venn-diagram of figure 3.3 shows the relation between $Q, \mathcal{P}_{0}, \ldots, \mathcal{P}_{N_{\mathcal{P}}}$ and $\mathcal{P}$ for $N_{\rho}=3$.

Remark 3.2 The aim of model uncertainty bounding techniques is to find a description of $\mathcal{P}$ that is as tight as possible. It has been argued in section 2.3.2 that this description should generally be taken such that it contains all of $\mathcal{P}$,
although descriptions containing at least one element of $\mathcal{P}$ are encountered in literature as well. Robust control design techniques should design a controller having desired properties for all elements of $\mathcal{P}$. This will be discussed further in section 3.3.3.
$\mathcal{M}_{\text {set }} \triangleright$ Existing techniques for robust control design are based on a nominal model and a bound on the error or uncertainty in this nominal model. It is the job of the identification step to select a nominal model from a set of candidate nominal models. This set of candidate models, normally called simply the model set, is denoted by $\mathcal{M}_{\text {set }}$.

For all $M \in \mathcal{M}_{\text {set }}$, there exists a causal relationship from inputs to outputs. A specific technique may or may not require that $\mathcal{M}_{\text {set }} \subset \mathcal{Q}$. In general $\mathcal{M}_{\text {set }} \cap \Omega \neq 0$.

Example (cont.): Suppose, that one wants to model the SISO transfer from $U_{1}$ to $I_{1}$ of the asynchronous motor by an output error model:

$$
\begin{align*}
y(k) & =\frac{b(\zeta)}{a(\zeta)} u(k)+\epsilon(k) \\
a(\zeta) & =1+a_{1} \zeta^{-1}+\cdots+a_{n} \zeta^{-n}  \tag{3.13}\\
b(\zeta) & =b_{0}+b_{1} \zeta^{-1}+\cdots+b_{n} \zeta^{-n} \\
n & \in \mathbb{N} .
\end{align*}
$$



Figure 3.3: Venn diagram of $Q, \mathcal{P}_{0}, \ldots, \mathcal{P}_{N \mathcal{F}}$ and $\mathcal{P}$

$$
\begin{align*}
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
\nabla \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

$\epsilon(k)$ is assumed to be a white noise sequence, although asymptotically unbiased results will be obtained in output error identification if it is coloured. One may additionally require that for some $m \in \mathbb{N}, m<n$

$$
\forall k=0, \ldots, m, \quad b_{k}=0
$$

The set $\mathcal{M}_{\text {set }}$ is now equal to the set of all linear, time-invariant processes where the relation between $u$ and $y$ can be written in the form (3.13).
(Note that the noise sequence $\epsilon$ is left out of the description of $\mathcal{M}_{\text {set }}$.)
For the purpose of model uncertainty bounding one may further require that $Q$ is the set of linear, time-invariant processes whose impulse response parameters fall off with a certain exponential rate. Because the model set $\mathrm{M}_{\text {set }}$ contains in this example even unstable models, it holds

$$
\mathcal{M}_{\text {set }} \not \subset Q
$$

For the class of parameter set estimation techniques, it holds on the other hand

$$
Q:=\mathcal{M}_{\text {set }}
$$

$\mathcal{M}_{i d}$ and $J_{i d} \triangleright$ As already mentioned, identification should select an optimal model from the model set $\mathcal{M}_{\text {set }}$. This optimal model will be denoted $\mathcal{M}_{i d}$ and referred to as the nominal model.

In fact the possibility is left open that the identification selects more than one optimal model. This hardly complicates the formalism and is slightly more general. Therefore $\mathcal{M}_{i d}$ uses a script " M " and it would be more accurate to call $\mathcal{M}_{i d}$ the set of nominal models. However, as the majority of identification procedures yields a single nominal model, our verbal descriptions will be such that $\mathcal{M}_{i d}$ contains only one model.

Which model in the model set is to be considered the optimal model should depend on many factors. The degree in which $\mathcal{M}_{i d}$ is able to explain the data $W$ is one of the most important ones. The capability of $\mathcal{M}_{i d}$ to explain the data should be balanced with the complexity of $\mathcal{M}_{i d}$ : for models of sufficiently high order for instance, any finite data set can be explained completely by a linear, time-invariant model. For models for robust control design, the selection of the optimal model should further be based on "control relevant" arguments.
It is not the aim here to delve deep into the problems related to formulating an optimal notion of optimality. For the purpose of this formalism, the identification criterion function $J_{i d}$ attributes a cost to each model $M \in$ $\mathcal{M}_{\text {set }}$ for given data $W$. This is achieved by requiring $J_{i d}$ to be a function from $M_{\text {set }} \times 2^{W^{T}}$ to a set $\mathbb{E}$, where $\mathbb{E}$ is a totally ordered set:

$$
J_{i d}: \mathcal{M}_{\text {set }} \times 2^{W^{T}} \rightarrow \mathbb{E}
$$

For $J_{i d}$ to be suitable as a misfit criterion, it is required that

$$
\begin{aligned}
& \forall M_{1}, M_{2} \in \mathcal{M}_{\text {set }}, W \subset W^{T}, \\
& \\
& \qquad M_{1} \subset M_{2} \Longrightarrow J_{i d}\left(M_{1}, W\right) \geq J_{i d}\left(M_{2}, W\right)
\end{aligned}
$$

and

$$
\begin{aligned}
& \forall M \in \mathcal{M}_{\text {set }}, W_{1}, W_{2} \subset W^{T}, \\
& \qquad W_{1} \subset W_{2} \Longrightarrow J_{\mathrm{id}}\left(M, W_{1}\right) \leq J_{\mathrm{id}}\left(M, W_{2}\right)
\end{aligned}
$$

It is customary that $\mathbb{E}=\mathbb{R}$ or $\mathbb{E}=\mathbb{R}_{+}$, i.e. $J_{i d}$ often takes its values in $\mathbb{R}$ or $\mathbb{R}_{+}$.

We now simply have

$$
\mathcal{M}_{i d}=\arg \min _{M \in \mathcal{M}_{\mathrm{set}}} J_{i d}(M, W)
$$

Because $\mathcal{M}_{\text {set }}$ is not necessarily a subset of $Q$ it may turn out that

$$
\mathcal{M}_{i d} \not \subset Q
$$

$$
\begin{align*}
& \left.\left.\begin{array}{cc}
\Delta \in \mathcal{S} \rightarrow & \Delta W \\
& \triangleright C_{\text {stab }} \\
& \Delta \forall u,\left(\|u\| \leq d_{u} \Longrightarrow\|\xi\| \leq d_{\xi}\right) \\
\triangleright \mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{\mathcal{P}}}
\end{array}\right\} \rightarrow \mathcal{P} \subset \mathcal{Q}\right\} \rightarrow  \tag{3.1}\\
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{\mathrm{id}} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{\mathrm{id}} "+" \Delta \Delta \\
d(\Delta) \leq d_{\Delta} \\
D \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathrm{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ?
\end{align*}
$$

$\Delta, d(\Delta)$ and $d_{\Delta} \triangleright$ Not only should an estimate for the nominal model be given. For robust control design one also wants to have an indication of the uncertainty or error in the nominal model. For this purpose the model uncertainty $\Delta$ is introduced. $\Delta$ "interacts" in some way, not specified by this formalism, with the nominal model to yield a perturbed model $\mathcal{M}_{i d}$ " $+" \Delta$. All perturbed models are assumed to be causal models.
$\Delta$ is assumed to be taken from some set $\Delta$. It is not possible to give a precise description of $\Delta$ without specifying what the operation " + " means, so the formalism leaves the exact meaning of $\Delta$ open. $\Delta$ should be such, that all its elements can be used to perturb the nominal model in the way described above.
It is customary to bound the model uncertainty by bounding $\Delta$ in some measure. This measure is represented here by the mapping $d: \Delta \rightarrow \mathbb{D}$, where $I D$ is some partially ordered set. Bounding $\Delta$ in this measure is then formalised by

$$
d(\Delta) \leq d_{\Delta}
$$

Example (cont.): The most widely used perturbation is probably that of additive model uncertainty. For this kind of perturbations, $\Delta$ is a transfer function with the same number of inputs and outputs as the nominal model $\mathcal{M}_{i d}$ and the perturbed model is simply obtained by the addition of $\mathcal{M}_{i d}$ and $\Delta$. Taking into account that $\mathcal{M}_{i d}$ was actually a set of nominal models, this translates to

$$
\mathcal{M}_{i d} "+" \Delta:=\left\{M+\Delta \mid M \in \mathcal{M}_{i d}\right\}
$$

$M$ being a linear model and assuming $\Delta$ is also linear, we may define the addition of two behaviours simply by associating, say, an impulse response sequence
to both $M$ and $\Delta$, adding those sequences and translating the resulting impulse response sequence back to a behaviour.

Many other forms of perturbations have been reported in literature: multiplicative uncertainty at the inputs or outputs, additive coprime factor uncertainty, etc.

Another way to bound model uncertainty is to bound the uncertainty in the model parameters. This will be developed further for the output error model structure given by (3.13). Let $\mathcal{M}_{i d}$ consist of one model, denoted here in transfer function form,

$$
M^{\prime}(z)=\frac{b_{0}^{\prime}+b_{1}^{\prime} z^{-1}+\ldots b_{n}^{t} z^{-n}}{1+a_{1}^{\prime} z^{-1}+\cdots+a_{n}^{\prime} z^{-n}}
$$

We may then proceed to define

$$
\begin{aligned}
\Delta & =\mathbb{R}^{2 n+1} \\
\mathcal{M}_{\mathrm{id}} "+" \Delta & =\mathcal{M}_{i d} "+"\left(\Delta b_{0}, \ldots, \Delta b_{n}, \Delta a_{1}, \ldots, \Delta a_{n}\right)^{T}= \\
& =\left\{\frac{\left(b_{0}^{\prime}+\Delta b_{0}\right)+\left(b_{1}^{\prime}+\Delta b_{1}\right) z^{-1}+\cdots+\left(b_{n}^{\prime}+\Delta b_{n}\right) z^{-n}}{1+\left(a_{1}^{\prime}+\Delta a_{1}\right) z^{-1}+\cdots+\left(a_{n}^{\prime}+\Delta a_{n}\right) z^{-n}}\right\}
\end{aligned}
$$

and

$$
d: \mathbb{R}^{2 n+1} \rightarrow \mathbb{R}^{4 n+2}, \Delta \mapsto\left[\Delta^{T},-\Delta^{T}\right]^{T}
$$

so that

$$
D=\mathbb{R}^{4 n+2}
$$

The partial ordering on $\mathscr{D}$ as used by $d(\Delta) \leq d_{\Delta}$ is then defined as

$$
\begin{align*}
\left(x_{1}, \ldots x_{4 n+2}\right)<\left(y_{1}, \ldots, y_{4 n+2}\right): \Longleftrightarrow & \forall k=1, \ldots, 4 n+2, x_{k}<y_{k}
\end{align*}
$$

and

$$
\begin{align*}
\left(x_{1}, \ldots x_{4 n+2}\right)=\left(y_{1}, \ldots, y_{4 n+2}\right): & \Longleftrightarrow \\
& \forall k=1, \ldots, 4 n+2, x_{k}=y_{k} \tag{3.15}
\end{align*}
$$

Now let

$$
d_{\Delta}=\left(\bar{b}_{0}, \ldots, \bar{b}_{n}, \bar{a}_{1}, \ldots, \bar{a}_{n},-\underline{b}_{0}, \ldots,-\underline{b}_{n},-\underline{a}_{1}, \ldots,-\underline{a}_{n}\right)^{T} .
$$

The set

$$
\mathcal{M}_{r e l}:=\left\{M "+" \Delta \mid M \in \mathcal{M}_{i d}, \Delta \in \Delta, d(\Delta) \leq d_{\Delta}\right\}
$$

$$
\begin{align*}
& S \in \mathcal{S} \rightarrow \quad D W \\
& \left.\begin{array}{cc}
\triangleright C_{\text {sta } b} \\
\triangleright \forall u,\left(\|u\| \leq d_{u} \Longrightarrow\right. \\
\triangleright \mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{\mathfrak{J}}}
\end{array}\|\xi\| \leq d_{\xi}\right)\{\rightarrow \quad \mathcal{P} \subset Q\} \rightarrow  \tag{3.1}\\
& \rightarrow \underset{\text { D requirements }}{\left\{\begin{array}{c}
\mathcal{M}_{\text {id }} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{\text {id }}+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
\text { requin }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? ~}
\end{align*}
$$

(see below) contains those processes

$$
\tilde{M}(z)=\frac{\tilde{b}_{0}+\tilde{b}_{1} z^{-1}+\ldots \tilde{b}_{n} z^{-n}}{1+\tilde{a}_{1} z^{-1}+\cdots+\tilde{a}_{n} z^{-n}}
$$

for which it holds

$$
\begin{array}{ll}
\forall k=0, \ldots, n & \tilde{b}_{k} \in\left[b_{k}^{\prime}+\underline{b}_{k}, b_{k}^{\prime}+\bar{b}_{k}\right] \quad \wedge \\
\forall k=1, \ldots, n & \tilde{a}_{k} \in\left[a_{k}^{\prime}+\underline{a}_{k}, a_{k}^{\prime}+\bar{a}_{k}\right] . \tag{3.16}
\end{array}
$$

(3.16) represents an uncertainty bound that could have been the result of a parameter set estimation technique.
(3.14) and (3.15) specify a partial ordering and not a total ordering. For example, neither of the three following relations is considered true:

$$
\begin{gathered}
(0,1)^{T}<(1,0)^{T} \\
(0,1)^{T}=(1,0)^{T} \\
(0,1)^{T}>(1,0)^{T}
\end{gathered}
$$

The set of relevant processes is defined as

$$
\mathcal{M}_{\mathrm{rel}}\left(d_{\Delta}\right):=\left\{M "+" \Delta \mid M \in \mathcal{M}_{\mathrm{id}}, \Delta \in \Delta, d(\Delta) \leq d_{\Delta}\right\}
$$

The dependence on $d_{\Delta}$ has been made explicit in this notation. These processes are called relevant because they contain those processes that are relevant to the step of robust control design. As far as this design is concerned, the true system can be any element of $\mathcal{M}_{\text {rel }}$. Often one takes for $d_{\Delta}$ the smallest value such that

$$
\begin{equation*}
\mathcal{M}_{\mathrm{rel}}\left(d_{\Delta}\right) \supset \mathcal{P} \tag{3.17}
\end{equation*}
$$

although (Zhou and Kimura, 1993; Zhou and Kimura, 1994) take the approach that $d_{\Delta}$ should be the minimal value such that

$$
\begin{equation*}
\mathcal{M}_{r e l}\left(d_{\Delta}\right) \cap \mathcal{P} \neq \emptyset \tag{3.18}
\end{equation*}
$$

Figure 3.4 shows what figure 3.3 on page 53 could look like for two different approaches to determine $\mathcal{M}_{r e l}\left(d_{\Delta}\right) . \mathcal{M}_{r e l, a}$ corresponds to equation (3.17), $\mathcal{M}_{r e l, b}$ to equation (3.18).


Figure 3.4: Different approaches for $\mathcal{M}_{\text {rel }}$

The role that $\mathcal{P}$ and $\xi$ and consequently also $\mathcal{M}_{\text {rel }}\left(d_{\Delta}\right)$ and $\xi$ should play in robust control design will be considered in the next section.

## Summary of modelling aspects

In several steps a representation for the true system $S$ has been derived, consisting of a nominal model $\mathcal{M}_{i d}$, a process uncertainty $\Delta$ and a signal $\xi$ containing "all that remains." This is schematically shown in figure 3.5. The outer-most block represents the true system $S$. Disturbances and measurement noise are considered here to be part of $S$. The signal $\xi$ represents those parts of $S$ that do not fit in $Q$. The block marked $Q$ represents the parts of $S$ that do fit in Q. For a certain $\Delta \in \Delta$ such that $d(\Delta) \leq d_{\Delta}$ the interconnection of $\mathcal{M}_{i d}$ and $\Delta$ should be identical to the block marked with a $Q$. Because neither $\mathcal{M}_{i d}$ nor $\mathcal{M}_{i d} "+" \Delta$ need to be an element of $Q$, the blocks $\mathcal{M}_{i d}$ and $\Delta$ stick out of the block 2 .

$$
\begin{align*}
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d}+" \Delta \Delta \\
d(\Delta) \leq d_{\Delta} \\
\triangleright \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ? \tag{3.1}
\end{align*}
$$

Note that the true system is described by the nominal model(s) $\mathcal{M}_{i d}$, the uncertainty $\Delta$ and its bound $d_{\Delta}$ and the bound $d_{\xi}$ on $\xi$. This will be further discussed in section 3.3.2.

## Control design, implementation and validation

Not all of the symbols in (3.1) have been introduced yet. The symbols representing the control design still need to be discussed. These parts of the formalism are much less detailed than the ones discussed so far, because the formalism is mainly concerned with identification and uncertainty bounding. Because these are performed with the explicit goal of applying the results for robust control design, it is felt that the control design and implementation should have their place in (3.1) too. It is beyond the scope of this thesis to give a detailed account of these steps.


Figure 3.5: Processes and signals in the formalism
requirements $\triangleright$ These are the requirements the controller has to meet. They can be very diverse: robustness, disturbance rejection, tracking behaviour, cost of the implemented control and computing time needed to calculate a control action are all factors that determine the quality of a control design. Many other factors may play a role as well.
$\mathrm{C}_{0} \triangleright$ This is the set of candidate controllers from which the control design procedure has to pick the optimal one. Often controllers are taken from the set of linear, time-invariant systems. In practice they may have to meet an a priori bound on their McMillan degree due to requirements following from the way in which they will be implemented. $C_{0}$ contains all controllers satisfying these requirements in as far as the control design takes them into account.
$C_{0} \triangleright$ The controller that is eventually chosen from $C_{0}$ is denoted $C_{0}$. This is the controller that, hopefully, is designed such that it meets all requirements, or finds the best possible compromise between these.
$\Delta_{C}, d_{\Delta_{C}} \triangleright$ Due to variations in components, finite word length in the calculation of results, the analog-to-digital or digital-to-analog conversion and possibly other effects, the controller that is actually implemented is not identical to $C_{0}$, although we may hopefully assume that it is a good approximation of it. Because there is also some uncertainty involved in the actual behaviour of the controller, the actual implementation is one out of a set of possible implementations.
The difference between the designed controller and the implemented controller is described, analogously to the difference between the nominal model and the true system, by an uncertainty $\Delta_{C}$. This uncertainty is bounded by

$$
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
$$

The controller that is actually implemented is denoted $C_{0}$ " + " $\Delta_{C}$, where " + " denotes an interconnection similar to the interconnection of the nominal model $\mathcal{M}_{i d}$ and the process uncertainty $\Delta$. It is reasonable to assume, that

$$
C_{0} "+" \Delta_{C} \approx C_{0}
$$

although it may be wise to keep in mind, that

$$
C_{0} "+" \Delta_{C} \neq C_{0}
$$

$O K ? ~ T h e ~ u l t i m a t e ~ c r i t e r i o n ~ t o ~ j u d g e ~ w h e t h e r ~ c o n t r o l ~ d e s i g n, ~ i d e n t i f i c a-~$ tion, uncertainty bounding, etc. have met their goals is to evaluate the performance of the controller on the real process. If the performance meets the

$$
\begin{align*}
& \left.\left.S \in \mathcal{S} \rightarrow \begin{array}{c}
\triangleright W \\
\triangleright C_{\text {stab }} \\
\\
\\
\triangleright \forall u,\left(\|u\| \leq d_{u} \Longrightarrow\|\xi\| \leq d_{\xi}\right) \\
\triangleright \mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{\mathcal{F}}}
\end{array}\right\} \rightarrow \mathcal{P \subset Q} \begin{array}{l} 
\\
\\
\\
\\
\\
\\
\\
\end{array}\right\}  \tag{3.1}\\
& \rightarrow\left\{\begin{array}{c}
\mathcal{M}_{i d} \subset \mathcal{M}_{\text {set }} \\
\mathcal{M}_{i d} "+" \Delta \\
d(\Delta) \leq d_{\Delta} \\
D \text { requirements }
\end{array}\right\} \rightarrow C_{0} \in \mathcal{C}_{0} \rightarrow\left\{\begin{array}{c}
C_{0} "+" \Delta_{C} \\
d\left(\Delta_{C}\right) \leq d_{\Delta_{C}}
\end{array}\right\} \rightarrow O K ?
\end{align*}
$$

requirements, all previous steps have apparently given satisfactory results. If not, lots of options are open: a better implementation $C$ of the controller may solve the problem, or a better design for the controller $C_{0}$. Maybe the requirements are not realistic and need to be reviewed. The model set $\mathcal{M}_{\text {set }}$ or the uncertainty bound $d_{\Delta}$ may be too tight, the uncertainty description denoted by " + " may not be appropriate for the situation at hand, the identification data $W$ may not be rich enough. Some of the prior knowledge $\mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{\rho}}$ may be incorrect. Finally, one may decide to redesign the system $S$ that needs to be controlled or choose another set of inputs and outputs. In short, any aspect of the formalism may need to be revised if it turns out in the end that the controlled behaviour of the plant does not meet the requirements.

The reader is assumed to be familiar by now with the symbols in the algorithm. The repetition of the formalism on every left-hand side page is therefore terminated from this point onwards.

### 3.3 Discussion

### 3.3.1 Iterations

The previous section may have given the impression that the steps in the framework are all one-shot algorithms, apart from a final validation at the end. It should be stressed that this is not the case. In practice, validation does not only take place at the very end, but after about every step. The nominal models are normally validated immediately after the identification and it may be decided then already that the nominal models require refinement. Also the model uncertainty description and the uncertainty bounds should be subjected to critical evaluation before passing them on to the control design. It is also not likely that the first attempt at a control design is immediately implemented at the final plant.

The actual steps may further not be performed in the order indicated by (3.1). The values for $d_{u}$ or $d_{\xi}$ may very well be determined after the identification of nominal models. The nominal model may also give an indication of how fast, say, the impulse response of the true system falls off. This may be reflected in $Q$ and/or $\mathcal{P}$.

### 3.3.2 Relation between $\xi$ and $\Delta$

It has already been mentioned that the uncertainty in the system's behaviour is bounded both by $d(\Delta) \leq d_{\Delta}$ and $\|\xi\| \leq d_{\xi}$. In this section it is argued why this is the case and what the relation between $\xi$ and $\Delta$ should look like.

Write

$$
\xi=\xi_{1}+\xi_{23},
$$

where $\xi_{23}$ accounts for disturbances and measurement noise and $\xi_{1}$ for the fact that even in the noise-free, disturbance-free case it would hold $S \notin \mathcal{Q}$. This is in line with the partitioning of $\xi$ into three effects on page 46 . The effect mentioned under 1. is represented by $\xi_{1}$ and the other two by $\xi_{23}$. No distinction is made between effect 2 . (disturbances) and 3. (measurement noise), because they are equivalent as far as identification and model uncertainty bounding is concerned.

## Effect of noise and disturbances

Consider first the exclusive effect of $\xi_{23}$, so assume $\xi_{1}=0$, i.e. we know that under noiseless, disturbance-free circumstances it would hold $S \in \mathbb{Q}$. Because it was assumed that

$$
\forall w \in W, \quad\|u\| \leq d_{u}
$$

the implied inequality

$$
\|\xi\|=\left\|\xi_{23}\right\| \leq d_{\xi}
$$

applies for all $w \in W$.
It was stated that $\mathcal{P}_{0}$ contains all elements of $Q$ that are consistent with the data $W$, given the uncertainty in this data. Which elements of $Q$ are to be considered consistent with $W$ depends on a noise level $\rho$ :

$$
P_{0}(\rho)=\{Q \in \mathcal{Q} \mid \forall w \in W \exists \tilde{w} \in Q, \quad\|w-\tilde{w}\| \leq \rho\} .
$$

An obvious candidate for the noise level $\rho$ is $d_{\xi}$, the bound on $\xi . P_{0}\left(d_{\xi}\right)$ contains all elements of $Q$ that can not be falsified by the data and the bound $d_{\xi}$ on the uncertainty in the data. If we dispose of exact data, i.e. $d_{\xi}=0$, there may still
be several elements in $Q$ that could have generated this data. However, $P_{0}\left(d_{\xi}\right)$ will grow if $d_{\xi}$ increases and if $\mathcal{P}_{0}=P_{0}\left(d_{\xi}\right)$ this will generally imply that $\mathcal{P}$ grows. If the latter implication would not hold, the process uncertainty set $\mathcal{P}$ would not depend on the data and we might as well have omitted collecting the data in the first place. However, this may happen if the a priori information is very restricting.

Another candidate for $\rho$ is

$$
\begin{equation*}
d_{\min }:=\min \left\{d \in \mathbb{R}_{+} \cup\{0\} \mid P_{0}(d) \neq \emptyset\right\} \tag{3.19}
\end{equation*}
$$

It is assumed for simplicity that the minimum in (3.19) exists. The idea is to take $d_{m i n}$ as small as possible given the constraint that $P_{0}\left(d_{m i n}\right)$ should be non-empty and to take the noise level involved in the decision whether a $Q \in Q$ is consistent with the data $W$ equal to this $d_{m i n}$. For this value of $\rho$ as little of the data uncertainty as possible is expressed in $P_{0}(\rho)$. Contrary to $d_{\xi}, d_{m i n}$ is not known a priori.

To decide whether to use $\mathcal{P}_{0}=P_{0}\left(d_{\xi}\right)$ or $\mathcal{P}_{0}=P_{0}\left(d_{m i n}\right)$ as the proper interpretation for $\mathcal{P}_{0}$, some further consideration is required. As it was assumed that $\xi_{1}=0$, it is known that the true (noise- and disturbance-free) relation between $u$ and $y$ can be expressed by a $Q \in \mathbb{Q}$. Consequently the true system can be represented in the form

$$
\begin{equation*}
S^{\prime} \in \mathcal{M}_{i d} "+" \Delta \tag{3.20}
\end{equation*}
$$

where $S^{\prime}$ is the noise-free, disturbance-free part of $S$.
Remark 3.3 Actually (3.20) is not certain yet but (3.20) needs to be made sure: the set $\Delta$ from which $\Delta$ 's are taken and the perturbed system $\mathcal{M}_{i d} "+" \Delta$ should be taken such that

$$
\mathcal{M}_{i d} "+" \Delta:=\bigcup_{\Delta \in \Delta} \mathcal{M}_{i d} "+" \Delta \supset Q
$$

Otherwise we may end up in a situation that part of the system behaviour is represented by $\mathcal{P}$, but not by $\mathcal{M}_{i d}$ " + " $\Delta$ for a $\Delta \in \Delta$ such that $d(\Delta) \leq d_{\Delta}$, so that the robust control design does not take it into account. Part of the model uncertainty was lost in the process of bounding it.

One may argue that this is allowed, as long as the aspects that are dropped in this way are not relevant for the control design. Although there is probably much truth in this, it is in principle better to let the control design decide this. If these aspects are really not relevant, they will - hopefully - not influence the control design. On the other hand, it may not be possible to represent the aspects that are dropped accurately by the interconnection $\mathcal{M}_{i d}$ " + " $\Delta$ and a bound $d_{\Delta}$ or it may take a large effort to do so.

Therefore, the point of view expressed above can not be considered to be "carved in stone."

It is known that $P_{0}\left(d_{\xi}\right)$ contains the manifestation of $S$ in $Q$. Consequently, if $\mathscr{P}_{0}=P_{0}\left(d_{\xi}\right)$ then $\mathcal{P}$ will contain the manifestation of $S$ in $Q$. This manifestation has the property that for all signals in the relevant amplitude ranges it holds

$$
\|\xi\| \leq d_{\xi} .
$$

Therefore the control design has to count on disturbance signals $\xi$ of a size $\|\xi\|$ that is less than $d_{\xi}$.

If it is decided on the other hand to use $P_{0}=P_{0}\left(d_{\min }\right)$ as the interpretation for $\mathcal{P}_{0}$, then the value of $d_{\Delta}$ will possibly be smaller, which has in itself a beneficial effect on the control design. However, assume that $d_{\min }=0$, so there exist systems in $Q$ that could have generated the noisy data in $W$. The fact that $d_{\min }$ can be taken equal to zero does not mean that the data in $W$ is actually noiseless. It means that the systems in $P_{0}\left(d_{m i n}\right)$ consider, unrightfully, the noise realisations present in $W$ to belong to the input-output behaviour of $S$. Suppose now that the actual noise realisation present in $W$ was $\xi^{\prime}$, with $\left\|\xi^{\prime}\right\|=d_{\xi}$. Suppose further that the part of $\xi^{\prime}$ corresponding to the inputs $u$ is equal to zero and that the systems in $Q$ are time-invariant. If, at a later time, the same inputs as those present in $W$ are applied to the models in $P_{0}\left(d_{\text {min }}\right)$, the same noise realisation $\xi^{\prime}$ will be reproduced, only shifted in time. If the same inputs would be applied later to the true system $S$, a noise realisation equal to a shifted version of $-\xi^{i}$ could actually occur. Apparently there is a difference $2 \xi^{\prime}$ between the signals observed for the true system and those generated by the models in $P_{0}\left(d_{\text {min }}\right)$. The control design therefore has to count on a noise level at least as large as $2 d_{\xi}$ !

The above line of reasoning is depicted in figure 3.6. In this figure, $\tilde{w}$ represents the input and output trajectories that the system would have generated during the experiments if there were no noise and disturbances. Due to noise and disturbances represented by the vector $\xi^{\prime}$, the experimental data set does not consist of the trajectories $\tilde{w}$ but of the trajectories $w_{1}$. If at a later time the same experiment would be repeated, the noise and disturbance free process behaviour would again be $\tilde{w}$ in the example given above. If the noise and disturbances are now equal to $-\xi^{\prime}$, the experimental data will now consist of $w_{2}$ instead of $\tilde{w}$. However, all behaviours in $P_{0}\left(d_{\min }\right)$ will reproduce the behaviour $w_{1}$ under these conditions: apparently there is a difference $2 \xi^{\prime}$ between the observed process behaviour and the behaviour of the models in $P_{0}\left(d_{\text {min }}\right)$.

If this situation is analysed a little further, we see that the noise that the control design encounters can be split into two components $\xi_{\alpha}$ and $\xi_{\beta}$. $\xi_{\alpha}$ represents the difference between the signals generated by the systems in $P_{0}\left(d_{m i n}\right)$ and those generated by the real manifestation of $S$ in $Q$. $\xi_{\beta}$ represents the difference between the signals generated by the manifestation of $S$ and those generated by $S$ itself. It can be relied on that $\left\|\xi_{\beta}\right\|$ will not exceed $d_{\xi}$ if the inputs satisfy $\|u\| \leq d_{u}$, but the same does not apply for $\left\|\xi_{\alpha}\right\|$. For input


Figure 3.6: Increase of noise level for $\mathcal{P}_{0}\left(d_{\text {min }}\right)$
signals completely different than those in $W$, the value of $\left\|\xi_{\alpha}\right\|$ may very well be larger than $d_{\xi}$. Consequently, the bound on $\xi$ of $2 d_{\xi}$ derived above can be guaranteed only for those input trajectories that are present in $W$. The control design has to expect noise levels even larger than $2 d_{\xi}$.

The problem of determining the maximum noise level that the control design can expect will not be considered further here, because it will be argued in section 3.3 .4 that it is to be preferred to bound the model uncertainty by $d_{\Delta}$ instead of by $d_{\xi}$.

Summarising the above line of reasoning, the influence of $\xi_{23}$ can be accounted for by a bound on $\Delta$ and if we do this, the bound on the disturbances need not be increased for the control design.

## Effect of undermodelling

In the previous paragraphs it was assumed that $\xi$ accounted only for noise effects and disturbances, so $\xi_{1}=0$. Now consider the other extreme, namely

$$
\xi_{1} \neq 0 \wedge \xi_{23}=0
$$

In this case we are purely looking for what was called the manifestation of $S$ in $Q$, no noise or disturbances are present in the system. It should be stressed that there is no $\Delta \in \boldsymbol{\Delta}$ such that

$$
S=\mathcal{M}_{i d} "+" \Delta
$$

One may therefore decide to make $d_{\Delta}$ as small as possible. This can be done by taking $\rho$ in (3.12) equal to $d_{\text {min }}$ defined in (3.19) because the smallest possible value of $\rho$ will results in the smallest bound on $\Delta$.

This is probably not a good idea. All that can be concluded from $W$ and the bound $d_{\xi}$ is that the manifestation of $S$ in $Q$ must be a member of $P_{0}\left(d_{\xi}\right)$ :

$$
P_{0}\left(d_{\xi}\right)=\left\{Q \in Q \mid \forall w \in W, \exists \tilde{w} \in Q, \quad\|w-\tilde{w}\| \leq d_{\xi}\right\}
$$

If one uses an interpretation for $\mathcal{P}_{0}$ such that it does not necessarily contain the manifestation of $S$ in $Q$, and $\mathcal{P}_{0}=P_{0}\left(d_{\min }\right)$ is such an interpretation, then one can not be sure any more that there is an element $Q^{\prime}$ in $\mathcal{P}_{0}$, such that

$$
\forall w=(u, y) \in S \quad\|u\| \leq d_{u} \Rightarrow\left(\exists \tilde{w} \in Q^{\prime}, \quad\|w-\tilde{w}\| \leq d_{\xi}\right)
$$

the noise level that the control design must count on, on top of the uncertainty bounded by $d_{\Delta}$, exceeds $d_{\xi}$. In this case the signal $\xi$ can again be split into a component $\xi_{\alpha}$ and $\xi_{\beta}$ where $\xi_{\alpha}$ accounts again for the mismatch between the systems in $\mathcal{P}_{0}$ on the one hand and the manifestation of $S$ in $Q$ on the other and $\xi_{\beta}$ accounts for the mismatch between $S$ and its manifestation in $Q$. $\left\|\xi_{\beta}\right\|$ will not exceed $d_{\xi}$, but $\left\|\xi_{\alpha}\right\|$ may be larger. Again a smaller value of $d_{\Delta}$ has been achieved at the expense of a larger value of $d_{\xi}$. It was mentioned above that this is undesirable, so in the case $\xi_{23}=0$ one should use $\rho=d_{\xi}$ too.

## Conclusion

The effects of $\xi_{1}$ and $\xi_{23}$ now turn out to lead to the same conclusion, namely that $P_{0}$ should be taken equal to $P_{0}\left(d_{\xi}\right)$. In other words, the noise level $\rho$ that should be used in (3.12) to decide which elements of $Q$ are consistent with the data $W$ should be taken equal to $d_{\xi}$. Yet the situation is different for these two effects. As far as $\xi_{23}$ is concerned, there exists a relation between the "true" values of $u$ and $y$ that can be expressed by a $Q$ in $Q$. The problem here is that due to the uncertainty in the data this $Q$ simply can not be observed exactly. By using $\mathcal{P}_{0}=P_{0}\left(d_{\xi}\right)$ we can be sure that $\mathcal{P}$ contains the behaviour that represents this true relation between $u$ and $y$. It is not known which behaviour this is, but if we knew, then we could use it for example as a simulation model for the system $S . \xi_{23}$ results in strict sense uncertainty, see section 2.2 .2 on page 22. For $\xi_{1}$ we have the situation that there is no $Q$ in $Q$ that can express the true relation between inputs and outputs, $\xi_{1}$ is part of this true relation. $\xi_{1}$ represents undermodelling as discussed in section 2.2.1. This undermodelling is not bounded by $d_{\Delta}$. The eventual value of $d_{\Delta}$ only makes sure that the value of $\xi$ due to undermodelling and other effects can be bounded by $d_{\xi}$. As a consequence of this, $\xi_{1}$ may have its influence on the stability of the closed loop. This is discussed in the next section.

The results of this section can be summarised as follows. If the process uncertainty set $\mathcal{P}$ uses $\mathcal{P}_{0}=P_{0}\left(d_{\xi}\right)$ and if $d_{\Delta}$ is chosen such, that $\mathcal{M}_{\text {rel }}\left(d_{\Delta}\right) \supset$ $\mathcal{P}$ then the same value for $d_{\xi}$ can be used in identification and model error bounding on the one hand and robust control design on the other hand. If this is not the case, then the effects of undermodelling, disturbances and noise as expressed by $\xi$ may exceed the bound $d_{\xi}$ by a factor two and more.
$d_{\Delta}$ alone does not suffice to characterise the uncertainty in the nominal model $\mathcal{M}_{i d}$. The value of $d_{\xi}$ also accounts for part of the uncertainty.

### 3.3.3 Closed loop influence of $\xi$

As the signal $\xi$ accounts, among other things, for a relation between $u$ and $y$ that can not be expressed by $Q, \xi$ accounts also for influences that are "in the closed loop," and may therefore have its influence on the stability of the closed loop. Nevertheless it may seem at first glance that robust control design takes, as far as stability robustness is concerned, only the influence of $\Delta$ into account.


Figure 3.7: Interconnection of $S$ and $C_{\text {stab }}$, complete $\xi$

Consider again figure 3.1, which has been redrawn in figure 3.7 in a modified form. In this figure, all of $\xi$ has been "pulled out of" $S$, leaving the block denoted " $S-\xi$." Note that, contrary to figure 3.1, some of the input/output behaviour of $S$ is now represented outside the block $S-\xi$ : the part of $\xi$ that accounts for the fact that $S \notin Q$ represents input/output behaviour of $S$, but it is extracted from the block " $S-\xi$." It now holds

$$
\xi=\left(\xi_{a}, \xi_{b}, \xi_{c}, \xi_{d}\right)^{T}
$$

and $C_{\text {stab }}$ stabilises $S$ if and only if

$$
\begin{equation*}
\left(\left\|u_{1}\right\|<\infty \wedge\|r\|<\infty \wedge\|\xi\|<\infty\right) \Longrightarrow\left(\left\|u_{2}\right\|<\infty \wedge\|y\|<\infty\right) \tag{3.21}
\end{equation*}
$$

If the controller now meets for every $M \in \mathcal{M}_{\text {rel }}$, and thus for every $M \in \mathcal{P}$, the stronger requirement

$$
\begin{equation*}
\left(\left\|u_{1}\right\| \leq d_{u 1} \wedge\|r\| \leq d_{r} \wedge\|\xi\| \leq d_{\xi}\right) \Longrightarrow\left(\|u\| \leq d_{u} \wedge\|y\|<\infty\right) \tag{3.22}
\end{equation*}
$$

then it also guarantees to stabilise the true plant $S$, provided we ensure that

$$
\begin{equation*}
\left\|u_{1}\right\| \leq d_{u 1} \wedge\|r\| \leq d_{r} \tag{3.23}
\end{equation*}
$$

The reasoning behind this is simple: if the controller ensures that $\|u\| \leq d_{u}$ for any $\xi$ such that $\|\xi\| \leq d_{\xi}$ then it will also do so far that particular $\xi$ that represents the relation between $u$ and $y$ that could not be represented by a $Q \in Q$. Ensuring that $\|u\| \leq d_{u}$ implies in turn that $\|\xi\| \leq d_{\xi}$.

It now turns out that it was correct to require for all elements in $\mathcal{P}$ to be stabilised by the controller $C_{\text {stab }}$, of which it was actually only known that it stabilises the true system and possibly not its manifestation in $Q$. However this is only shown above to be correct under the condition that $C_{s t a b}$ satisfies (3.22) and that the data set $W$ was collected under circumstances satisfying (3.23).

### 3.3.4 $d_{\Delta}$ is more detailed than $d_{\xi}$

In section 3.3 .2 situations were encountered in which a smaller value of $d_{\Delta}$ could possibly be obtained at the expense of a larger value of $d_{\xi}$. In fact, the signal $\xi$ can account for anything that $\Delta$ can account for, provided it is allowed to grow sufficiently large. It is possible as an extreme case to have $d_{\Delta}=0$ (or some other appropriate value for which it holds that $\mathcal{M}_{\text {rel }}=\mathcal{M}_{i d}$ ) while $\xi$ accounts for all the uncertainty and errors in the nominal model.

The reason why this is possible is closely related to the reason why it should be avoided: $\xi$ can account for just about anything, whereas $\Delta$ implies some structuring in the error, however rough this may be. If $\Delta$ is for example an additive error that is bounded by some induced operator norm, then it is known that the influence of $\Delta$ on the outputs $y$ is small, in some sense, if $u$ is small. Such structure is absent from a bound on $\xi$.

As a result, a robust control design faces in principle a more difficult problem if all uncertainty has been expressed in $\xi$ and not in $\Delta$. It should therefore be preferred to use $\Delta$ where possible and $\xi$ only if there is no other way, as was done in section 3.3.2.

### 3.3.5 Multiple manifestations of $S$ in $Q$

A refinement of the interpretation of the formal framework can be given by allowing multiple manifestations of $S$ in $Q$. During the discussion of the manifestation of $S$ in $Q$ it was already mentioned that this manifestation, say $Q^{\prime}$ should only be a good approximation of $S$ locally, both in time and in amplitude. As far as time was concerned, the normal life-span of the system $S$ was taken to be the time-interval on which $Q^{\prime}$ should be a good approximation of $S$.

An approach allowing a more detailed description of $S$ would be to require $Q^{\prime}$ to be only a good approximation of $S$ for an even shorter period of time. This period of time should at least contain several times the largest relevant time-constant in $S$ and this should also be contained by the time intervals on which the data sets in $W$ are taken, or one can not expect to find an accurate approximation for $Q^{\prime}$ from $W$. What manifestation of $S$ actually applies at a certain moment depends on a number of conditions, such as nominal values of the inputs to the process, external operating conditions, etc. A set of conditions
under which a certain manifestation of $S$ applies is normally called an operating point.

Based on physical reasoning one could then proceed to make sure that $W$ contains a sufficient number of data sets to determine all relevant manifestations of $S$ in $Q$, in other words $W$ should cover all operating points of $S$. The signal $\xi$ accounts then also for the effects that occur during the transition from one operating point to another. An example of this approach is given in the next chapters.

### 3.3.6 Norms in the formalism

In many places in this chapter signal norms have been used to express the size of a signal. For example, the input/output range that is to be considered relevant as far as model validity is concerned was assumed to be induced by the requirement $\|u\|<d_{u}$. The optimal $w_{2}$ in (3.6) on page 44 was defined through minimisation of a norm, the disturbances $\xi$ are assumed to be norm-bounded signals, etc. For the purposes of this formalism, norms are merely used as a measure for the size of a signal. It need not be the case that the measure that is appropriate at a certain point is also the one that should be used at another point.

For ease of exposition, little attention has been given to this issue in this chapter. In fact, up to technical details, the statements made in this chapter generalise straightforwardly to a situation in which different norms are used explicitly or in which even signal measures are used that are not norms in the strict mathematical sense.

### 3.4 Summary

A formal framework for the process of identifying nominal models and bounding the uncertainty and/or error in these models has been presented. By means of the concepts introduced in this framework it was analysed how to handle the practical situation that the combination of nominal models and bounded process uncertainties is known to provide insufficient freedom to represent the true plant exactly. It turned out that the effect of this inability to describe the true plant exactly can be handled in a way similar to the way in which disturbances and measurement noise can be handled.

Most concepts in the formal framework have been left rather open for the sake of generality.

## 4

## An algorithm for structured and unstructured error bounds

4.1 Overview of contents
4.2 Basic ideas
4.3 Model parametrisation and uncertainty description
4.4 Experimental data and prior knowledge
4.5 Outline of the algorithm
4.6 Review of systems and models
4.7 Step 1-4: obtaining a MIMO basis
4.8 Step 5: estimating auxiliary models

4.9 Step 6 \& 7: estimating the error structure<br>4.10 Step 8: translating data and prior knowledge<br>4.11 Step 9: bounding the local structured error<br>4.12 Step 10: bounding the local unstructured error<br>4.13 Step 11: combining the local results<br>4.14 Influence of the choice of the basis<br>4.15 Summary

### 4.1 Overview of contents

In this chapter we will present an algorithm for bounding model uncertainty. A few considerations are underlying the design of this algorithm. These will be discussed in section 4.2. In section 4.3 the model parametrisation and uncertainty description used in the algorithm are presented. Section 4.4 describes the input to the algorithm. This will consist of experimental data and prior knowledge about the plant that the algorithm can handle or requires. After this basis, the algorithm is given in section 4.5 , the main section of this chapter. The various steps are worked out in more detail in section 4.7 to 4.13. Some of these steps are taken from chapter 5 of Hakvoort (1994). This will be indicated where appropriate. In this chapter we follow the steps taken there closely. In the next chapter some of these steps are reconsidered or extended. This will hopefully help the reader who is already familiar with this work to focus better on the main issues instead of on technical details.

This chapter ends with a summary of the choices that were made and of the constraints under which they were made. This is intended as an aid for those who want to change some of the aspects of the proposed algorithm. Also, the original aspects of the algorithm are summarised.

This algorithm was published earlier in highly condensen form in (Ariaans et al., 1996).

Example: Throughout this chapter, several steps in the algorithm will be illustrated by means of a simple SISO example. This example will be shown in the same way as this paragraph.

### 4.2 Basic ideas

The combination of model and uncertainty bound that is aimed for, should be suitable for current robust control design techniques. These techniques should include $\mathcal{H}_{\infty}$ control design and/or $\mu$-synthesis, as these seem to have the widest practical acceptance. To our knowledge, current solution methods for the $\mathcal{H}_{\infty}-$ control design problems use linear, time-invariant nominal models. This does not mean that non-linear processes can not be handled at all. As long as the non-linearity can be represented as an $\mathcal{H}_{\infty}$-bounded model "uncertainty" these techniques can still be employed. Although the model uncertainty itself does not have to be linear, we will restrict attention to model uncertainties that are linear time-invariant operators. This means that only linear time-invariant processes will fall in the model uncertainty description. Fortunately, practical experience has shown, that for many processes operating in a fixed operating point, a linear time-invariant model is quite adequate for the purpose of control design to describe the dynamic behaviour of that process. We therefore restrict attention to a combination of nominal model and process uncertainty description yielding a set of linear, time-invariant processes.

For causal, stable, linear, time-invariant processes, the generalised orthonormal basis functions described in (Heuberger, 1991) and later successfully applied in (De Vries, 1994) and (Hakvoort, 1994) seem to yield an attractive model parametrisation in the light of uncertainty bounding. Roughly speaking, generalised orthonormal basis functions enable us to incorporate a rough model of the process dynamics into the model parametrisation. This rough model is called the basis generating system. The closer this model is to the true system, the fewer model parameters are needed to give an accurate description of the process. Moreover, bounds on the undermodelling can be given under weak conditions, which also become smaller if the basis generating system is a better approximation of the true process. In principle, these parametrisations are only suitable for stable processes, but in (De Vries, 1994, chapter 6) a way to circumvent this problem due to (Schrama, 1992) is discussed. Based on these
considerations, we would like to use this model parametrisation, together with its attractive properties, in our algorithm as well.

Having looked briefly at how the process uncertainty is to be described, with still many open questions, we should now ask ourselves what the process uncertainty is meant to cover.

In general, a whole set of process transfers is consistent with the data we gathered about the process in an identification experiment and the prior knowledge and assumptions we may have. This corresponds to the process uncertainty set $\mathcal{P}$ of chapter 3 . The pair of nominal model and process uncertainty bound is one way to cover all elements in $\mathcal{P}$. We do not assume, that our nominal model or any other model in the model set are a member of $\mathcal{P}$. This idea is a common starting point for current uncertainty bounding techniques. It is found for example in Helmicki et al. (1991), Goodwin et al. (1992), Van den Boom (1993), Mäkilä et al. (1994), De Vries (1994), Hakvoort (1994), Zhou and Kimura (1994) and many others.

For application in robust control design, we want our algorithm to go one step further. Not only should our uncertainty description reflect the fact that the nominal model is only an approximate description of a system that we can not pin-point exactly. It should also reflect, that if the process has moved from one operating point to another, a different linear time-invariant model may have become the most appropriate one to describe the true process. The process uncertainty set will also have changed in this situation: transfers that were consistent with the dynamical behaviour of the plant in one operating point, may not be consistent with the dynamical behaviour in another operating point and vice versa. Other, less intentional causes for a shift in the dynamic behaviour of the plant are conceivable too. Different characteristics of raw material, changing environmental conditions, aging catalysts etc. may change the process dynamics as well.

The description of reality would be better if we went even one step further, and described not only the behaviour in various operating points, but also the way in which transitions between different operating points are accomplished. This would lead us into the field of non-linear and/or time-varying models. For reasons mentioned above, we do not want to enter this complex field. This means that some dynamic behaviour of the plant, namely the non-linear and changing dynamics, will be missing from our process uncertainty set. However, we feel that recognising the effects that non-linearities have in operating points while omitting the effects between operating points is to be preferred over ignoring the effect of non-linearities both in and between operating points.

The concept of varying operating points also applies to a situation in which the plant to be described is known to be one of a set of similar but not identical plants and in which it is unknown which of the plants will actually be dealt with. If a product is produced in large series, small disturbances during the production process will result in, hopefully small, differences between the final
products. If a controller is part of the product, the same design will face slightly different plants. Each realisation of the product can be interpreted as a different operating point. In this case, the controller will not encounter transitions between these operating points.

The list of situations that can be interpreted as a plant operating in different operating points can be extended further. One may think for example of standard controllers for slightly different production lines or of "reliable stabilisation" in environments were plants are affected by failures in actuators and sensors. In the latter example different operating points correspond to different sets of available actuators and sensors. Further extension is left to the reader.

As the name implies, time-invariant processes have the same behaviour at any time instant. There seems to be a fundamental contradiction in using multiple linear time-invariant models and process uncertainty sets for a single plant. However, this is not a real contradiction. It is not the true process that we take time-invariant. It is only the approximate descriptions of it that we assume to be time-invariant.

Successful application of this idea puts stronger requirements on our experimental data and prior knowledge. Obviously, if we want our process uncertainty set to cover several operating points, all of these operating points have to be present in the data. This implies that we have to do experiments in each of these operating points. It also means that we have to adjust our behaviour in case of multiple data sets. "Traditionally," the overall process uncertainty set, resulting from the combination of data sets, could be taken equal to the intersection of the local uncertainty sets for each data set: if the true process is known to be in both uncertainty sets then it is known to be in the intersection of both sets. In our situation this is still valid, provided the data sets belong to the same operating point. If the data sets belong to different operating points, we should take the union of the uncertainty sets. A discussion about how many operating points to use and tools to guide this decision is deferred to section 5.6.2.

Taking the union of local uncertainty sets instead of the intersection is not a great help in our aim to reduce uncertainty bounds. Fortunately, the concept of different operating points also enables us to employ a technique that gives a tighter uncertainty description. Especially in the case of MTMO processes, we can expect the changes in process transfer occurring between operating points to be highly structured. There is only a limited number of physical parameters that change if we move between operating points and these changes may even depend on each other. (With parameter is meant here a physical quantity causing the change in behaviour, not the parameters of the mathematical description that change as a result of this.) This suggests that there is some dependency between the change of the individual model parameters. If we manage to find this dependency and separate this effect from the other contributions to process uncertainty, we can split the uncertainty in a structured
part and an unstructured part. The structured part accounts for operating point changes, the unstructured part for all the effects that were not accounted for in the structured part. The structured part hopefully covers the dominating aspects of the process uncertainty and is easy to use in $\mu$-synthesis. The unstructured part is much less detailed, but hopefully also much smaller.

The algorithm tries to accomplish the separation in structured and unstructured parts by estimating auxiliary nominal models for each operating point. Using a technique called principal component analysis we try to find the correlation between the changes in model parameters and let these dictate the structured parts. We will discuss the implications this has for the allowable model parametrisations later. A brief introduction to principal component analysis is given in section 4.9 . In section 5.7 we will discuss other ways to determine the error structure.

Every new algorithm that tries to solve a problem in the field of model error bounding runs the risk that it actually does not solve the problem, but only shifts it to the problem of obtaining the prior "knowledge." To avoid this, the algorithm should provide sensible guesses for every element of prior knowledge that it requires. If this is not possible without extra information, practical ways should be available to obtain that information.

Finally the algorithm should indicate as far as possible what determined the final error bounds. This information is especially important if some "educated guesses" were made for the prior knowledge: elements of prior knowledge that really determine the error bounds should be investigated critically whether they were realistic and not over-optimistic indeed. On the other hand, preliminary cautious choices could be considered for refinement if they appear to be nonrestrictive at all.

### 4.3 Model parametrisation and uncertainty description

### 4.3.1 Generalised orthonormal basis functions

The standard basis for $h_{2}$ is $\{\delta(t), \delta(t-1), \delta(t-2), \ldots\}$ with the corresponding basis for $\mathcal{H}_{2}$ given by $\left\{1, z^{-1}, z^{-2}, \ldots\right\}$ If the time-domain basis is denoted $\left\{b_{k}(t)\right\}_{k=0}^{\infty}$, where in this case $b_{k}(t)=\delta(t-k)$, then any element $g(t)$ of $h_{2}$ can be written

$$
\begin{equation*}
g(t)=\sum_{k=0}^{\infty} \theta_{k} b_{k}(t) \tag{4:1}
\end{equation*}
$$

where $\theta_{k}$ are the parameters of $g(t)$ expressed in the basis $\left\{b_{k}(t)\right\}$. For this particular choice of basis $\left\{b_{k}\right\}$ it holds $\theta_{k}=g(k)$.

An obvious model parametrisation is obtained by truncating the infinite summation in (4.1) to a finite one. For this particular set of $b_{k}(t)$, this corresponds to finite impulse response (FIR) models. An important advantage of this parametrisation is, that it is linear in the parameters $\theta_{k}$. The most important disadvantage is, that to approximate a system with large time constants (compared to the sampling time), a large number of parameters is required.

This disadvantage can be alleviated by using a different set of basis functions $\left\{b_{k}(t)\right\}_{k=0}^{\infty}$. It is for example well known, that a system with a single dominant pole can be approximated fairly well by a limited number of Laguerre functions. Systems with a dominant pair of complex conjugate poles can be approximated using Kautz models. A unifying framework as well as a generalisation for Laguerre and Kautz models is given by so called "system based orthonormal basis functions," described extensively in Heuberger (1991). In appendix B a more or less self-contained, though incomplete, introduction to these functions is given.

At this point we suffice with the following imprecise interpretation: given a basis generating system $G_{b}(\zeta) \in \mathcal{H}_{2}^{q \times 1}$, a mapping

$$
\mathcal{B}: \mathcal{H}_{2}^{q \times 1} \rightarrow 2^{\mathcal{H}_{2}^{1 \times 1}} \quad G_{b}(\zeta) \mapsto\left\{B_{0}(\zeta), B_{1}(\zeta), \ldots\right\}
$$

can be formulated such that the set $\mathcal{B}\left(G_{b}(\zeta)\right)$ is an orthonormal basis for $\mathcal{H}_{2}$. (Note that we may use a SIMO system to generate a set of SISO basis functions. The construction of $B_{0}(\zeta), \ldots$ from $G_{b}$ will be treated in more detail in appendix B.) Any (stable) system $G(\zeta) \in \mathcal{H}_{2}^{1 \times 1}$ can therefore be expressed as

$$
G(\zeta)=\sum_{k=0}^{\infty} \theta_{k} B_{k}(\zeta)
$$

Moreover, there exist $M \in \mathbb{R}_{+}$and $\rho \in[0,1)$ such that

$$
\begin{equation*}
\forall k \in \mathbb{N} \quad\left|\theta_{k}\right|<M \rho^{k} \tag{4.2}
\end{equation*}
$$

and $\rho$ is smaller if the poles of the basis generating system $G_{b}$ are chosen closer to the poles of the system $G$. This implies that from a certain value of $k$ on, the values of $M \rho^{k}$ will be smaller, even if the value of $M$ increases by the change in basis generating system.

In the remainder of this chapter we will also need a property and a conjecture. The property will be proven in appendix B. In this appendix an argumentation will be given as well for the conjecture. Because both are used heavily in the rest of this chapter, they are restated here for convenience. They will be used to bound the effects of undermodelling, although that is probably not evident at this point yet.

Property 4.1 Let $\left\{B_{0}(\zeta), B_{1}(\zeta), \ldots\right\}$ be a basis generated by a finite dimensional system $G_{b}(\zeta) \in \mathcal{H}_{2}^{q \times 1}$. Let $n$ denote the McMillan degree of $G_{b}(\zeta)$,
then

$$
\forall k \in \boldsymbol{n}, \ell \in \mathbb{N}, \omega^{\prime} \in \mathbb{R} \quad\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right|=\left|B_{k+n \ell}\left(e^{j \omega^{\prime}}\right)\right|
$$

Note that $B_{0}(z)$ is intentionally omitted from property 4.1. This property is based on the fact that through the construction of the basis functions, $B_{k}(\zeta)$ and $B_{k+n \ell}(\zeta)$ differ only by an integer power of an all pass function.

Conjecture 4.2 Let $\left\{B_{0}(\zeta), B_{1}(\zeta), \ldots\right\}$ and $n$ be as in property 4.1.

$$
\forall p \geq 0, k \in \mathbb{Z}_{+}, \exists c_{1}, c_{2} \in \mathbb{R} \quad \sum_{t=0}^{\infty} t^{p}\left|b_{k}(t)\right|<c_{1}+c_{2}\left\lfloor\frac{k-1}{n}\right\rfloor^{p+1 / 2}
$$

### 4.3.2 Model parametrisation

In the previous section a basis for $\mathcal{H}_{2}^{1 \times 1}$ was discussed. As we will be dealing with MIMO systems, this concept needs to be extended. We take a very pragmatic approach: for a system with $p$ inputs and $q$ outputs, a basis for the transfer from each separate input to each separate output is generated. To formalise this, we introduce some notation and terminology:

Notation: An entry of the transfer function of a MIMO system with $p$ inputs and $q$ outputs will be referred to as a subtransfer of that system.

The set

$$
\mathcal{S}:=q \times p
$$

is the set of subtransfer indices. Elements of $S$ will in general be denoted $\sigma$. The matrices $E^{\sigma} \in \mathbb{R}^{q \times p}, \quad \sigma \in \mathcal{S}$ are defined as follows:

$$
\left[E^{\sigma}\right]_{i j}:= \begin{cases}1 & \text { if }(i, j)=\sigma \\ 0 & \text { if }(i, j) \neq \sigma\end{cases}
$$

A basis for $\mathcal{H}_{2}^{1 \times 1}$ will be called a SISO basis. A collection of $p q$ SISO bases constitutes a MIMO basis for $\mathcal{H}_{2}^{q \times p}$ given by

$$
\left\{B_{0}^{\sigma}(\zeta), B_{1}^{\sigma}(\zeta), \ldots\right\}, \quad \sigma \in \mathbb{S}
$$

This means, that any system $G \in \mathcal{H}_{2}^{q \times p}$ can be written

$$
G(\zeta)=\sum_{\sigma \in S} \sum_{k=0}^{\infty} \theta_{k}^{\sigma} E^{\sigma} B_{k}^{\sigma}(\zeta)
$$

where $\left\{B_{0}^{\sigma}(\zeta), B_{1}^{\sigma}(\zeta), \ldots\right\}_{\sigma \in \mathcal{S}}$ denotes the MIMO basis. For fixed $\sigma \in \mathcal{S}$ the sequence $\left\{B_{0}^{\sigma}(\zeta), B_{1}^{\sigma}(\zeta), \ldots\right\}$ denotes a SISO basis for $\mathcal{H}_{2}$ with reference to the $\sigma$-entry of $G$.

The model parametrisation used in our algorithm is now simply obtained by taking only a finite number of basis functions. Given the basis

$$
\left\{B_{0}^{\sigma}(\zeta), B_{1}^{\sigma}(\zeta), \ldots\right\}_{\sigma \in S}
$$

the model set therefore becomes

$$
\begin{equation*}
\mathcal{M}_{\text {set }}=\left\{G(\zeta) \in \mathcal{H}_{2}^{q \times p} \mid G(\zeta)=\sum_{\sigma \in S} \sum_{k=0}^{\bar{k}^{\sigma}} \theta_{k}^{\sigma} E^{\sigma} B_{k}^{\sigma}(\zeta)\right\} \tag{4.3}
\end{equation*}
$$

where $\theta_{k}^{\sigma} \in \mathbb{R}$ for $\sigma \in S, k=0, \ldots, \bar{k}^{\sigma}$ and $\left\{\bar{k}^{\sigma}\right\}_{\sigma \in S}$ is a set of properly chosen integer constants specifying the model complexity for each subtransfer. In this context, the $\theta_{k}^{\sigma}$ are called generalised FIR parameters.

We will not discuss the general problem of choosing the right model complexity. For notational convenience, we will drop the superscript $\sigma$ from $\bar{k}^{\sigma}$ wherever possible. The algorithm does not require that the $\bar{k}$ are the same for all subtransfers, however. This can be especially convenient if, say, we want low model complexity for all subtransfers except one.

The $\theta_{k}^{\sigma}$ involved in (4.3) are collected in a parameter vector $\theta \in \mathbb{R}^{c}$, where obviously $c=\sum_{\sigma \in \delta}\left(\bar{k}^{\sigma}+1\right)$. Given the values of $\bar{k}$ and the MIMO basis $\left\{B_{k}^{\sigma}(\zeta), k \in \mathbb{N}, \sigma \in \mathcal{S}\right\}$, equation (4.3) implies a bijection between members of $\mathcal{M}_{\text {set }}$ and parameter vectors $\boldsymbol{\theta} \in \mathbb{R}^{c}$. In general we will assume the MIMO basis and the values of $\bar{k}$ implicitly. These are fixed in the early stages of the algorithm. We will therefore denote the member of $\mathcal{M}_{\text {set }}$ corresponding to some $\theta \in \mathbb{R}^{c}$ by $G_{\boldsymbol{\theta}}$, although it should actually be denoted $G_{\boldsymbol{\theta}}\left(\bar{k}^{\sigma}, B_{k}^{\sigma} ; \sigma \in S\right)$.

Remark 4.1 Even if minimal state-space realisations are used for the $B_{k}^{\sigma}$, a MIMO state-space model obtained by simply combining the models for the subtransfers as indicated by (4.3) will in general not be minimal. Especially if the same SISO basis is used for all subtransfers, a considerable reduction in state-dimension can be achieved. Reducing the state-dimension afterwards does not influence the validity of the (reduced) results.

### 4.3.3 Uncertainty description

The uncertainty description is given by

$$
\begin{equation*}
\hat{G}+\Delta+\sum_{i=1}^{n_{s}} \delta_{i} A_{i} \tag{4.4}
\end{equation*}
$$

where

$$
\begin{gathered}
\hat{G}, \Delta, A_{i} \in \mathcal{H}_{2}^{q \times p} \\
\underline{d}_{i} \leq \delta_{i} \leq \bar{d}_{i} \\
\|\Delta\| \leq d_{\Delta} \\
d_{\Delta}, \underline{d}_{i}, \delta_{i}, \bar{d}_{i} \in \mathbb{R} \\
i \in n_{s}
\end{gathered}
$$

$\hat{G}$ represents the overall nominal model. The algorithm which we present can work with an a priori given nominal model, or it can suggest a nominal model along the way. $\delta_{i} A_{i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$ are the structured error components. The transfers $A_{i}$ are usually determined by the algorithm, however, they can be specified by the user as well. These error components are uncertain, because the gain factors $\delta_{i}$ can vary within the intervals $\left[d_{i}, \bar{d}_{i}\right]$. The transfer $\Delta$ will be called the unstructured error. It is bounded in some measure, indicated by $\|\Delta\|$. Although $\|\Delta\|$ suggests that the measure used to bound $\Delta$ is a norm, this will not always be the case. The algorithm can bound the $\mathcal{H}_{\infty}$ norm for every separate entry of $\Delta$. It is also possible to bound the impulse response of $\Delta$ : for each time instant and for every separate entry of $\Delta$, an interval will be determined to which the impulse response is restricted. This can also be done for the impulse response after weighting by a stable weighting filter and for the step response. These time domain bounds on $\Delta$ leave the possibility open to link the results of the algorithm with time domain based control strategies optimising over a finite horizon, such as the class of model based predictive controllers. Given the wide-spread application of these controllers in process industry, this is of major practical importance.

Remark 4.2 The nominal model $\hat{G}$ need not be expressed in the model parametrisation of section 4.3.2. If the algorithm suggests a $\hat{G}$, it will be a model taken from $\mathcal{M}_{\text {set }}$ in (4.3). If the user specifies this model instead, he is free to specify any model in $\mathcal{H}_{2}^{q \times p}$.

A fundamental choice has been made in this uncertainty description: the process signals have been divided into inputs and outputs. After chapter 3, where for that matter this distinction was made too, some justification of this choice may be in order. The first reason for this is that current techniques for robust control design use this separation. As the model and the uncertainty bounds are derived for application in robust control design, the separation into inputs and outputs is going to be made anyway, so we might as well make it now. Moreover, if signals can be separated into actuators and sensors, this leads to a natural separation into inputs and outputs. Finally it was argued in section 4.2 that a model parametrisation in terms of system based orthonormal basis functions was desired as such parametrisation enables us to obtain an
accurate description of a system using few parameters and to derive explicit bounds on the undermodelling. These benefits have been shown for systems interpreted as (linear, time-invariant) input/output operators.

### 4.4 Experimental data and prior knowledge

The algorithm will need several "inputs:" information that it can not or need not generate itself. Required information will be marked with ' $\varphi$ ' at the start of the paragraph, optional information by ' 0 '.

The algorithm will involve identifying auxiliary models in every operating point of the process. We therefore require at least one experimental data set for each operating point. Multiple data sets in one operating point are possible.

Notation: Let $n_{w}$ denote the number of operating points. $\mathcal{W}$ shall be used to denote the set of operating points. Elements of $\mathcal{W}$ are by convention denoted $w$. With each $w \in \mathcal{W}$ is associated a finite set $\mathcal{D}(w)$ of data set indices for $w$ : each $d \in \mathcal{D}(w)$ specifies a data set taken in the operating point $w$, but it is not the data set itself. A data set is typically a record of input and output measurements taken from the process during an experiment. The elements of $\mathcal{D}(w)$ are used to enumerate these experiments.

$$
\mathcal{D}:=\bigcup_{w \in \mathcal{W}} \mathcal{D}(w)
$$

$\mathcal{D}$ is the index set of all data sets in all operating points. Elements of $\mathcal{D}$ and $\mathcal{D}($.$) are normally denoted d . d$ is a data set index then.
$n_{d}(w)$ is the number of elements in $\mathcal{D}(w), w \in \mathcal{W}$, while $n_{d}$ is the total number of elements in $\mathcal{D}$.

The required prior information can now be stated more precisely:

- For each data set $d \in \mathcal{D}$, it is assumed that measurements $u^{d}(t)$ have been taken of the process inputs and $y^{d}(t)$ of the process outputs. The measurements have been taken on the time interval $\mathbb{T}^{d} \subset \mathscr{Z}$, so $u^{d}$ and $y^{d}$ are defined on $\mathbb{T}^{d}$. Because this information will be used to identify time-invariant models, it is assumed without loss of generality that each $\mathbb{T}^{d}$ starts at $t=0$. The length of the experiment interval shall be denoted $T^{d}$, so $\mathbb{T}^{d}=\left\{0, \ldots, T^{d}-1\right\}$.

The data sets will be used to identify models in each operating point. This implies that the data sets have to be suitable for this purpose: the sampling frequency should have been chosen sufficiently high to capture the fastest process behaviour that should still be modelled and the length of the data sets
should be sufficiently long to identify reliably the largest relevant time constants of the process. Also the input signal should be sufficiently rich to be sure that all relevant process dynamics are indeed observable from the output. The amplitude of the input signal should be small enough not to excite too much of the non-linearities present in the true plant, yet be large enough to have a sufficiently good signal to noise ratio at the outputs of the process.

In order to fulfill these conditions, extra prior knowledge about the plant is required. Also proper anti-aliasing measures need to be taken to make sure that neglected high-frequency dynamics do not influence the results in the frequency range of interest. As the algorithm is not so much concerned with the identification of models as with the bounding and structuring of model uncertainty, this knowledge is not listed explicitly here.

- For some data sets $d \in \mathcal{D}$, an instrumental variable $v^{d}(t), t \in \mathbb{T}^{d}$ may be available. Whether this is needed or not depends on the identification methods that will be used in the algorithm and on the kind of experimental data that is available, closed loop or open loop.
For the identification method that will be discussed as an example, the instrumental variables should be correlated with the inputs of the system, but not with the disturbances acting on the system. In an open loop situation, this means that the inputs of the system can be used as instrumental variables if one so desires. Therefore instrumental variables are considered optional in open loop: if they are not available the inputs will be used as instruments. In closed loop separate signals, for example the external references of the plant, are required.
For the example identification method, the instrumental variable should contain $p$ signals, i.e. equal to the number of inputs.
- The algorithm requires a MIMO basis for $\mathcal{H}_{2}^{q \times p}$ as described in the previous section. For optimal results, the dynamics of this basis should resemble the dynamics of the true process, see appendix B. Therefore, either a MIMO basis has to be available a priori, or knowledge about the dominant dynamics of the process, so that a basis can be generated as explained in appendix $B$.
- A nominal model $\hat{G}(\zeta)$ may be available for the process. Any stable, linear time-invariant model can be used. It need not be expressed in the model parametrisation used by the algorithm.

Given the availability of a MIMO basis, the (linear manifestation of the) true system $G_{t r}^{w}$ can be written in each operating point $w \in \mathcal{W}$ as

$$
G_{t r}^{w}(\zeta)=\sum_{\sigma \in S} \sum_{k=0}^{\infty}\left(\theta_{t r}^{w}\right)_{k}^{\sigma} E^{\sigma} B_{k}^{\sigma}(\zeta)
$$

- For each operating point, and for each subtransfer $\sigma \in \mathcal{S}$, values $M^{\sigma} \in \mathbb{R}_{+}$, $\rho^{\sigma} \in[0,1), \theta_{\max }^{\sigma} \in \mathbb{R}_{+}$and $\left(k^{*}\right)^{\sigma} \in \mathbb{N}$ are known, such that

$$
\left|\left(\theta_{t r}\right)_{k}^{\sigma}\right| \leq \ddot{\theta}_{k}^{\sigma}:= \begin{cases}\theta_{\max }^{\sigma} & k<\left(k^{*}\right)^{\sigma}  \tag{4.5}\\ M^{\sigma}\left(\rho^{\sigma}\right)^{k} & k \geq\left(k^{*}\right)^{\sigma}\end{cases}
$$

Essentially, (4.5) means, that the tail of the parameter sequence $\left\{\left(\theta_{t r}\right)_{k}^{\sigma}\right\}_{k=0}^{\infty}$ decays exponentially. This implies, that the system has to be stable! This implication is not trivial, it is a property of system based orthonormal basis functions.
(4.5) goes further than (4.2) in that not only the existence of $M$ and $\rho$ is stated, but values for $M$ and $\rho$ are assumed to be known. To avoid too large bounds for small values of $k$, there is the option to switch to another bound for small values by means of $\left(k^{*}\right)^{\sigma}$ and $\theta_{\max }^{\sigma}$. In fact, it would be possible to make $\theta_{\max }^{\sigma}$ dependent on $k$ for $k<\left(k^{*}\right)^{\sigma}$. In the actual implementation this has been carried out. However, this possibility will not be pursued here for notational convenience.

In section 4.8 .3 it will be discussed how one may estimate this prior information.

- For each operating point $w \in \mathcal{W}$, the data sets $d \in \mathcal{D}(w)$ are assumed to be affected by additive noise or disturbances $\epsilon^{d}(t)$ at the output only. Therefore, the following relation holds

$$
\begin{equation*}
\forall t \in \mathbb{T}^{d} \quad y^{d}(t)=G_{t r}^{w}(\zeta) u^{d}(t)+\epsilon^{d}(t) \tag{4.6}
\end{equation*}
$$

For ease of reference, $\epsilon^{d}(t)$ will be called output noise, although in chapter 3 it has been argued that $\epsilon^{d}(t)$ should be interpreted broader than just noise effects, see the discussion of $\xi$ in chapter 3 .

- For each data set $d \in \mathcal{D}$, a signal $\bar{e}^{d}: \mathbb{T}^{d} \rightarrow \mathbb{R}_{+}^{q}$ is available, such that

$$
\begin{equation*}
\forall t \in \mathbb{T}^{d} \quad\left|\epsilon^{d}(t)\right| \leq \bar{e}^{d}(t) \tag{4.7}
\end{equation*}
$$

For $q>1$, i.e. for multiple output signals, the absolute value and the inequality should be taken element-wise.

- For each data set $d \in \mathcal{D}$, a vector $\bar{u}^{d} \in \mathbb{R}_{+}^{p}$ is available, such that

$$
\begin{equation*}
\forall t<0 \quad\left|u^{d}(t)\right| \leq \bar{u}^{d} \tag{4.8}
\end{equation*}
$$

This information will be used later to bound the influence of unknown initial conditions.

At this point it is worthwhile to recall the discussion about the bound $\bar{u}$ on the input signals in chapter 3, denoted $d_{u}$ there: the linear manifestation
of the process in an operating point depends not only on the operating point itself but may also depend on the level of excitation in that operating point. Therefore the level of excitation in the data sets should be similar to the level of excitation that can be expected in the practical situation for which the nominal model and uncertainty bounds are derived. Due to the heuristic nature of this argument it is not formally listed here as required prior information.

- For some data sets $d \in \mathcal{D}$, instrumental variables $r_{\ell}^{d}(t): \mathbb{T}^{d} \rightarrow \mathbb{R}$ and constants $\bar{c}_{\ell}^{d} \in \mathbb{R}_{+}^{q}, \ell=n_{r}^{d}, n_{r}^{d} \in \mathbb{N}$ may be available, such that the crosscovariance between $\epsilon^{d}(t)$ and $r_{\ell}^{d}(t)$ is bounded by known constants $\bar{c}_{\ell}^{d}$ :

$$
\begin{equation*}
\frac{1}{\sqrt{T^{d}}}\left|\sum_{t \in \mathbb{T}^{d}} \epsilon^{d}(t) r_{\ell}^{d}(t)\right| \leq \bar{c}_{\ell}^{d} \tag{4.9}
\end{equation*}
$$

Here also, the absolute value and inequality should be taken element-wise. In order to limit the effect of initial conditions, the summation may be started from some $0<t_{s}<T^{d}$. The value of $\bar{c}_{\ell}^{d}$ should be updated accordingly. It is discussed in section 4.8 .3 how $\bar{c}_{\ell}^{d}$ can be estimated.

The instrumental variables $v^{d}(t)$ on page 81 and $r_{\ell}^{d}(t)$ above are used for different purposes. Each signal $r_{\ell}^{d}(t)$ is scalar valued, but as many $r_{\ell}^{d}(t)$ can be used as one wishes. The $v^{d}(t)$ on the other hand have the same dimension as the process input and only one $v^{d}(t)$ can be used. Provided one can obtain the corresponding $\bar{c}_{\ell}^{d}$, the components of $v^{d}(t)$ can be used as (some of) the $r_{\ell}^{d}(t)$.

Example: The simulation example discussed throughout this chapter will use a "true" process having two linear time-invariant manifestations. The two operating points corresponding to these manifestations are denoted $a$ and $b$ respectively, so that in this example

$$
\mathcal{W}=\{a, b\}
$$

The transfer in operating point $a$ is given by

$$
G_{t r}^{a}(z)=\frac{1}{z-0.97}
$$

and that in operating point $b$ by

$$
G_{t r}^{b}(z)=1.5 G_{t r}^{a}(z)=\frac{1.5}{z-0.97}
$$

For both operating points, one data set was generated. We can therefore identify the set of data set indices $\mathcal{D}$ with $\mathcal{W}$ and define $\mathcal{D}(a)$ and $\mathcal{D}(b)$ correspondingly:

$$
\mathcal{D}=\{a, b\}, \quad \mathcal{D}(a)=\{a\}, \quad \mathcal{D}(b)=\{b\}
$$

The inputs in both operating points were zero mean, normally distributed white noise sequences. Input sequences of 1500 samples were generated. These were applied to the process, but only the last 1000 samples obtained in this way were recorded in the data set. Prior to these excitations of the process, the process was in steady state. In this way a non-zero but realistically sized transient can be expected in the data sets. As both data sets contain 1000 samples it holds

$$
T^{a}=T^{b}=1000, \quad T^{a}=\mathbb{T}^{b}=\{0,1, \ldots, 999\}
$$

Normally distributed, zero-mean white noise was added to the output signals in the data sets. The signal to noise ratio in data set a was taken equal to 40 dB . The same absolute noise level was used in data set $b$. The noise was not correlated with the inputs.

No separate instrumental variable $v^{d}(t), d \in \mathcal{D}$ was used for the purpose of identification. The instruments were taken equal to the input sequences:

$$
v^{a}(t)=u^{a}(t) \quad v^{b}(t)=u^{b}(t)
$$

For the cross-covariance bounds no separate instrumental variables were used either. Here too the inputs were used instrumental variables. The number of instrumental variables was therefore equal to 1 in both data sets,

$$
n_{r}^{a}=1, \quad n_{r}^{b}=1
$$

and

$$
r_{1}^{a}(t)=u^{a}(t), \quad r_{1}^{b}(t)=u^{b}(t)
$$

In this simulation example, values for the parameter bounds $\bar{\theta}_{k}, k \in \mathbb{N}$, the input bounds $\bar{u}^{d}, d \in\{a, b\}$, the noise bounds $\bar{e}^{d}, d \in\{a, b\}$ and the crosscovariance bounds $c_{1}^{d}, d \in\{a, b\}$ can be determined easily. This is not done here, as this approach is not realistic in practical cases. These bounds will be estimated later on in this chapter from information that one can assume to be available also in a practical situation.

For completeness, the input and output signals in both data sets have been plotted in figure 4.1.
(Continued on page 90.)

### 4.5 Outline of the algorithm

The steps involved in the "basic" algorithm are now presented. This section is not intended as a self-contained description of the algorithm. It should be considered a frame-work in which the sections 4.7 to 4.13 fit. In these sections, each step is described in more detail.

In the following, "a priori" means "before the algorithm starts." A priori information therefore includes information that was obtained in an identification step carried out before the algorithm.


Figure 4.1: Input and output signals in both data sets

STEP 1 If a MIMO basis is available a priori goto step 5.
STEP 2 If we have a nominal model $\hat{G}$, goto step 4.
STEP 3 Identify one nominal model $\hat{G}$ (i.e. for all operating points) to generate a MIMO basis.

STEP 4 Generate a MIMO basis $\left.\left\{B_{k}^{\sigma}\right\}_{\sigma \in s}\right\}$ for the model parametrisation.

STEP 5 For each operating point $w \in \mathcal{W}$, estimate an auxiliary model, represented by a parameter vector $\boldsymbol{\theta}^{w}$.

STEP 6 Construct a central model from the $\theta^{w}, w \in \mathcal{W}$. Denote the corresponding parameter vector $\boldsymbol{\theta}^{c}$. If the structured error components $A_{i}$ are specified a priori, continue with step 8.

Step 7 Define $\Delta \boldsymbol{\theta}^{w}:=\boldsymbol{\theta}^{w}-\boldsymbol{\theta}^{c}, w \in \mathcal{W}$. Analyse the $\Delta \boldsymbol{\theta}^{w}$ using principal component analysis. This will yield $n_{s}$ dominating directions in the parameter space, denoted $\boldsymbol{\theta}_{A, i}, i \in \boldsymbol{n}_{s}$. These are the parameters of the structured error components $A_{i}$, for which bounds will be derived in the next steps.

Step 8 For each operating point $w \in \mathcal{W}$, translate the experimental data and prior knowledge to a set of linear constraints on the model parameters. The set of model parameters satisfying these constraints will be denoted $\mathcal{L}_{\boldsymbol{\theta}}^{w}$. This set constitutes a polytope in parameter space.

STEP 9 For each operating point $w \in \mathcal{W}$, bound the structured errors by solving a linear programming problem. The parameters in this LP problem are restricted to $\mathcal{L}_{\boldsymbol{\theta}}^{w}$. This yields bounds for $\delta_{i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$, locally in each operating point.

STEP 10 For each operating point, bound the unstructured error, taking into account the structured error components.

Step 11 Combine the results for all operating points to bounds on $\delta_{i}, i \in n_{s}$, and $\Delta$ that are valid globally, i.e. for all operating points.

A more detailed summary of the algorithm is given on pages 146 and 147. This summary is also printed on a separate sheet coming with the thesis, so that it can be kept alongside the text. The algorithm is divided into mayor steps, which are further split into a priori information for the step, marked by a '<', a posteriori information for the step, marked by a ' $>$ ', and possibly sub-steps. The numbers at the right edge of the page refer to the page at which the relevant concept is introduced.

The main steps in the algorithm as listed above can be recognised as follows in the summary. The steps "Do experiments" and "Gather prior information" are actually not part of the algorithm, they precede step 1 of the algorithm. "Choose basis functions and model orders" covers steps 1 to 4 . The auxiliary models of step 5 are estimated in "Estimate auxiliary models" and the analysis of the error structure of steps 6 and 7 is carried out under the heading "Estimate error structure," Step 8, the translation of the prior knowledge to linear constraints on the model parameters is referred to on page 146 as "Construct set of linear constraints". This step is further divided into the
steps "Extend noise bound" and "Extend cross-covariance bound." In chapter 5 more steps will be added under this heading. Steps 9 and 10 are both summarised under the heading "Bound errors locally." Step 9 corresponds to the sub-heading "Bound structured errors," whereas step 10 corresponds to both the sub-headings "Bound unstructured errors in frequency domain" and "Bound unstructured errors in time domain." The sub-step "Split parameter uncertainty in structured and unstructured parts" is listed as a separate step on page 147. Actually this is not a separate step but a step that is performed many times as part of the steps bounding the structured (step 9) and unstructured (step 10) error. Step 11 of the algorithm is finally called "Combine local results" on page 147.

At the beginning of a section in which a step of the algorithm is presented in detail, the summary on pages 146 and 147 is repeated in condensed form and in smaller print. In these condensed summaries, only the step being discussed is split further into required information, sub-steps and generated information. The other steps will only be indicated by their heading. Moreover, these other steps will be shown in lighter print.

### 4.6 Review of systems and models

At this point it may be worthwhile to reconsider the various systems and models that have occurred so far,

First of all there is the true system $G_{t r} . G_{t r}$ can be non-linear, infinite dimensional, time-varying etc. However, it is assumed that in an operating point $w \in \mathcal{W}, G_{t r}$ can be approximated by its linear manifestation $G_{i r}^{w}$, where $G_{t r}^{w}$ is linear and time-invariant, possibly infinite dimensional. Note that it would be more accurate to refer to $G_{i r}^{w}$ as the linear and time-invariant manifestation of $G$, but for brevity it is simply called the linear manifestation of $G_{t r}$. The approximation of $G_{t r}$ by $G_{t r}^{w}$ should be interpreted as explained in chapter 3, in particular in section 3.3.5. Note that the behaviour of the true system $G_{t r}$ that is not represented by $G_{t r}^{w}$ ends up in the noise sequence $\epsilon(t)$. For a system that can hardly be approximated in an operating point by a linear time-invariant system $G_{t r}^{w}$, the noise bound $\bar{e}(t)$ will become excessively large.

The nominal model $\hat{G}$ can be any linear time-invariant model for all operating points, supplied by the user. In the remainder of this chapter we will also encounter $G_{\hat{\theta}}$. Let the parameter vector be $c$-dimensional, then $G_{\hat{\theta}}$ is related to $\hat{G}$ through

$$
\hat{\theta}=\arg \min _{\theta \in \mathbb{R}^{c}}\left\|\hat{g}(t)-g_{\theta}(t)\right\|_{h_{2}}
$$

$G_{\hat{\theta}}$ is represented in the parametrisation used internally by the algorithm.
The central model is merely a by-product of the algorithm during the analysis of the structure in the model uncertainty. The central model is needed
only during the determination of the structured error components $A_{i}(\zeta)$. It is not directly related to the nominal model. It is indirectly related to the nominal model because both are in some measure an approximation of all linear manifestations of the true system $G_{t r}$. The user may choose to use the central model as the nominal model. In this case the central model and the nominal model "happen" to be identical. This is never assumed by the algorithm.

The auxiliary models $G_{\theta^{w}}$ are models estimated for the linear manifestation of the true system in the corresponding operating point $w$, so $G_{\theta^{w}}$ is an approximation of $G_{t r}^{w}$. Like the central model, they are required for the analysis of the structure in the model uncertainty. The auxiliary models may also be used to estimate some of the required prior knowledge, as will be explained in sections 4.8.3 and 4.10.3.

Finally, the structured error components $A_{i}$ are the models parametrised by $\boldsymbol{\theta}_{A, i}$, so

$$
A_{i}(\zeta) \equiv G_{\theta_{A, i}}(\zeta)
$$

### 4.7 Step 1-4: obtaining a MIMO basis

```
Abor:hm
```




```
-Choose basis functions and model orders .... p. p. }
    bt(t): set of basis functions
    M: E. mer or basis functions
```







The first four steps are concerned with generating a MIMO basis. For our purposes, this will be equivalent to finding a basis generating system. The procedure of appendix $B$ will subsequently be followed to generate a basis that will be used for all operating points.

As pointed out in section 4.3.1, the expansion coefficients in the tail of the basis tend to zero fastest if the poles of the basis generating system are close to the poles of the true process. More exactly, this property applies for all expansion coefficients $\theta_{k}$ for which $k \geq \max \left\{\tilde{k}+1, k^{*}\right\}$. The faster the tail coefficients tend to zero, the smaller the undermodelling will be due to the truncation of the basis in the model parametrisation. In our case, the true process has a different set of pole locations for each operating point. The poles of the generating system should be chosen close to the poles in all these sets. As these pole locations are unknown, a rough model that finds a compromise between the different sets of pole locations is needed. If we have a model $\hat{\vec{G}}$,
this is a natural candidate. If we do not, we need to get an indication of the pole locations by other means. This is the purpose of step 3 .

The model order should not be taken too high. Some properties of the parameters $\theta_{t r, k}$ can only be observed if $\bar{k}$ (estimated) model parameters are available, where $\bar{k}$ is equal to several times $n$, the McMillan degree of the basis generating system $G_{b}$. If $n$ is large, many model parameters need to be estimated in order to observe these properties. This leads to a significant increase in the computational complexity of the algorithm and also to an increase in the variance error that can be expected in the estimated model parameters. On the other hand, the model order should be sufficiently high to represent all of the dominating poles of the true process.

Most identification methods can in principle be used to find a basis generating system: any method yielding a more or less accurate description of the (dominant) process poles will be suitable. Estimating FIR models of sufficient length does in general not yield a good basis generating system. All model poles will be located in the origin. Using this as a basis generating system will consequently generate the standard basis, thereby defeating the whole purpose of using system based orthonormal basis functions. A model reduction step should be performed first in such a case.

Not withstanding this, the standard basis can be used as a MIMO basis. Conceptually there is nothing prohibiting this. It will be pointed out in section 4.14, how in general this will lead to conservative error bounds.

Apart from these considerations, there are some practical constraints. Given the observation that the dominant poles of the process should be present in the basis generating system, the length of the impulse response of the generating system will be of the order of the length of the impulse response of the true system. Provided that we have done experiments of sufficient length, we can therefore assume that the impulse response of the basis generating system fits, say, five to ten times in the experimental data intervals for each data set.

To be able to estimate values for $M$ and $\rho$ in (4.5), the models to be estimated in step 5 need to have a number of parameters that is several times the McMillan degree of the basis generating system (see section 4.10). This implies, that parameters have to be estimated for basis functions that have impulse responses that are longer than that of the basis generating system. Depending on the poles of the basis generating system, this may result in impulse responses that are longer than the length of the experimental data sets. The more energy of the impulse response of a basis function is outside the experimental window, the less faith we can have that we can estimate the contribution of this basis function reliably. In such a case the parameters corresponding to higher order basis functions will be unreliable. They will in general not reveal the exponentially decaying behaviour of the parameters that we were trying to estimate in the first place. This will be explained in more detail in the next section.

If this situation occurs, the "ideal" solution would be to do longer experiments. This is not always possible, or may lead to a computational load later in the algorithm that exceeds the available computing power or the patience of the user. In these cases, the only solution left is to generate the basis with a basis generating system that has a shorter impulse response.

Example (Continued from page 84): A basis was generated for the example process using the basis generating system

$$
G_{b}(z)=\frac{1}{z-0.95}
$$

Note that the pole of this process is close to but not identical to the pole of both linear manifestations of the true process. In figure 4.2 a the impulse responses of the first five functions of a basis generated by $G_{b}$ are plotted. Figure $4.2 b$ shows the Bode amplitude plot of the same functions. Due to property 4.1 on page 77 all amplitude plots lie on top of each other in this plot.


Figure 4.2: First five basis functions in the example

### 4.8 Step 5: estimating auxiliary models

| Agontin <br> Wharabmens <br>  <br>  <br> - Estimate auxiliary models . $\qquad$ <br> $\vdash^{<} v(t), v(t), v(t)$, model parametrisation <br> $8^{w}$ : parameters of auxiliary model for operating point $w$ p. 92 <br>  <br>  <br>  |
| :---: |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

The next step in the algorithm is the estimation of auxiliary models, i.e. "nominal" models for each operating point. To avoid confusion with the "overall" nominal model $\hat{G}$, these nominal models will be referred to as auxiliary models.

As with the estimation of $\hat{G}$, the algorithm does not prescribe what identification method to use for the estimation of auxiliary models. Contrary to $\hat{G}$, the auxiliary models need to be expressed in the parametrisation of section 4.3. In this section we therefore describe an algorithm that estimates models of the form (4.3).

### 4.8.1 Instrumental variable estimation

The estimation of auxiliary models should be repeated for every operating point. We will drop the superscript $w$ that indicates that the true process is considered in an operating point. Also, we will first consider multiple input, single output processes.

For each data set $d \in \mathcal{D}(w)$, we define matrices $U_{i}^{d}$ and $V_{i}^{d}, i \in \boldsymbol{p}$ as follows:

$$
\begin{array}{r}
U_{i}^{d}:=\left[\begin{array}{ccc}
{\left[B_{0}^{(1, i)}(\zeta) u_{i}^{d}\right](0)} & \ldots & {\left[B_{\bar{k}^{(1, i)}}^{(1, i)}(\zeta) u_{i}^{d}\right](0)} \\
\vdots & & \vdots \\
{\left[B_{0}^{(1, i)}(\zeta) u_{i}^{d}\right]\left(T^{d}-1\right)} & \ldots & {\left[B_{\bar{k}^{(1, i)}}^{(1, i)}(\zeta) u_{i}^{d}\right]\left(T^{d}-1\right)}
\end{array}\right] \\
V_{i}^{d}:=\left[\begin{array}{ccc}
{\left[B_{0}^{(1, i)}(\zeta) v_{i}^{d}\right](0)} & \ldots & {\left[B_{\bar{k}^{(1, i)}}^{(1, i)}(\zeta) v_{i}^{d}\right](0)} \\
\vdots & & \vdots \\
{\left[B_{0}^{(1, i)}(\zeta) v_{i}^{d}\right]\left(T^{d}-1\right)} & \ldots & {\left[B_{\bar{k}(1, i)}^{(1, i)}(\zeta) v_{i}^{d}\right]\left(T^{d}-1\right)}
\end{array}\right]
\end{array}
$$

The columns of $U_{i}^{d}$ contain the $i$ th input filtered by the basis functions for this input in the model parametrisation. (Recall that $B_{k}^{(1, i)}$ is the $k$ th basis function of the transfer from input $i$ to the only output, numbered 1.) $V_{i}^{d}$ is similar with the process input replaced by the instrumental variable. $U^{d}$ and $V^{d}$ are in turn built from these $U_{i}^{d}$ and $V_{i}^{d}$ :

$$
U^{d}:=\left[\begin{array}{lll}
U_{1}^{d} & \ldots & U_{p}^{d}
\end{array}\right] \quad V^{d}:=\left[\begin{array}{lll}
V_{1}^{d} & \ldots & V_{p}^{d}
\end{array}\right]
$$

$Y^{d}$ shall be a column vector containing the output measurements:

$$
Y^{d}:=\left[\begin{array}{lll}
y(0) & \ldots & y\left(T^{d}-1\right)
\end{array}\right]^{T}
$$

Assuming that the order of parameters in $\theta$ is compatible with the order of filtered inputs in $U^{d}$, the instrumental variable estimate $\hat{\boldsymbol{\theta}}^{d}$ for data set $d$ would be

$$
\begin{equation*}
\hat{\boldsymbol{\theta}}^{d}:=\left[\frac{1}{T^{d}}\left(V^{d}\right)^{T} U^{d}\right]^{-1} \frac{1}{T^{d}}\left(V^{d}\right)^{T} Y^{d} \tag{4.10}
\end{equation*}
$$

This estimation method has been analysed in (Hakvoort, 1994), section 5.3 and 5.4. If the inputs are used as instrumental variables, (4.10) becomes the least squares estimate. This has been analysed using the framework of (Ljung, 1987) in (Van den Hof et al., 1995). It is shown there that, provided the inputs are rich enough, the estimate (4.10) converges under weak conditions asymptotically to the true process parameters.

To obtain one model for all data sets of an operating point, we finally introduce $U, V$ and $Y$ :

$$
\begin{equation*}
U=\operatorname{stack}_{d \in \mathcal{D}(w)} f^{d} U^{d}, \quad V=\operatorname{stack}_{d \in \mathcal{D}(w)} f^{d} V^{d}, \quad Y=\operatorname{stack}_{d \in \mathcal{D}(w)} f^{d} Y^{d} \tag{4.11}
\end{equation*}
$$

where $f^{d} \in \mathbb{R}_{+}, d \in \mathcal{D}(w)$, are weighting factors for each data set. As the parameter estimate for the operating point is taken

$$
\begin{equation*}
\hat{\boldsymbol{\theta}}:=\left[V^{T} U\right]^{-1} V^{T} Y \tag{4.12}
\end{equation*}
$$

Being an instrumental variable estimate, it is hard to tell what criterion is optimised by the "optimal" parameter vector $\hat{\boldsymbol{\theta}}$. If the instrumental variables are taken equal to the inputs, the instrumental variable method reduces to a least squares method. This can be restated as follows. Let $\hat{\epsilon}^{d}(t, \boldsymbol{\theta})$ be the prediction error for data set $d$ at time $t$ for the model with parameter vector $\boldsymbol{\theta}$. It can easily be verified, that

$$
Y-U \boldsymbol{\theta}=\operatorname{stack}_{d \in \mathcal{D}(w)} f^{d}\left[\begin{array}{c}
\hat{\epsilon}^{d}(0, \theta) \\
\vdots \\
\hat{\epsilon}^{d}\left(T^{d}-1, \theta\right)
\end{array}\right]=: \hat{E}(\boldsymbol{\theta})
$$

The estimate $\hat{\boldsymbol{\theta}}$ from (4.12) minimises $\|\hat{E}(\theta)\|_{2}^{2}$, in other words

$$
\hat{\theta}=\arg \min _{\theta}\|\hat{E}(\theta)\|_{2}^{2}=\arg \min _{\theta} \sum_{d \in \mathcal{D}(w)} f^{d} \sum_{t \in \mathbb{T}^{d}} \hat{\epsilon}^{d}(t, \theta)^{2}
$$

Taking all $f^{d}$ equal to 1 will weight each data set proportionally to its length. Provided noise levels are comparable in each data set, this seems most
reasonable. If noise levels are different, data sets with a higher noise level should be weighted less.

For processes with multiple outputs, the procedure outlined above can simply be applied for each output, taking instead of $\boldsymbol{\theta}$ the relevant sub-vector of $\boldsymbol{\theta}$. This is possible, because transfers to different outputs are parametrised independently.

Remark 4.3 The "gluing together" of the data sets does not give problems because of mismatching initial conditions: the inputs are filtered by the basis functions first and then concatenated.

This is not meant to imply that the initial conditions at the start of the data sets do not complicate the estimation. At the start of each data set there is a possible transient in the output, that is, at least in the least squares estimation discussed here, not taken into account in the estimation. Depending on the magnitude of the transient and the length of the transient compared to the length of the data set, this may lead to a smaller or larger bias in the estimation.

This is the case for a single data set as well and does not result from the combination of data sets. The point that is made here, is that the concatenation does not cause additional problems, as would have been the case for example if the data sets were concatenated first and then filtered.

See also section 5.2.

Remark 4.4 The estimate (4.12) has essentially the same properties as the corresponding estimate for a single data set. This means for example, that the prediction error can be weighted in frequency domain by prefiltering input, output and instrumental variable. (See for example (Ljung, 1987).) Also, the estimate (4.12) converges asymptotically to the true parameter values under the same conditions as the normal instrumental variable method does. "Asymptotically" should be interpreted here such, that the length of at least one of the data sets goes to infinity. If the length of the data sets remains finite but the number of data sets tends to infinity, additional assumptions on the initial conditions in each data set are required for consistency of the estimate.

Example (Continued from page 90): The results obtained for the simulation example used in this chapter by the estimation method discussed above are shown in figure 4.3. This figure clearly shows, that the estimated model fails to capture the initial transient. Indeed, the estimation method discussed in this section provides no means to represent the transient. After this transient, the residuals become small compared to the output signal. The power in the residuals becomes then comparable to the power in the true noise signals.

Note that the comparison of figure 4.3 is carried out on the estimation set, not on a validation set.



Figure 4.3: Residuals ('-') and outputs ('--') for data set a (top) and b (bottom)

No plots of true and estimated impulse responses or Bode plots are provided, as the estimated plots can hardly be distinguished from the corresponding true plots.
(Continued on page 99.)

### 4.8.2 Mismatch between data length and basis length

As already indicated in section 4.7 , identification will be troublesome if the length of the impulse responses of the basis functions in the model set exceeds the length of the data. (See also (1.2) on page 14.) To illustrate this, consider the situation of a SISO system for which one data set is available. The instrumental variable is taken equal to the input. The estimate is given by (4.10), which reduces in this situation to

$$
\hat{\boldsymbol{\theta}}=\left[\frac{1}{T} U^{T} U\right]^{-1} \frac{1}{T} U^{T} Y
$$

Let $Q$ denote the matrix to be inverted in the right hand side:

$$
Q:=\frac{1}{T} U^{T} U
$$

Assume the input $u$ is a white, quasi-stationary sequence, where the value of $u$ on each time instant $t \in \mathbb{T}^{d}$ is normaily distributed with zero mean and standard deviation $\sigma_{u}$. For the length of the data set tending to infinity, the element of $Q$ in the $j$ th row and the $i$ th column becomes

$$
\begin{align*}
{[Q]_{j i} } & =\lim _{T \rightarrow \infty} \frac{1}{T} \sum_{t \in \mathbb{T}}\left[B_{j}(\zeta) u\right](t)\left[B_{i}(\zeta) u\right](t) \\
& =\lim _{T \rightarrow \infty} \frac{1}{T} \sum_{t \in \mathbb{T}}\left(\sum_{\tau_{1}=0}^{t} b_{j}\left(\tau_{1}\right) u\left(t-\tau_{1}\right)\right) \cdot\left(\sum_{\tau_{2}=0}^{t} b_{i}\left(\tau_{2}\right) u\left(t-\tau_{2}\right)\right) \\
& =\lim _{T \rightarrow \infty} \sum_{\tau_{1} \in \mathbb{T}} \sum_{\tau_{2} \in \mathbb{T}} b_{j}\left(\tau_{1}\right) b_{i}\left(\tau_{2}\right) \cdot \frac{1}{T} \sum_{t=\max \left\{\tau_{1}, \tau_{2}\right\}}^{T-1} u\left(t-\tau_{1}\right) u\left(t-\tau_{2}\right) \\
& =\sigma_{u}^{2} \sum_{\tau=0}^{\infty} b_{j}(\tau) b_{i}(\tau)=\sigma_{u}^{2} \delta(j-i) \tag{4.13}
\end{align*}
$$

where the third equality follows from the whiteness of the noise and the fourth from the orthonormality of the basis functions. (4.13) simply means that $Q$ is asymptotically equal to a scalar times the identity matrix for the indicated input.

To investigate the properties of this least-squares identification method, data was generated by filtering an input sequence with the system

$$
G_{t r}(z)=\frac{z-0.5381}{(z-0.95)(z-0.80)}
$$

The input $u$ was a white noise sequence with a distribution $\mathcal{N}(0,1)$, consisting of 1000 samples. The output $y$ was the response of $G_{t r}(z)$ to this input. Initial conditions were set to 0 , no noise was added to either $u$ or $y$. A basis was generated with the system

$$
G_{b}(z)=\frac{1.7918}{z-0.99}
$$

30 basis functions were incorporated in the model, so $\bar{k}$ is taken 30.
The true and the estimated impulse response of the system are shown in figure 4.4. In figure 4.4a the step responses are plotted for the first 100 samples. In figure 4.4 b the same responses are plotted for the first 7000 samples.

Although the true process seems to be estimated fairly accurately judging from the short step responses, the plot of the step responses for 8500 samples clearly demonstrates that the match between the estimated model and the true system is actually very bad.

This can not be due to a variance effect for which the identification procedure happens to be extremely sensitive, because no noise was added to the


Figure 4.4: Step responses, '-': true system, '--': estimated model
input or output signals. The only "noise" that is present is the quantisation noise caused by round-off errors in the computer. Double precision arithmetic was used, so the signal to noise ratio is extremely good. As can be seen in figure 4.4 a, the input record of 1000 samples should be long enough to identify this system.

The actual singular values of $Q$ are shown in figure 4.5. These do not correspond to $Q=I$ which holds asymptotically according to (4.13). This in turn is caused by the fact, that the response of most of the basis functions is much longer than the 1000 samples in the experiment record. The responses of the basis functions are not plotted here, but one may get an idea of their length from the observation that the length of the response of the estimated model is roughly equal to the length of the response of the last basis functions. The length of the response of the model can be judged from figure 4.4 b .

Theoretically it can be shown, that the true model parameters decay exponentially with $\rho=0.9135$. This can not be seen from the estimated parameters in figure 4.6!

Remark 4.5 In section 5.3, a way to suppress the ill-conditioning of this problem will be introduced and the results will be compared to those obtained in this section. A brief discussion of other methods known from literature to handle this problem will be held there as well.

These results are not presented here, as the purpose of the current section was only to illustrate the problems that can be expected if the model set contains basis functions with an impulse response length that exceeds the experimental data record significantly.


Figure 4.5: Singular values of $Q$

### 4.8.3 Estimating prior information

Prior information that has to be estimated from experimental data is not really prior information. However, as the kind of information discussed here is generally called prior information, this name will be used here for it as well. This section is intended as a means to obtain the information if it is not known a priori. If it is, then the estimation treated in this section can be skipped.

If no values of $M^{\sigma}, \rho^{\sigma}$ and $\theta_{\max }^{\sigma}$ in (4.5) are available, they may be estimated from the estimated auxiliary models. In the first place values $M^{\sigma}$ and $\rho^{\sigma}$ need to be fitted to the magnitude of the estimated parameters. To quantify the mismatch between an estimated parameter $\hat{\theta}_{k}^{\sigma}$ and the bound $M^{\sigma}\left(\rho^{\sigma}\right)^{k}$ on its magnitude it is important to realise that a small difference $M^{\sigma}\left(\rho^{\sigma}\right)^{k}-\left|\hat{\theta}_{k}^{\sigma}\right|$ should be considered a large mismatch if $\hat{\theta}_{k}^{\sigma}$ is small as well.

This would suggest using the sum of "relative misfits"

$$
\sum_{k=0}^{\bar{k}^{\sigma}}\left|\frac{M^{\sigma}\left(\rho^{\sigma}\right)^{k}-\left|\hat{\theta}_{k}^{\sigma}\right|}{\left|\hat{\theta}_{k}^{\sigma}\right|}\right|
$$

as a measure for the overall misfit. This measure is very sensitive to differences $M^{\sigma}\left(\rho^{\sigma}\right)^{k}-\left|\hat{\theta}_{k}^{\sigma}\right|$ for values of $\hat{\theta}_{k}^{\sigma}$ close to zero. If this value is close to zero because of the exponential decay of the model parameters, this is exactly what is intended. However, such small values may also occur for smaller values of $k$ for which the envelope $M^{\sigma}\left(\rho^{\sigma}\right)^{k}$ should actually still be large, i.e. not close to zero, because the value of, say, $\hat{\theta}_{k-1}^{\sigma}$ and $\hat{\theta}_{k+1}^{\sigma}$ is much larger.

One can also weight the relative misfit but be more robust against interme-
diate small values of $\hat{\theta}_{k}^{\sigma}$ by using

$$
\begin{equation*}
\sum_{k=0}^{\bar{k}^{\sigma}}\left|\ln M^{\sigma}+k \ln \left(\rho^{\sigma}\right)-\ln \right| \hat{\theta}_{k}^{\sigma}| | \tag{4.14}
\end{equation*}
$$

as a measure for the overall misfit.
While (4.14) is a reasonable measure for the misfit between the estimated model parameters and the exponentially decreasing envelope $M^{\sigma}\left(\rho^{\sigma}\right)^{k}$ for it, it is not sufficient to minimise (4.14) for $M^{\sigma}$ and $\rho^{\sigma}$ to find the "optimal" envelope: the envelope is required to overbound the magnitude of the model parameters. This can be achieved by adding the constraints

$$
\begin{equation*}
\forall k=0, \ldots, \bar{k}^{\sigma} \quad M^{\sigma}+\left(\rho^{\sigma}\right)^{k} \geq\left|\hat{\theta}_{k}^{\sigma}\right| \tag{4.15}
\end{equation*}
$$

to the optimisation. This implies that the outer modulus signs may also be dropped from (4.14). Finally, the envelope should be exponentially decreasing. This corresponds to the constraint

$$
\begin{equation*}
\rho^{\sigma}<1 \tag{4.16}
\end{equation*}
$$

By changing to the variables $M^{\prime}=\ln M^{\sigma}$ and $\rho^{\prime}=\ln \rho^{\sigma}$ equations (4.14) to (4.16) can (almost) be translated to the following linear programming problems:

$$
\begin{equation*}
\operatorname{minimise} \quad \sum_{k=0}^{\bar{k}^{\sigma}}\left|M^{\prime}+k \rho^{\prime}-\ln \right| \hat{\theta}_{k}^{\sigma}| | \quad \text { for } M^{\prime}, \rho^{\prime}, \tag{4.17}
\end{equation*}
$$

subject to

$$
\begin{gather*}
\forall k=0, \ldots, \bar{k}^{\sigma} \quad M^{\prime}+k \rho^{\prime} \geq \ln \left|\hat{\theta}_{k}^{\sigma}\right|  \tag{4.18}\\
\rho^{\prime} \leq 0 \tag{4.19}
\end{gather*}
$$

For different $\sigma$ 's, different optimisations have to be performed, yielding different values for $M^{\prime}$ and $\rho^{\prime}$. As a concession to numerical solvability, (4.19) also allows $\rho^{\prime}=0 \Leftrightarrow \rho=1$. For an exponentially decreasing bound this is actually not allowed. As an alternative one could require that $\rho^{\prime} \leq-\epsilon$ for some small, positive value of $\epsilon$.
$\theta_{\max }^{\sigma}$ can be taken equal to

$$
\theta_{\max }^{\sigma}=s_{\theta} \cdot \max _{k=0, \ldots, \bar{k}^{\sigma}}\left|\hat{\theta}_{k}^{c}\right|
$$

where $s_{\theta} \geq 1$ is a safety margin to be chosen properly. $\left(k^{*}\right)^{\sigma}$ should be taken the largest integer such that

$$
M^{\sigma}\left(\rho^{\sigma}\right)^{\left(k^{*}\right)^{\sigma}}>\theta_{\max }^{\sigma}
$$



Figure 4.7: Estimated model parameters and parameter bounds for the simulation example

Example (Continued from page 94): The parameters that were estimated in operating point a for the simulation example are shown in figure 4.7. The corresponding values of $M^{a}$ and $\rho^{a}$ are 12.95 and 0.4480 respectively. These bounds will be reconsidered in section 5.2.
(Continued on the next page.)
An estimate for the noise bound $\bar{e}^{d}(t)$ in (4.7) is easily obtained by taking

$$
\left.\forall t \in \mathbb{T}^{d} \quad \bar{e}^{d}(t)=s_{e} \cdot \max _{\tau \in \mathbb{T}^{d}} \mid \hat{\epsilon}^{d}(\tau)\right) \mid
$$

where $s_{e}$ is another safety margin, $s_{e} \geq 1$. The noise bound thus becomes a constant signal. For multivariate $\bar{e}^{d}(\bar{t})$, maximisation and absolute values should be taken element-wise.

A tighter bound would be given by

$$
\left.\forall t \in \mathbb{T}^{d} \quad \bar{e}^{d}(t)=s_{e} \cdot \mid \hat{\epsilon}^{d}(\tau)\right) \mid
$$

but this bound may be less realistic. As an extreme example, if $\hat{\epsilon}\left(t^{\prime}\right)$ is equal to zero for some $t^{\prime}$, the noise bound will be zero. For small $\hat{\epsilon}\left(t^{\prime}\right)$ something similar, though less drastic, occurs. Therefore, it seems safer to deduce one noise level from the residuals for all samples and determine the noise bound for all samples from this noise level.

Estimating the cross-covariance bounds $\bar{c}_{\ell}^{d}, \ell \in n_{r}^{d}$ proceeds analogously:

$$
\forall \ell \in n_{r}^{d} \quad \bar{c}_{\ell}^{d}:=\frac{s_{c}}{\sqrt{T^{d}}}\left|\sum_{t \in \mathbb{T}^{d}} \hat{\epsilon}^{d}(t) r_{\ell}^{d}(t)\right|
$$

where $s_{c} \geq 1$ is a safety margin.
If the order of the models considered during estimation is chosen (too) high, it is likely that part of the particular noise realisation that is present in a data set is fitted by the estimated models. This will lead to smaller estimates of the residuals $\hat{\epsilon}^{d}(t)$ and consequently too smaller values of $\bar{e}^{d}(t)$ and $\bar{c}_{\ell}^{d}$. If the model orders are such, that there is a clear risk of this, it is to be preferred to base the residuals $\hat{\epsilon}(t)$ on separate validation data sets and estimate the values of $\bar{e}$ and $c_{\ell}$ from these. This requires for each data set $d \in \mathcal{D}$ a companion data set in which the same noise level can be expected as in $d$. Note that under certain conditions, the same companion data set can be used for all $d \in \mathcal{D}(w)$ with $w \in \mathcal{W}$.

Example (Continued from the preceding page): The estimated values for $\bar{e}(t)$ are shown in figure 4.8 for both operating points. For $s_{e}$ the value 1.2 was used.


Figure 4.8: Estimated residuals and noise bounds

The noise bound is dominated by the transient in the residuals.
The estimated cross-covariance bounds are

$$
c_{1}^{a}=0.1687, \quad c_{1}^{b}=1.4681
$$

where for $s_{c}$ too the value 1.2 was used.
In section 5.2 further attention will be paid to the influence of the transient on the noise bound and the cross-covariance bound.
(Continued on page 106.)

### 4.9 Step 6 \& 7: estimating the error structure



Step 6 and step 7 are treated together, because the way in which the error structure is determined will turn out to be dictating the way in which the central model should be calculated. The error structure will be determined in this section by principal component analysis. Other ways are discussed in section 5.7.

### 4.9.1 Principal component analysis

Principal component analysis is a technique encountered often in multivariate data analysis. In fact, subtle variants on the same theme are collectively referred to as principal component analysis. For a thorough introduction into this subject, the reader is referred to Jolliffe (1986). This reference interprets principal component analysis in many different ways. Most textbooks on multivariate data analysis can be consulted as well. In this section we interpret principal component analysis in a way that fits best to the application we have in mind. Our aim is not to be completely rigorous, just to provide enough information to appreciate the choices made later on.

Consider a zero-mean $n$-dimensional stochastic variable $\underline{x}$. This variable could be generated by a single, unknown random process $\underline{e}$ :

$$
\underline{x}=p \underline{e} \quad \underline{x}, p \in \mathbb{R}^{n}, \underline{e} \in \mathbb{R} .
$$

(The notation $\underline{x} \in \mathbb{R}^{n}$ means that $\underline{x}$ is a stochastic variable taking its values in $\mathbb{R}^{n}$.) On the other hand, it could be generated by $n$ independent random
processes $\underline{e}_{1}, \ldots, \underline{e}_{n}$, also assumed to be unknown:

$$
\underline{x}=\sum_{i=1}^{n} p_{i} \underline{e}_{i} \quad \underline{x}, p_{i} \in \mathbb{R}^{n}, \underline{e}_{i} \in \mathbb{R}
$$

If $\left\langle p_{i}\right\rangle_{i=1}^{n}=\mathbb{R}^{n}$, it is clear that $\underline{x}$ being generated by more than $n$ independent random processes can not be distinguished from the case that $\underline{x}$ is generated by $n$ independent processes, provided we have no prior information about the probability distribution of $\underline{e}_{i}$. If $\left\langle p_{i}\right\rangle_{i=1}^{n} \subsetneq \mathbb{R}^{n}$ we can find $p_{j}^{\prime}$ and $\underline{e}_{j}^{\prime}$, where $j=1, \ldots, n-1$ such that

$$
\underline{x}=\sum_{j=1}^{n-1} p_{j}^{\prime} \underline{e}_{j}^{\prime} \quad \underline{x}, p_{j}^{\prime} \in \mathbb{R}^{n}, \underline{e}_{j}^{\prime} \in \mathbb{R}
$$

The stochastic properties of $\underline{x}$ are not known, but $m$ samples have been taken from $\underline{x}$, denoted $x_{1}$ through $x_{m}$. Principal component analysis proceeds as follows. The $x_{1}$ to $x_{m}$ are collected in a matrix $X$ :

$$
X:=\left[\begin{array}{lll}
x_{1} & \ldots & x_{m} \tag{4.20}
\end{array}\right]
$$

Denote the singular value decomposition of $X$

$$
\begin{equation*}
X=: U \Sigma V^{T} \tag{4.21}
\end{equation*}
$$

with $\sigma_{1} \geq \sigma_{2} \geq \cdots \geq \sigma_{n}$ the singular values. (This assumes $m \geq n$. If $m<n$ there are only $m$ singular values.) $\underline{x}$ can now be written as

$$
\begin{equation*}
\underline{x}=\sum_{i=1}^{n} U_{* i} \underline{e}_{i}, \quad \underline{e}_{i} \in \mathbb{R}, \operatorname{var} \underline{e}_{i}=\sigma_{i}^{2} \tag{4.22}
\end{equation*}
$$

This decomposition is optimal in the sense that $U_{* 1} e_{1}$ minimises the variance of $\underline{x}-\underline{\hat{x}}$ for all $\underline{\hat{x}}$ taken from the set of random variables generated by a single random process. Moreover, $U_{* 1} \underline{e}_{1}+U_{* 2} \underline{e}_{2}$ minimises the variance of $\underline{x}-\underline{\hat{x}}$ if $\hat{x}$ is taken from the set of random variables generated by two independent stochastic processes. This can be extended for up to $\min \{m, n\}$ independent stochastic processes.

The columns of $U$ are called the principal components of $X$. According to (4.22), the singular values can be interpreted as the standard deviation of the processes "driving" the principal components. A column of $V$ indicates how the different samples contribute to the corresponding principal component. This will be discussed in more detail in section 5.6.2.

In figure 4.9 an example is shown for a stochastic variable $\underline{x} \in \mathbb{R}^{2}$. In this example

$$
\underline{x}=p_{1} \underline{e}_{1}+p_{2} \underline{e}_{2} \quad \quad p_{1}=\binom{1.86}{1.23}, \quad p_{2}=\binom{0.99}{-0.06} .
$$

The generating processes $\underline{e}_{1}$ and $\underline{e}_{2}$ are independent with a standard normal distribution. The dots in figure 4.9 indicate different samples taken from $\underline{x}$. The two straight lines are the vectors $p_{1}$ and $p_{2}$. The axes of the ellipse are $\sigma_{1} U_{* 1}$ and $\sigma_{2} U_{* 2}$. Denote the processes driving the principal components $\underline{e}_{a}$ and $e_{b}$ respectively. In figure 4.9 the principal components are clearly not aligned with $p_{1}$ or $p_{2}$. So although the principal components $\sigma_{1} U_{* 1} \underline{e}_{a}$ and $\sigma_{2} U_{* 2} \underline{e}_{b}$ have the interpretation of independent stochastic variables yielding together the stochastic variable $\underline{x}$, principal component analysis does not necessarily find the "true" stochastic variables $p_{1} \underline{e}_{1}$ and $p_{2} \underline{e}_{2}$ that constitute $\underline{x}$.


Figure 4.9: Example of principal component analysis (see text)

Remark 4.6 The requirement that $\underline{x}$ is a zero-mean signal is in practice removed by subtracting the mean of $\underline{x}$ from the samples before doing the analysis.

Remark 4.7 Principal component analysis is able to reveal correlation between the elements of a varying, or rather, stochastic vector. This means that the principal components describe the dominating linear relationships in a random vector. If the second element of a vector is for example always the square of the first element, principal component analysis will not detect this. It may reveal a linear relationship that happens to fit to the samples taken from the stochastic variable. What this relationship looks like depends entirely on what samples have been taken.

### 4.9.2 Principal component analysis and error structure

As was demonstrated in the previous section, principal component analysis can be used to find the dominant (linear) relationships between the elements of a varying vector, interpreted there as a multivariate random variable, based on a number of samples taken from that variable. The problem that is actually faced by the algorithm is to find the dominant relationships between the variations in a process transfer when the process moves between different operating points. The latter problem can be mapped onto the former by representing a process transfer in an operating point by the parameter vector of the auxiliary model for that operating point. This parameter vector is a varying quantity because the operating point of the process may vary. Because models are identified for a number of operating points, a number of "samples" of this varying quantity is available.

In the previous paragraph it is implicitly assumed, or rather hoped, that the structure that is present in the changes in the transfer of the process, induced by the change of operating point, is reflected in the changes of the auxiliary model parameter vectors in a way that can be detected by principal component analysis. The structured error components in (4.4) are additive uncertainties. Such an additive uncertainty corresponds to a linear relationship between parameter vectors if the model parametrisation is linear in the parameters. So, provided the structure in the process uncertainty is indeed of the form (4.4), principal component analysis of the auxiliary model parameter vectors will detect this structure if the model is linearly parametrised. Note that the model parametrisation used by the algorithm is indeed linear in the parameters. (In the remainder of this chapter, reasons of computational tractability will be given that restrict the model parametrisation to a linear parametrisation too.)

Structure in the model uncertainty that is known to exist from physical reasoning, but that can not be expressed as an additive error with a varying gain, will not be found by the principal component analysis. At best only a linear approximation to the structure will be found.

One may wonder how restrictive the limitation to additive structured uncertainties is in practice. An important consequence of it is, that moving poles can only be described approximatingly as a structured error component. This kind of process uncertainty is often found in practical applications. On the other hand, in many practical applications reported in literature an additive process uncertainty is assumed, albeit an unstructured one, without physical or theoretical justification. As this leads nevertheless to useful control designs, one may be optimistic concerning the limitations imposed by the restriction to additive structured error components.

If one wants to generalise the algorithm to different model parametrisations it should be verified what a linear relation between the model parameters means in terms of process transfers. Consider for example a simple second order ARX
structure

$$
\hat{y}(t)=b_{0} u(t)+b_{1} u(t-1)+b_{2} u(t-2)-a_{1} y(t-1)-a_{2} y(t-2)
$$

It is hard to think of a general reason why there should be a linear dependency between the model parameters if the process moves to different operating points. This does not imply that no particular cases are conceivable where such a structure could give good results. To incorporate such structuring in the process uncertainty, the uncertainty description (4.4) has to be revised. Many other changes to the algorithm would be necessary as well.

In section 5.8 an example is given in which a different uncertainty description can be used with relatively small adjustments to the algorithm.

The central model is introduced into the algorithm because of remark 4.6 on page 103. This remark stated, that the samples of a stochastic process analysed by principal component analysis should have a mean value of zero. This makes it clear that the parameters of the central model should be taken equal to the mean of the parameters for each operating point:

$$
\begin{equation*}
\boldsymbol{\theta}^{c}:=\frac{1}{n_{w}} \sum_{w \in \mathcal{W}} \theta^{w} \tag{4.23}
\end{equation*}
$$

Following the interpretation outlined above, the set of $\Delta \boldsymbol{\theta}^{w}:=\boldsymbol{\theta}^{w}-\boldsymbol{\theta}^{c}, w \in$ $\mathcal{W}$ is now a set of samples from a zero mean process. Collecting these samples in the columns of a matrix $X$ as in (4.20), the principal components are given by the columns of the matrix $U$ in the singular value decomposition (4.21) of $X$.

How many principal components to use is not easy to tell. The singular values can be taken as a first indication. If there is a sharp drop and then a stabilisation to an approximately constant level in the sequence of singular values, the location of the drop is likely to be a good value for the number of principal components. The variation beyond this number can be ascribed to "noise effects," which would translate in this situation to variance errors in the model parameters. In practice however, there are probably too few operating points for which we have experimental data to be able to see such a clear pattern. A more pragmatic approach would be to take a number of principal components of the order of the number of operating points first and check whether decreasing this number inflates the resulting unstructured error $\Delta$. As long as this is not the case, one can safely reduce the number of structured error components. In section 4.12 it will be argued that "the more the better" need not apply to the number of principal components. See remark 4.15 on page 132 .

Once the number $n_{s}$ has been determined, the parameters $\boldsymbol{\theta}_{A, i}, i \in \boldsymbol{n}_{s}$ are defined to be the first $n_{s}$ columns of $U$ in the singular value decomposition $X=U \Sigma V^{T}$. Using the model parametrisation (4.3) this results in the transfers
$A_{i}:$

$$
A_{i}(\zeta) \equiv G_{\boldsymbol{\theta}_{A, i}}(\zeta), \quad i \in \boldsymbol{n}_{\boldsymbol{s}}
$$

Example (Continued from page 101): In the simulation example with only two operating points the problem of determining the error structure reduces to a somewhat trivial situation. For two operating points one can use either zero or one structured error components. Moreover, it holds

$$
\Delta \theta^{a}=-\Delta \boldsymbol{\theta}^{b}=\frac{\theta^{a}-\theta^{b}}{2}
$$

so that the structured error component, if it is used, is aligned with the difference between the auxiliary models for the two operating points.
(Continued on page 109.)

### 4.10 Step 8: translating data and prior knowledge

The experimental data and prior knowledge listed in section 4.4 will now be translated to a set of linear constraints on the parameter vector $\boldsymbol{\theta}$. The results presented here were published earlier in (Hakvoort, 1994, chapter 4). Computational details were presented in appendix 4.C of that reference. Readers familiar with these results can skip this section.

The translation will consist mainly of an extension of the noise bound (4.7) and of the cross-covariance bound (4.9). The parameter bounds (4.5), possibly estimated following section 4.8.3, need no further translation.

### 4.10.1 Extended noise bound

|  |  |  |
| :---: | :---: | :---: |
| - - Be mentimax thas |  |  |
|  |  |  |
| - C6men |  |  |
|  |  |  |
|  |  |  |
| - Construct set of linear constraints ......................... . p. 100 |  |  |
|  |  |  |
| - Extend noise boumd. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . p, 106 |  |  |
| -くe (t): (unextended) noise bound. p. 82 |  |  |
| $\longrightarrow \hat{a}(t)$ : influence of basis $\bar{k}+1$ to $\hat{k}$, no transient p, 107 |  |  |
| $\longrightarrow \hat{b}(t)$ : transient of basis 0 to $\hat{k}$ |  |  |
| $\rightarrow \delta:$ inflnence of basis $\tilde{k}+1$ and higher, incl. transient p. 108 |  |  |
| $\longrightarrow\left\|y(t)-G_{\theta}(\zeta) u(t)\right\| \leq \bar{e}_{e}(t):=\bar{e}(t)+\hat{a}(t)+\hat{L}(t)+\delta \quad$ p. 108 |  |  |
|  |  |  |
| $\rightarrow$ Ly: set of linear constraints on $\theta$ for every operating point |  |  |
|  |  |  |

Introduce the noise-free process output for data set $d, d \in \mathcal{D}(w)$ :

$$
\forall t \in \mathbb{T}^{d} \quad y_{t r}^{d}(t):=y^{d}(t)-\epsilon^{d}(t)
$$

Based on (4.6), we have

$$
\forall t \in \mathbb{T}^{d} \quad y_{t r}^{d}(t)=\sum_{\sigma \in S} \sum_{k=0}^{\infty}\left(\theta_{t r}\right)_{k}^{\sigma} E^{\sigma} B_{k}^{\sigma}(\zeta) u^{d}(t)
$$

Consider for a moment the case of a single input, single output system, so that we can drop the superscript $\sigma$ and the summation over $\mathcal{S}$. We will also drop the superscript $d$ on the signals $u$ and $y$. We thus have for all $t \in T^{d}$ :

$$
\begin{align*}
y_{t r}\left(t^{\prime}\right) & =\sum_{k=0}^{\infty} \theta_{t r, k} B_{k}(\zeta) u\left(t^{\prime}\right) \\
& =\sum_{k=0}^{\infty} \theta_{t r, k} \sum_{\tau=0}^{\infty} b_{k}(\tau) u\left(t^{\prime}-\tau\right) \tag{4,24}
\end{align*}
$$

This involves basis functions that are not incorporated in the model and also values $u(t)$ are used for $t<0$, which we do not have in our data set. (4.24) can be split into four components as follows

$$
\begin{align*}
y_{t r}\left(t^{\prime}\right) & =\sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{\tau=0}^{t^{\prime}} b_{k}(\tau) u\left(t^{\prime}-\tau\right)+\sum_{k=\tilde{k}+1}^{\hat{k}} \theta_{t r, k} \sum_{\tau=0}^{t^{\prime}} b_{k}(\tau) u\left(t^{\prime}-\tau\right) \\
& +\sum_{k=0}^{\tilde{k}} \theta_{t r, k} \sum_{\tau=t^{\prime}+1}^{\infty} b_{k}(\tau) u\left(t^{\prime}-\tau\right)+\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \sum_{\tau=0}^{\infty} b_{k}(\tau) u\left(t^{\prime}-\tau\right) \tag{4.25}
\end{align*}
$$

where $\tilde{k} \in \mathbb{N}, \tilde{k} \geq \bar{k}$ is a parameter that can be manipulated to influence the conservatism introduced in the following steps.

Now consider the second term in the right hand side of (4.25) (or rather its absolute value):

$$
\begin{equation*}
\left|\sum_{k=\bar{k}+1}^{\tilde{k}} \theta_{t r, k} \sum_{\tau=0}^{t^{\prime}} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right| \leq \sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|\sum_{\tau=0}^{t^{\prime}} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right|=: \hat{a}\left(t^{\prime}\right) \tag{4.26}
\end{equation*}
$$

For all $t \in \mathbb{T}, \hat{a}(t)$ can be calculated with a finite number of calculations. The third term can be bounded as follows

$$
\begin{align*}
& \left|\sum_{k=0}^{\tilde{k}} \theta_{t r, k} \sum_{\tau=t^{\prime}+1}^{\infty} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right| \leq \sum_{k=0}^{\tilde{k}} \bar{\theta}_{k}\left|\sum_{\tau=t^{\prime}+1}^{\infty} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right| \leq \\
& \leq \sum_{k=0}^{\bar{k}} \bar{\theta}_{k} \sum_{\tau=t^{\prime}+1}^{\infty}\left|b_{k}(\tau)\right| \bar{u}=\bar{u} \sum_{k=0}^{\bar{k}} \bar{\theta}_{k}\left(\left\|B_{k}(\zeta)\right\|_{\ell_{1}}-\sum_{\tau=0}^{t^{\prime}}\left|b_{k}(\tau)\right|\right)=: \hat{b}\left(t^{\prime}\right) \tag{4.27}
\end{align*}
$$

This can be calculated to any desired accuracy, see for example Fialho and Georgiou (1995). The results in this reference are derived for uncertain linear systems but can easily be reduced to the case of a single known linear system. The last term of (4.25) requires unfortunately use of conjecture 4.2 on page 77 with $p$ taken equal to 0 :

$$
\begin{align*}
&\left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \sum_{\tau=0}^{\infty} b_{k}(\tau) u(t-\tau)\right| \leq \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \sum_{\tau=0}^{\infty}\left|b_{k}(\tau)\right| u_{m} \leq \\
& u_{m} \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left\|B_{k}(\zeta)\right\|_{\ell_{1}} \leq u_{m} \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left(c_{1}^{\prime}+c_{2}^{\prime} \sqrt{k}\right)=: \delta \tag{4.28}
\end{align*}
$$

where $u_{m}:=\max \{\bar{u},|u(0)|,|u(1)|, \ldots,|u(t)|\}$ and $c_{1}^{\prime}$ and $c_{2}^{\prime}$ are taken such that for all $k$

$$
\begin{equation*}
\left\|B_{k}(\zeta)\right\|_{\ell_{1}} \leq c_{1}+c_{2} \sqrt{\left\lfloor\frac{k-1}{n}\right\rfloor} \leq c_{1}^{\prime}+c_{2}^{\prime} \sqrt{k} \tag{4.29}
\end{equation*}
$$

$c_{1}$ and $c_{2}$ are chosen according to conjecture $4.2, p=0$. Calculation of (4.28) involves summations of the form

$$
\sum_{k=k_{0}}^{\infty} \rho^{k} \sqrt{k}
$$

For this infinite sum, no analytic expression is known. It can be calculated to arbitrary accuracy however using

$$
\sum_{k=k_{0}}^{k_{1}} \rho^{k} \sqrt{k}<\sum_{k=k_{0}}^{\infty} \rho^{k} \sqrt{k}<\sum_{k=k_{0}}^{k_{1}} \rho^{k} \sqrt{k}+\sum_{k=k_{1}+1}^{\infty} \rho^{k} k
$$

The last term can be calculated analytically and can be made arbitrarily small by taking $k_{1}$ sufficiently large.

The necessity to rely on a conjecture makes the foundation for the latter bound not rock solid. However, we will show later that in practice there is no risk that the bounds based on (4.26), (4.27) and (4.28) will be too optimistic.

The noise bound (4.7) can now be extended:

$$
\begin{align*}
&\left|y_{t r}\left(t^{\prime}\right)-\sum_{k=0}^{\bar{k}} \theta_{t r} \sum_{\tau=0}^{t^{\prime}} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right| \leq \\
& \leq\left|y_{t r}\left(t^{\prime}\right)-y\left(t^{\prime}\right)\right|+\left|y\left(t^{\prime}\right)-\sum_{k=0}^{k} \theta_{t r} \sum_{\tau=0}^{t^{\prime}} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right| \leq \\
& \leq \bar{e}\left(t^{\prime}\right)+\hat{a}\left(t^{\prime}\right)+\hat{b}\left(t^{\prime}\right)+\delta=: \bar{e}_{e}\left(t^{\prime}\right) \tag{4.30}
\end{align*}
$$

(The bound (4.30) accounts for effects of noise, undermodelling and unknown initial conditions. Following the terminology introduced in (Hakvoort, 1994), this will be referred to collectively as the extended noise bound.) For every $t \in \mathbb{T},(4.30)$ corresponds to two linear constraints on $\theta_{t r}$ : the upper and lower bound. The model parameters in $\boldsymbol{\theta}$ should obviously obey the same constraints.

If the truncation value $n$ would not have been introduced, or if it is taken equal to $\bar{k}$, the only extension of the noise bound is actually through the value of $\delta$. It is clear that this must be a very conservative bound. It does not depend on $t$ for example, so that the influence of unknown initial conditions does not die out. The unknown initial conditions that are accounted for by $\hat{b}$ do die out for increasing $t$. Also the inputs that are known are not used in $\delta$, except for possibly increasing the value of $u_{m}$.

For multiple inputs, each input gives a separate $\hat{a}(t), \hat{b}(t)$ and $\delta$. These contributions should all be added to $\bar{e}(t)$ to yield the proper $\bar{e}_{e}(t)$. For multiple outputs, the calculations should be carried out for each output separately. This is possible, because transfers to different outputs are parametrised independently.

Example (Continued from page 106): Table 4.1 gives an excerpt of the values of the signals $\left[B_{k}(\zeta) u^{a}\right](t)$, i.e. the values of the input signals of data set $a$ after filtering by the $k$ th basis function, where $k=0,1, \ldots, 9$. From this table follows, that the following constraints on the model parameters are part of $\mathcal{L}_{\boldsymbol{\theta}}^{a}$ :

$$
\begin{aligned}
0.0163 \cdot \theta_{0}+0.4504 \cdot \theta_{1}+\cdots-0.3382 \cdot \theta_{9} & \leq-0.2358+3.1364 \\
0.0163 \cdot \theta_{0}+0.4504 \cdot \theta_{1}+\cdots-0.3382 \cdot \theta_{9} & \geq-0.2358-3.1364 \\
-1.2871 \cdot \theta_{0}+0.4330 \cdot \theta_{1}+\cdots-0.8529 \cdot \theta_{9} & \leq-0.1864+3.1364 \\
-1.2871 \cdot \theta_{0}+0.4330 \cdot \theta_{1}+\cdots-0.8529 \cdot \theta_{9} & \geq-0.1864-3.1364 \\
-1.5533 \cdot \theta_{0}+0.0095 \cdot \theta_{1}+\cdots-1.3874 \cdot \theta_{9} & \leq-1.4514+3.1375 \\
-1.5533 \cdot \theta_{0}+0.0095 \cdot \theta_{1}+\cdots-1.3874 \cdot \theta_{9} & \geq-1.4514-3.1375
\end{aligned}
$$

(Continued on page 119.)

Table 4.1: Filtered inputs, output and extended noise bound

| $t$ | $\left[B_{0}(\zeta) u\right](t)$ | $\left[B_{1}(\zeta) u\right](t)$ | $\ldots$ | $\left[B_{9}(\zeta) u\right](t)$ | $y(t)$ | $e_{e}(t)$ |
| ---: | ---: | ---: | :--- | ---: | ---: | ---: |
| 400 | 0.0163 | 0.4504 | $\ldots$ | -0.3382 | -0.2358 | 3.1364 |
| 401 | -1.2871 | 0.4330 | $\ldots$ | -0.8529 | -0.1864 | 3.1364 |
| 402 | -1.5533 | 0.0095 | $\ldots$ | -1.3874 | -1.4514 | 3.1375 |

### 4.10.2 Extended cross-covariance bound

| Agonithen |  |
| :---: | :---: |
| - De mbetments |  |
| Csthen phor tumats |  |
|  |  |
| Estmate atamay mo |  |
| - Examate arme atment |  |
| - Construct set of linear constraints . . . . . . . . . . . . . . . . . . . p. 100 |  |
| Feili, $\bar{z}_{k}, M, p, b_{k}(t)$ |  |
|  |  |
| - Extend cross-covaiance bound |  |
| - $<\bar{c}_{\text {erer }}, \underline{e}(t)$ : (unextended) cross-covariauce bound | p. 83 |
| $\rightarrow$ d(0) basis function $\bar{k}+1$ to $\hat{k}$, known inputs | p. 111 |
| $\rightarrow \mathrm{fl}$ ( ) : basis fuaction $\bar{k}+1$ to $\bar{k}$, inputs u(-i) to w $(-1)$ | p. 111 |
| $\rightarrow$ d $_{1}(\hat{)}$ : basis furiction $k+1$ to $k$, | p. 111 |
| $\rightarrow \delta_{2}(\ell)$ basis finction $\dot{k}+1$ and higher, all inputs $\longrightarrow\left\|\sum_{t} r_{l}(t)\left(y(t)-G_{\theta}(O) u(t)\right)\right\|$ | p. 111 |
| $\leq d(\ell)+f(\theta)+b_{1}(\hat{l})+b_{2}(\ell)+\dot{c}_{\sim} \sqrt{T^{4}}$ set of linear coustraints on $\theta$ for every operating | 1.112 |
|  |  |
| Grumber men |  |

For the cross-covariance bound (4.9) we have the same problems as for the noise bound (4.7): the model contains only a finite number of basis functions and we do not have access to inputs prior to $t=0$. The solution is also similar.

Consider first the single input, single output case. Choose truncation values $\tilde{k} \geq \bar{k}$ and $\tilde{t} \in \mathbb{N}$. Then

$$
\begin{align*}
& \sum_{t=0}^{T^{d}-1} \epsilon(t) r_{\ell}(t)=\sum_{t=0}^{T^{d}-1} y(t) r_{\ell}(t)-\sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{k=0}^{\infty} \theta_{t r, k} \sum_{\tau=0}^{\infty} b_{k}(\tau) u(t-\tau) \\
& \quad=\sum_{t=0}^{T^{d}-1} y(t) r_{\ell}(t)-\sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=0}^{t} b_{k}(\tau) u(t-\tau) \\
& \quad-\sum_{k=\tilde{k}+1}^{\hat{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=0}^{t} b_{k}(\tau) u(t-\tau) \\
& \quad-\sum_{k=0}^{k} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=t+1}^{t+\tilde{t}} b_{k}(\tau) u(t-\tau) \\
& \quad-\sum_{k=0}^{\tilde{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=t+\tilde{t}+1}^{\infty} b_{k}(\tau) u(t-\tau) \\
& \quad-\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=0}^{\infty} b_{k}(\tau) u(t-\tau) \tag{4.31}
\end{align*}
$$

The various terms of the right hand side will be bounded similarly to the situation in the previous section. The third term is treated first:

$$
\begin{align*}
\mid \sum_{k=\tilde{k}+1}^{\hat{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) & \sum_{\tau=0}^{t} b_{k}(\tau) u(t-\tau) \mid \leq \\
& \leq \sum_{k=\bar{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|\sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{r=0}^{t} b_{k}(\tau) u(t-\tau)\right|=: d(\ell) . \tag{4.32}
\end{align*}
$$

For the fourth it is found

$$
\begin{array}{r}
\left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=t+1}^{t+\bar{t}} b_{k}(\tau) u(t-\tau)\right| \leq \\
\leq \sum_{k=0}^{\bar{k}} \bar{\theta}_{k}\left|\sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=t+1}^{t+\tilde{t}} b_{k}(\tau) u(t-\tau)\right| \leq \sum_{k=0}^{\bar{k}} \bar{\theta}_{k} \sum_{\tau=1}^{\bar{t}}\left|\sum_{t=0}^{T^{d}-1} r_{\ell}(t) b_{k}(t+\tau) \bar{u}\right| \\
=\bar{u} \sum_{k=0}^{k} \bar{\theta}_{k} \sum_{\tau=1}^{\bar{t}}\left|\sum_{t=0}^{T^{d}-1} r_{\ell}(t) b_{k}(t+\tau)\right|=f(\ell) \tag{4.33}
\end{array}
$$

and for the fifth

$$
\begin{align*}
& \left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=t+\tilde{t}+1}^{\infty} b_{k}(\tau) u(t-\tau)\right| \leq \\
& \leq \sum_{k=0}^{\bar{k}} \bar{\theta}_{k} \sum_{t=0}^{T^{d}-1}\left|r_{\ell}(t) \sum_{\tau=t+\bar{t}+1}^{\infty} b_{k}(\tau) u(t-\tau)\right| \leq \sum_{k=0}^{\tilde{k}} \bar{\theta}_{k} \sum_{t=0}^{T^{d}-1}\left|r_{\ell}(t)\right| \sum_{\tau=t+\tilde{t}+1}^{\infty}\left|b_{k}(\tau)\right| \bar{u} \\
& =\bar{u} \sum_{k=0}^{\bar{k}} \bar{\theta}_{k} \sum_{t=0}^{T^{d}-1}\left|r_{\ell}(t)\right|\left(\left\|B_{k}(\zeta)\right\|_{\ell_{1}}-\sum_{\tau=0}^{t+\tilde{t}}\left|b_{k}(\tau)\right|\right)=: \delta_{1}(\ell) \tag{4.34}
\end{align*}
$$

The sixth finally can be bounded by

$$
\begin{array}{r}
\left|\sum_{k=\bar{k}+1}^{\infty} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=0}^{\infty} b_{k}(\tau) u(t-\tau)\right| \leq \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \sum_{t=0}^{T^{d}-1}\left|r_{\ell}(t)\right| \sum_{\tau=0}^{\infty}\left|b_{k}(\tau)\right| u_{m} \\
\leq u_{m} \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left(c_{1}^{\prime}+c_{2}^{\prime} \sqrt{k}\right) \sum_{t=0}^{T^{d}-1}\left|r_{\ell}(t)\right|=: \delta_{2}(\ell), \tag{4.35}
\end{array}
$$

where $u_{m}=\max \{\bar{u},|u(0)|,|u(1)|, \ldots,|u(\bar{k})|\}$ and $c_{1}^{\prime}$ and $c_{2}^{\prime}$ are chosen according to (4.29). We already had the prior knowledge that

$$
\begin{equation*}
\frac{1}{\sqrt{T^{d}}}\left|\sum_{t=0}^{T^{d}} \epsilon(t) r_{\ell}(t)\right| \leq \bar{c}_{\ell} \tag{4.36}
\end{equation*}
$$

Rearranging (4.31) and substituting (4.32) - (4.36) gives

$$
\begin{align*}
&\left|\sum_{t=0}^{T^{d}} y(t) r_{\ell}(t)-\sum_{t=0}^{T^{d}} r_{\ell}(t) \sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{\tau=0}^{i} b_{k}(\tau) u(t-\tau)\right| \leq \\
& \leq d(\ell)+f(\ell)+\delta_{1}(\ell)+\delta_{2}(\ell)+\bar{c}_{\ell} \sqrt{T^{d}} \tag{4.37}
\end{align*}
$$

This is the extended cross-covariance bound for a single input. In the multiple input case, every input gives a separate $d(\ell), f(\ell), \delta_{1}(\ell)$ and $\delta_{2}(\ell)$ that should be added to the cross-covariance bound to obtain the extended cross-covariance bound. In case of multiple outputs, calculations have to be repeated for each of the outputs. Here the independent parametrisation of different outputs is used again. This gives finally for every signal $r_{\ell}^{d}(t), \ell \in \boldsymbol{n}_{r}^{d}$ and for every output two linear constraints on the model parameters.

### 4.10.3 Estimating prior information

This section is similar to section 4.8.3. The objective is now to find $c_{1}^{\prime}$ and $c_{2}^{\prime}$ for all $\sigma \in \mathfrak{S}$, solving

$$
\begin{equation*}
\operatorname{minimise} \sum_{k=0}^{\vec{k}}\left|c_{1}^{\prime}+c_{2}^{\prime} \sqrt{k}-\left\|B_{k}(\zeta)\right\|_{\ell_{1}}\right| \quad \text { for } c_{1}^{\prime}, c_{2}^{\prime} \tag{4.38}
\end{equation*}
$$

subject to

$$
\begin{gather*}
\forall k=0, \ldots, \bar{k} \quad c_{1}^{\prime}+c_{2}^{\prime} \sqrt{k} \geq\left\|B_{k}(\zeta)\right\|_{\ell_{1}}  \tag{4.39}\\
c_{1}^{\prime} \geq 0, c_{2}^{\prime} \geq 0 \tag{4.40}
\end{gather*}
$$

The resulting $c_{1}^{\prime}$ and $c_{2}^{\prime}$ can then be used as estimated values for $c_{\mathrm{T}}^{\prime}$ and $c_{2}^{\prime}$ in (4.29).

The rationale behind this optimisation is similar to that of (4.17). The absolute value can be dropped from (4.38) because of (4.39). $\bar{k}$ will in general be taken equal to $\bar{k}^{\sigma}$, though any other sufficiently large value can be used instead. As in section 4.8.3, this optimisation has to be performed for every subtransfer in $\mathcal{S}$.

### 4.11 Step 9: bounding the local structured error



With local structured error is meant the components $\delta_{i} A_{i}, i \in \boldsymbol{n}_{s}$ in (4.4), considered in an operating point $w \in \mathcal{W}$. Local lower and upper bounds on $\delta_{i}$ will be derived. These will be denoted $d_{i}^{w}$ and $\bar{d}_{i}^{w}$, respectively. The local errors will be combined in section 4.13 to the global bounds $\underline{d}_{i}$ and $\bar{d}_{i}$ of (4.4).

Let $\hat{\boldsymbol{\theta}}$ contain those expansion coefficients of the nominal model $\hat{G}$ that correspond to basis functions incorporated in the model parametrisation. This does not contradict our earlier claim that $\hat{G}$ need not be expressed in the parametrisation of section 4.3.2. It is still allowed that

$$
G_{\hat{\theta}} \not \equiv \hat{G}
$$

The expansion coefficients corresponding to the tail of the basis need not be zero for $\hat{G}$.

In figure 4.10 a somewhat simplified situation is drawn. The parameter vector consists of two elements only. The two unit vectors lie along the axes of figure 4.10 . The set of linear constraints $\mathcal{L}_{\theta}^{w}$ is represented by a polytope containing all points that satisfy the constraints. There is one structured error component having a parameter vector $\theta_{A, 1}$. In this two dimensional example, the orthoplement of $\boldsymbol{\theta}_{A, 1}$ can be spanned by a single vector $\boldsymbol{\theta}_{A, 1}^{\perp}$. In practical cases, the orthoplement will be of higher dimension, corresponding to a hyper plane in the parameter space.

It is known through the construction of $\mathcal{L}_{\theta}^{w}$ that the true expansion coefficients of $G_{t r}$ (or rather those that are incorporated in the model parametrisation) lie within $\mathcal{L}_{\theta}^{w}$. For an arbitrary model in $\mathcal{L}_{\theta}^{w}$, denoted by $\boldsymbol{\theta}^{\prime}$, the difference between this model and the nominal model $\hat{\theta}$ can be decomposed into a component along the structured error component $\boldsymbol{\theta}_{A, 1}$, corresponding to the vector $\boldsymbol{\theta}_{\|}^{\prime}$, and a component perpendicular to $\boldsymbol{\theta}_{A, 1}$, indicated by the vector $\boldsymbol{\theta}_{\perp}^{\prime}$. The


Figure 4.10: Bounding of structured error components
vector $\boldsymbol{\theta}_{\|}^{\prime}$ is given by the equation

$$
\boldsymbol{\theta}_{\|}^{\prime}=\frac{\left(\boldsymbol{\theta}_{A, 1} \mid \boldsymbol{\theta}^{\prime}-\boldsymbol{\theta}^{c}\right)}{\left\|\boldsymbol{\theta}_{A, 1}\right\|^{2}} \boldsymbol{\theta}_{A, 1}=\left(\boldsymbol{\theta}_{A, 1} \mid \boldsymbol{\theta}^{\prime}-\boldsymbol{\theta}^{c}\right) \boldsymbol{\theta}_{A, 1}
$$

As the model parametrisation is linear in the parameters, this means that the model $G_{\theta^{\prime}}$ can be written as

$$
G_{\theta^{\prime}}=\hat{G}+\left(\theta_{A, 1} \mid \boldsymbol{\theta}^{\prime}-\theta^{c}\right) A_{1}(\zeta)+\Delta(\zeta)
$$

where $\Delta(\zeta)$ accounts for uncertainty in the direction perpendicular to $\boldsymbol{\theta}_{A, 1}$ and for uncertainty due to the tail of the basis. See also remark 4.9. As for any $\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{\boldsymbol{\theta}}$, the gain factor for the structured error component is equal to $\left(\boldsymbol{\theta}_{A, 1} \mid \boldsymbol{\theta}^{\prime}-\boldsymbol{\theta}^{c}\right)$, the largest gain factor that can be encountered for a $\boldsymbol{\theta}^{\boldsymbol{\theta}} \in \mathcal{L}_{\boldsymbol{\theta}}^{\boldsymbol{w}}$ is

$$
\begin{equation*}
-\left(\hat{\boldsymbol{\theta}} \mid \boldsymbol{\theta}_{A, 1}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}^{\prime} \mid \boldsymbol{\theta}_{A, 1}\right)=: \bar{d}_{1}^{w} \tag{4.41}
\end{equation*}
$$

If there are more than one structured error component, the bounds on the gain factor for the $i$ th structured error component, $i \in n_{s}$, can be found by substituting $\boldsymbol{\theta}_{A, i}$ for $\boldsymbol{\theta}_{A, 1}$ in (4.41). This is possible due to the orthogonality of the $\boldsymbol{\theta}_{A, i}$. An upper bound on $\delta_{i}$ is thus given locally by

$$
\begin{equation*}
\forall i \in \boldsymbol{n}_{s}, w \in \mathcal{W} \quad \bar{d}_{i}^{w}:=-\left(\hat{\boldsymbol{\theta}} \mid \boldsymbol{\theta}_{A, i}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}^{\prime} \mid \boldsymbol{\theta}_{A, i}\right) \tag{4.42}
\end{equation*}
$$

For all $i$ and $w$, this constitutes a linear programming problem.
The lower bound $\underline{d}_{i}^{w}, i \in n_{s}, w \in \mathcal{W}$ can be found completely analogously:

$$
\begin{equation*}
\forall i \in \boldsymbol{n}_{\boldsymbol{s}}, w \in \mathcal{W} \underline{d}_{i}^{w}:=-\left(\hat{\boldsymbol{\theta}} \mid \boldsymbol{\theta}_{A, i}\right)+\min _{\theta^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}^{\prime} \mid \boldsymbol{\theta}_{A, i}\right) \tag{4.43}
\end{equation*}
$$

In case the $\boldsymbol{\theta}_{A, i}$ are given by the user, they may not constitute an orthonormal set of vectors. This situation will be treated in section 5.7.

Remark 4.8 In the previous derivation, a scaling factor in parameter space could be used as a gain factor. As was mentioned, this was possible because of the linear parametrisation of the model. This is the most stringent reason why the model should be linearly parametrised.

Remark 4.9 In the example given in figure 4.10 , it was mentioned that $\Delta(\zeta)$ accounted for the uncertainty in the tail of the basis. The tail of the model $G_{\theta^{\prime}}$ is equal to zero. Therefore this uncertainty amounts in this particular case to $\hat{G}-G_{\hat{\boldsymbol{\theta}}}$. For the true system, the tail is in general not equal to zero.

Remark 4.10 If $0 \notin\left[\underline{d}_{i}, \bar{d}_{i}\right]$ for some $i \in n_{s}$, the nominal model is not consistent with the data and the prior knowledge. The nominal model is only consistent with the data and prior knowledge if $\delta_{i}=0$ for all $i \in \boldsymbol{n}_{\boldsymbol{s}}$ and $\Delta=0$ turns out to be an element of the uncertainty description (4.4), which is clearly not the case in the aforementioned situation. This need not be alarming, as the nominal model is supposed to "resemble" the process in all operating points. This accounting for all operating points may cause so much bias in the nominal model in an operating point that it is not consistent with the data and prior knowledge in that particular operating point. This does not imply that the nominal model is a bad approximation of all operating points.

The maxima and minima in (4.42) and (4.43) exist, provided $\mathcal{L}_{\theta}^{w}$ is not empty: From the definition of $\mathcal{L}_{\theta}^{w}$ is clear that it is a closed set. It is also bounded, provided the data sets in $\mathcal{D}(w)$ contain sufficient samples, because the model parameters in $\mathcal{L}_{\theta}^{\psi}$ can not grow without bounds without violating sooner or later the extended noise bound (4.30). This follows from the independence of the basis functions. If the basis functions were not independent, a combination of basis functions could be compensated by another combination. Orthonormal basis functions are independent provided the time interval that is considered is long enough. Having established that $\mathcal{L}_{\theta}^{w}$ is closed and bounded, taking into account that the function to be minimised is continuous and assuming that $\mathcal{L}_{\boldsymbol{\theta}}^{w}$ is not empty, the maxima and minima exist.

If the prior knowledge is estimated according to section 4.8 .3 and 4.10 .3 and the considered time-interval is long enough in the sense explained previously, the set $\mathcal{L}_{\theta}^{w}$ is not empty, because the auxiliary model for an operating point
satisfies all constraints for that operating point. $\mathcal{L}_{\theta}^{w}$ therefore has at least one element. If prior knowledge was gathered in another way, there may be some inconsistency in the prior knowledge and experimental data. This can make $\mathcal{L}_{\theta}^{w}$ empty. The minima and maxima in (4.42) and (4.43) will not exist. This can be detected easily, as the solver for the linear programs will report an infeasible problem.

### 4.12 Step 10: bounding the local unstructured error

After bounding the structured errors locally, now the remaining unstructured errors will be bounded locally. Results will be developed for bounds in frequency domain and time domain. The frequency domain bounds are more involved than the time domain results, so the former will be used to explain the procedure in detail. The time domain procedure will then be adapted from the frequency domain procedure.

The procedure is presented first in frequency domain without considering the structured error components. These results, and the results in time domain without structured errors, come from (Hakvoort, 1994, section 4.4 and 4.5). Then will be explained how the structured error components can be removed from $\Delta$. Finally, the time domain procedure is presented.

### 4.12.1 Structure-less frequency domain procedure



The aim of this section is mainly to translate the linearly constrained set of model parameters $\mathcal{L}_{\theta}^{w}$ to uncertainty regions in the complex plane. The uncertainty regions will contain the unknown true frequency response of $\Delta$ on a grid

a) $\mathcal{P}_{\psi_{i}}$ bounded in the direction $e^{j \phi \ell}$

b) $\mathcal{P}_{\omega_{i}}$ bounded by an explicitly known polytope $\mathcal{P}_{\boldsymbol{m}}\left(\omega_{i}\right)$

Figure 4.11: Bounding $\mathcal{P}_{\omega_{i}}$
of frequencies. For every frequency in this grid there is a separate uncertainty region. These regions can be translated to a (weighted) $\mathcal{H}_{\infty}$-bound on $\Delta$ by bounding the interpolation error between consecutive frequencies on the grid.

The procedure discussed here works only for SISO transfers. Therefore it has to be repeated for every sub-transfer of a MIMO transfer. Every element in $\Delta$ is consequently bounded separately. For notational convenience, superscript $w$ for the operating point and $\sigma$ for the sub-transfer will be dropped in this section.

## Frequency response uncertainty regions on a frequency grid

Let $\Omega:=\left\{\omega_{1}, \ldots, \omega_{n_{\boldsymbol{\Omega}}}\right\}$ where $n_{\Omega} \in \mathbb{Z}_{+}$and $\omega_{i} \in \mathbb{R}, i \in \boldsymbol{n}_{\boldsymbol{\Omega}}$. Consider the polytope in complex plane

$$
\mathcal{P}_{\omega_{i}}:=\left\{G_{\theta}\left(e^{j \omega_{i}}\right)-\hat{G}\left(e^{j \omega_{i}}\right) \mid \boldsymbol{\theta} \in \mathcal{L}_{\theta}\right\}
$$

This polytope is completely determined by the model parametrisation, $\hat{G}, \mathcal{L}_{\theta}$ and $\omega_{i}$, but unfortunately finding all vertices of $\mathcal{P}_{\omega_{i}}$ would require traversing all vertices of $\mathcal{L}_{\theta}$. This involves a computational load that gets out of hand very quickly.

However, $\mathcal{P}_{\omega_{i}}$ can be bounded in the direction $e^{j \phi_{\ell}}$ by calculating

$$
\begin{equation*}
\hat{\mu}_{\ell}\left(\omega_{i}\right):=\max _{\theta \in \mathcal{L}_{\boldsymbol{\theta}}} \operatorname{Re}\left(\left(G_{\theta}\left(e^{j \omega_{i}}\right)-\hat{G}\left(e^{j \omega_{i}}\right)\right) e^{-j \phi_{\ell}}\right) \tag{4.44}
\end{equation*}
$$

Once again, this is a linear programming problem. Figure 4.11a depicts this situation. The dashed line indicates the direction $e^{j \phi_{\ell}}$. Because obviously for $x \in \mathcal{L}_{\boldsymbol{\theta}}$

$$
\operatorname{Re}\left(\left(G_{x}\left(e^{j \omega_{i}}\right)-\hat{G}\left(e^{j \omega_{i}}\right)\right) e^{-j \phi_{t}}\right) \leq \hat{\mu}_{\ell}\left(\omega_{i}\right)
$$

the polytope $\mathcal{P}_{\omega_{i}}$ contains no points in the half plane indicated by the shaded area. This procedure can be repeated for a number of directions $\phi \ell, \ell \in m$. The intersection of all half planes that do contain elements of $\mathcal{P}_{\omega_{i}}$ is again a polytope, but this polytope has only a number of vertices less than or equal to the number of directions $\phi_{\ell}$ that have been investigated. Such polytope will be denoted $\mathcal{P}_{m}\left(\omega_{i}\right)$. Figure 4.11b shows a situation in which $m=8$, leading to an outer bounding polytope with 8 vertices. In practice, if a number $m$ of directions is investigated ( $m \geq 3$ ), $\phi_{\ell}$ is taken equal to

$$
\phi_{\ell}=e^{j 2(\ell-1) \pi / m}, \quad \ell \in m
$$

The polytope $\mathcal{P}_{\omega_{i}}$ does not necessarily contain the true response of $\Delta$ : the tail of the basis has not been taken into account. Choosing a truncation value $\tilde{k} \geq \bar{k}$, the contribution of the tail in a direction $\phi_{\ell}$ can be bounded as follows:

$$
\begin{align*}
& \left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right| \leq \\
& \leq\left|\sum_{k=\tilde{k}+1}^{k} \theta_{t r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right|+\left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right| \leq \\
& \quad \leq \sum_{k=\bar{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|B_{k}\left(e^{j \omega_{i}}\right)\right|+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k} \max _{k>\bar{k}}\left|B_{k}\left(e^{j \omega_{i}}\right)\right|=: \bar{\mu}\left(\omega_{i}\right) \tag{4.45}
\end{align*}
$$

$\max _{k>\bar{k}}\left|B_{k}\left(e^{j \omega_{i}}\right)\right|$ can be calculated easily because of property 4.1.
The true response of $\Delta$ can now be bounded to a polytope $\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$, characterised as follows:

$$
\begin{equation*}
\overline{\mathcal{P}}_{m}\left(\omega_{i}\right):=\left\{z \in \mathbb{C} \mid \forall \ell \in \boldsymbol{m} \quad \operatorname{Re}\left(z e^{-j \phi_{\ell}}\right) \leq \hat{\mu}_{\ell}\left(\omega_{i}\right)+\bar{\mu}\left(\omega_{i}\right)\right\} \tag{4.46}
\end{equation*}
$$

Remark 4.11 A possibly tighter upper bound can be obtained instead of (4.45) by using

$$
\begin{align*}
& \left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{k}}\right)\right| \leq \\
& \leq\left|\sum_{k=\bar{k}+1}^{\tilde{k}} \theta_{i r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right|+\left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right| \leq \\
& \quad \leq \sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|\operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right|+\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \max _{k>\tilde{k}}\left|B_{k}\left(e^{j \omega_{i}}\right)\right| \tag{4.47}
\end{align*}
$$

For consistency with later uses of $\bar{\mu}\left(\omega_{i}\right)$, this is not used here.
Remark 4.12 We deviate slightly from (Hakvoort, 1994) in (4.45) by taking $\max _{k>\tilde{k}}\left|B_{k}\left(e^{j \omega_{i}}\right)\right|$ instead of $\max _{k>\bar{k}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}}$. This is less conservative and easier to calculate.

Example (Continued from page 109): In table 4.2 the transfer function of the nominal model $\hat{G}$ and some of the basis functions used in the example are listed for $\omega^{\prime}=0.01166$.

Table 4.2: Some transfer function values for $\omega^{\prime}=0.01166$

| $f$ | $f\left(e^{j \omega^{\prime}}\right)$ |
| :---: | :---: |
| $B_{0}$ | 1.0000 |
| $B_{1}$ | $5.9302-1.3847 j$ |
| $\vdots$ | $\vdots$ |
| $B_{9}$ | $-4.7933+3.7562 j$ |
| $\hat{G}$ | $27.1152-11.4840 j$ |

Suppose now $m=8$, i.e. the outer bounding polytopes have at most eight vertices. For the set $\left\{e^{-j \phi_{\ell}}\right\}_{\ell=1}^{8}$ is taken $\left\{1, \frac{1-j}{\sqrt{2}},-j, \frac{-1-j}{\sqrt{2}},-1, \frac{-1+j}{\sqrt{2}}, j, \frac{1+j}{\sqrt{2}}\right\}$. The linear programming problems that need to be solved to determine $\mathcal{P}_{m}\left(\omega^{i}\right)$ are
maximise

$$
\begin{array}{r}
1.0000 \cdot \theta_{0}+5.9302 \cdot \theta_{1}+\cdots-4.7933 \cdot \theta_{9}-27.1152 \\
0.7071 \cdot \theta_{0}+3.2142 \cdot \theta_{1}+\cdots-0.7333 \cdot \theta_{9}-11.0529 \\
-1.3847 \cdot \theta_{1}+\cdots+3.7562 \cdot \theta_{9}+11.4840 \\
-0.7071 \cdot \theta_{0}-5.1724 \cdot \theta_{1}+\cdots+6.0454 \cdot \theta_{9}+27.2937 \\
-1.0000 \cdot \theta_{0}-5.9302 \cdot \theta_{1}+\cdots+4.7933 \cdot \theta_{9}+27.1152 \\
-0.7071 \cdot \theta_{0}-3.2142 \cdot \theta_{1}+\cdots+0.7333 \cdot \theta_{9}+11.0529 \\
1.3847 \cdot \theta_{1}+\cdots-3.7562 \cdot \theta_{9}-11.4840
\end{array}
$$

and

$$
0.7071 \cdot \theta_{0}+5.1724 \cdot \theta_{1}+\cdots-6.0454 \cdot \theta_{9}-27.2937
$$

subject to

$$
\left[\theta_{0}, \theta_{1}, \ldots, \theta_{9}\right]^{T} \in \mathcal{L}_{\boldsymbol{\theta}}
$$

The solutions to these linear programming problems provide the values for $\hat{\mu}_{\ell}\left(\omega^{\prime}\right), \ell \in m$. Calculation of (4.45) or alternatively (4.47) yields values for
$\bar{\mu}\left(\omega^{\prime}\right)$. The results of these calculations are shown in figure 4.12. The arrows in this figure indicate the direction $e^{j \phi_{\ell}}, \ell \in m$. The length of an arrow is equal to the corresponding value of $\hat{\mu}_{\ell}\left(\omega^{t}\right)+\bar{\mu}\left(\omega^{\prime}\right)$. The polytope $\overline{\mathcal{P}}_{m}\left(\omega^{\prime}\right)$ is determined from these arrows, as indicated in figure 4.11 on page 117.
(Continued on the next page.)

## Weighting, frequency domain interpolation error

The frequency response uncertainty regions $\overline{\mathcal{P}}_{m}(\omega), \omega \in \Omega$ will now be translated to an $\mathcal{H}_{\infty}$-norm bound on the weighted unstructured error. In other words, a value $d_{\Delta}$ is sought, such that

$$
\begin{equation*}
\|W(\zeta) \Delta(\zeta)\|_{\mathcal{H}_{\infty}} \leq d_{\Delta} \tag{4.48}
\end{equation*}
$$

where $W(\zeta) \in \mathcal{H}_{\infty}$ is a known, stable weighting function.
The bound (4.48) introduces minimal conservatism if $W(\zeta)$ is chosen such that $\left|W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)\right|$ is an approximately constant function of $\omega$, equal to $d_{\Delta}$. The maximal value of $\left|\Delta\left(e^{j \omega}\right)\right|$ is already known for $\omega \in \Omega$ from the results of the previous section. One can try and fit $\left|W\left(e^{j \omega}\right)\right|^{-1}$ to these values for $\omega \in \Omega$, while still obeying that $W(\zeta) \in \mathcal{H}_{\infty}$. This problem will not be discussed further.

Let $\left\{v_{k}\left(\omega_{i}\right)\right\}_{k=1}^{m}$ denote the set of vertices of the polytope $\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$. For simplicity we assume that this polytope has $m$ vertices. In case this polytope has less than $m$ vertices, only trivial modifications need to be made. Because


Figure 4.12: Determining $\overline{\mathcal{P}}_{m}\left(\omega^{\prime}\right)$ from $\hat{\mu}_{\ell}\left(\omega^{\prime}\right)+\bar{\mu}\left(\omega^{\prime}\right)$
$\Delta\left(\omega_{i}\right) \in \overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$ it holds

$$
\begin{equation*}
\forall \omega \in \Omega \quad\left|W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)\right| \leq \max _{k \in m}\left|W\left(e^{j \omega}\right) v_{k}(\omega)\right|=: \delta_{u}(\omega) \tag{4.49}
\end{equation*}
$$

This bounds the value of $W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)$ on the frequency grid $\Omega$.
At this point it is interesting to consider the situation that for some $\omega_{i} \in \Omega$ it holds $0 \notin \overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$. This means that the nominal model is not consistent with the data and the prior knowledge. This does not mean that the nominal model is useless. It does mean that the frequency domain uncertainty bound will be very conservative for this frequency, as only the largest distance to the origin of the polytope $\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$ is retained. This situation is drawn in figure 4.13 , where for simplicity $W$ has been taken equal to the identity. The darkest shaded


Figure 4.13: Conservatism in the frequency domain bound
area is the polytope $\mathcal{P}_{\omega_{i}}$. This corresponds to the set of linear constraints $\mathcal{L}_{\theta}$ augmented with a bound for the effect of the tail of the basis. This possibly complicated polytope is outer bounded by a polytope $\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$ where, in the situation of figure $4.13, m=8$, which is represented by the lighter shaded area. The information in this polytope is finally summarised to its largest distance to the origin, indicated by the light shaded disk.

Example (Continued from the facing page): The vertices $\left\{v_{k}\left(\omega^{\prime}\right)\right\}$ are indicated in figure 4.12 by a ' $o$ '. The vertex having the largest distance to the origin, i.e. the one determining $\delta_{u}\left(\omega^{\prime}\right)$, is indicated by an additional '+'. Note
that the vertex determining $\delta_{u}\left(\omega^{\prime}\right)$ is independent of the value of $W\left(e^{j \omega^{\prime}}\right)$.
(Continued on page 129.)
The bound (4.49) is extended to a bound for all frequencies by interpolating between the frequencies in $\Omega$. Basically a bound $\beta$ on the derivative of $W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)$ is derived, such that

$$
\left|\frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}\right| \leq \beta
$$

This is achieved by theorem 4.4 .8 of (Hakvoort, 1994). The theorem is restated here to facilitate the exposition of the adaptations that need to be made later to account for the structured error components. The basis of this theorem is the observation, that

$$
\begin{align*}
& \frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}= \\
& \quad \frac{d W\left(e^{j \omega}\right)}{d \omega} G_{t r}\left(e^{j \omega}\right)+W\left(e^{j \omega}\right) \frac{d G_{t r}\left(e^{j \omega}\right)}{d \omega}-\frac{d W\left(e^{j \omega}\right) \hat{G}\left(e^{j \omega}\right)}{d \omega} \tag{4.50}
\end{align*}
$$

from which follows through elementary manipulations that

$$
\begin{align*}
\forall \omega \in \mathbb{R} & \left|\frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}\right| \leq\left\|\frac{d W(z) \hat{G}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}+ \\
& +\left\|\frac{d W(z)}{d z}\right\|_{\mathcal{H}_{\infty}}\left\|G_{t r}(z)\right\|_{\mathcal{H}_{\infty}}+\|W(z)\|_{\mathcal{H}_{\infty}}\left\|\frac{d G_{t r}(z)}{d z}\right\|_{\mathcal{H}_{\infty}} \tag{4.51}
\end{align*}
$$

Supposing for a moment that values $\beta_{1}$ and $\beta_{2}$ are available such that

$$
\left\|\frac{d G_{t r}(z)}{d z}\right\|_{\mathcal{H}_{\infty}} \leq \beta_{1} \quad \text { and } \quad\left\|G_{t r}(z)\right\|_{\mathcal{H}_{\infty}} \leq \beta_{2}
$$

(4.51) leads to

$$
\begin{align*}
\forall \omega \in \mathbb{R} & \left|\frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}\right| \leq \\
& \leq \left\lvert\, \frac{d W(z) \hat{G}(z)}{d z}\left\|_{\mathcal{H}_{\infty}}+\right\| \frac{d W(z)}{d z}\left\|_{\mathcal{H}_{\infty}} \beta_{2}+\right\| W(z)\right. \|_{\mathcal{H}_{\infty}} \beta_{1}=; \beta \tag{4.52}
\end{align*}
$$

In the remainder of this section computable expressions for $\beta_{1}$ and $\beta_{2}$ will be derived.

Figure 4.14 depicts a situation in which 5 frequencies have been taken in $\Omega$. The dots indicate the corresponding values of $\delta_{u}(\omega)$. The dash-dotted line is


Figure 4.14: Worst case interpolating functions
taken as a "worst case interpolating line." The phrase "interpolating" may be a bit misplaced considering the discontinuities in this interpolation. The slopes of the dash-dotted lines correspond to the bound $\beta$. The solid line which is drawn slightly offset could be used as an alternative interpolating function. As can be seen from figure 4.14 , this may give a smaller interpolation error, at the expense of a somewhat more involved calculation.
$\lambda_{j}, j \in \boldsymbol{n}_{\mathbf{\Omega}}$ are the distances between subsequent frequencies in $\Omega$ :

$$
\begin{aligned}
\lambda_{1} & : \\
\lambda_{j} & :=\max \left\{2 \omega_{1}, \omega_{2}-\omega_{1}\right\} \\
\lambda_{n_{\Omega}} & \left.:=\max \left\{\omega_{j}-\omega_{j-1}, \omega_{j+1}-\omega_{j}\right\}, \quad j=2, \ldots, \omega_{n_{\Omega}-1}, 2\left(\pi-\omega_{n_{\Omega}}\right)\right\}
\end{aligned}
$$

Using the dash-dotted lines as interpolation we find

$$
\forall \omega \in \mathbb{R} \quad\left|W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)\right| \leq \max _{i \in n_{\Omega}} \delta_{u}\left(\omega_{i}\right)+\frac{1}{2} \lambda_{i} \beta,
$$

from which follows straightforwardly

$$
\begin{equation*}
\|W(\zeta) \Delta(\zeta)\|_{\mathcal{H}_{\infty}} \leq \max _{i \in n_{\Omega}} \delta_{u}\left(\omega_{i}\right)+\frac{1}{2} \lambda_{i} \beta \tag{4.53}
\end{equation*}
$$

Remains the problem of finding $\beta_{1}$ and $\beta_{2}$. A value for $\beta_{1}$ can be found as
follows. Choose a truncation value $\tilde{k}$ :

$$
\begin{aligned}
\forall \omega \in \mathbb{R}\left|\frac{d G_{t r}\left(e^{j \omega}\right)}{d \omega}\right| & \leq \sum_{k=0}^{\infty} \theta_{t r, k}\left\|\frac{d B_{k}(z)}{d z}\right\|_{\mathcal{H}_{\infty}} \leq \\
& \leq \sum_{k=0}^{\bar{k}} \bar{\theta}_{k}\left\|\frac{d B_{k}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left\|\frac{d B_{k}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}
\end{aligned}
$$

The last term can be bounded using conjecture 4.2 and the fact for any transfer $G(z)$ it holds $\|G(z)\|_{\mathcal{H}_{\infty}} \leq\|G(z)\|_{\ell_{1}}$ :

$$
\begin{aligned}
\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left\|\frac{d B_{k}(z)}{d z}\right\|_{\mathcal{H}_{\infty}} & \leq \sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left\|\frac{d \sum_{\tau=0}^{\infty} b_{k}(\tau) z^{-\tau}}{d z}\right\|_{\ell_{1}}= \\
& =\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \sum_{\tau=0}^{\infty} \tau\left|b_{k}(\tau)\right| \leq \sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left(c_{1}+c_{2} k \sqrt{k}\right)
\end{aligned}
$$

Here $c_{1}$ and $c_{2}$ have been chosen such that

$$
\sum_{\tau=0}^{\infty} \tau\left|b_{k}(\tau)\right| \leq c_{1}+c_{2} k \sqrt{k}
$$

The values of $c_{1}$ and $c_{2}$ can be estimated similarly to the values for $c_{1}$ and $c_{2}$ in section 4.10.3. Simple combination of results gives

$$
\beta_{1}:=\sum_{k=0}^{\bar{k}} \tilde{\theta}_{k}\left\|\frac{d B_{k}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left(c_{1}+c_{2} k \sqrt{k}\right)
$$

A value for $\beta_{2}$ can be obtained by using a procedure similar to the one used to bound $\|W(z) \Delta(z)\|_{\mathcal{H}_{\infty}}$ : Let $v_{k}^{\prime}\left(\omega_{i}\right), \omega_{i} \in \Omega$ be the vertices of the polytope

$$
\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)+\hat{G}\left(e^{j \omega_{i}}\right)
$$

and let $\delta_{u}^{\prime}\left(\omega_{i}\right)$ be defined similarly to $\delta_{u}\left(\omega_{i}\right)$ as

$$
\delta_{u}^{\prime}\left(\omega_{i}\right):=\max _{k \in m}\left|v_{k}^{\prime}\left(\omega_{i}\right)\right|,
$$

then

$$
\left\|G_{t r}\left(e^{j \omega}\right)\right\|_{\mathcal{H}_{\infty}} \leq \max _{i \in \boldsymbol{n}_{\boldsymbol{\Omega}}} \delta_{u}^{\prime}\left(\omega_{i}\right)+\frac{1}{2} \lambda_{i} \beta_{1}=: \beta_{2}
$$

### 4.12.2 Removing structured error components in frequency domain



The procedure in section 4.12 .1 did not take the structured error components into account. Obviously this should be done, as otherwise the structured error components make the process uncertainty needlessly bigger.

As all results in section 4.12 are local results, the results in this (sub)section apply in an operating point.

## Frequency response uncertainty regions on a frequency grid

The adaptation of $\hat{\mu}_{\ell}\left(\omega_{i}\right)$ as defined in (4.44) on page 117 is considered first. Recall that $\hat{\mu}_{\ell}\left(\omega_{i}\right)$ bounded the uncertainty in $G_{t r}\left(e^{j \omega_{i}}\right)$ resulting from the uncertainty in the model parameters of $G_{t r}$, in a direction $e^{j \phi_{\ell}}$. The uncertainty due to the tail of $G_{t r}$ was bounded by $\bar{\mu}\left(\omega_{i}\right)$.

Let $\delta_{t r, i}$ be defined as

$$
\begin{equation*}
\forall i \in n_{\boldsymbol{s}} \quad \delta_{t r, i}:=\left(\boldsymbol{\theta}_{A, i} \mid \boldsymbol{\theta}_{t r}-\hat{\boldsymbol{\theta}}\right) \tag{4.54}
\end{equation*}
$$

where $\hat{\boldsymbol{\theta}}$ contains the parameters of $\hat{G}$ as discussed in remark 4.9 and $\boldsymbol{\theta}_{t r}$ contains the true process parameters, i.e. the expansion coefficients of the linear manifestation of the process in the operating point that is considered (the superscripts $w$ are still dropped.) Suppose that the structured error components were known exactly. More precisely, that for each $i \in \boldsymbol{n}_{\boldsymbol{s}}$ the true value of $\delta_{i}$, given by $\delta_{t r, i}$ defined above, are known. The values of $\delta_{t r, i}$ are collected in a vector

$$
\delta_{t r}:=\left[\delta_{t r, 1}, \ldots, \delta_{t r, n_{s}}\right]^{T}
$$

For further notational convenience the values $\hat{\delta}_{i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$ are introduced

$$
\forall i \in \boldsymbol{n}_{\boldsymbol{s}} \quad \hat{\delta}_{i}:=\left(\boldsymbol{\theta}_{A, i} \mid \hat{\boldsymbol{\theta}}\right)
$$

as well as the nominal model without any components in the directions $\boldsymbol{\theta}_{A, i}$,

$$
\hat{G}^{\perp}(\zeta):=\hat{G}(\zeta)-\sum_{i=1}^{n_{s}} \hat{\delta}_{i} A_{i}(\zeta)
$$

By construction of $\mathcal{L}_{\boldsymbol{\theta}}$, the model parameters of the true system are known to lie within the polytope defined by $\mathcal{L}_{\boldsymbol{\theta}}$. If the values of $\delta_{\boldsymbol{t r}}$ are known as well, the model parameters of the true system are within the subset of $\mathcal{L}_{\theta}$ given by

$$
\begin{equation*}
\mathcal{L}_{\boldsymbol{\theta}}\left(\delta_{\boldsymbol{t r}}\right):=\left\{\boldsymbol{\theta} \in \mathcal{L}_{\boldsymbol{\theta}} \mid \forall i \in \boldsymbol{n}_{\boldsymbol{s}} \quad\left(\boldsymbol{\theta}_{\boldsymbol{A}, i} \mid \boldsymbol{\theta}-\hat{\boldsymbol{\theta}}\right)=\delta_{\boldsymbol{t r}, i}\right\} \tag{4.55}
\end{equation*}
$$

In figure 4.15 an example is drawn for three-dimensional parameter vectors $\boldsymbol{\theta}$. The set $\mathcal{L}_{\boldsymbol{\theta}}$ is indicated by a polytope. The light shaded area inside the polytope corresponds to the set above.


Figure 4.15: Removing structured error components from $\Delta$
All vectors $\boldsymbol{\theta} \in \mathcal{L}_{\boldsymbol{\theta}}\left(\delta_{t r}\right)$ have a component $\left(\delta_{t r, i}+\hat{\delta}\right) \boldsymbol{\theta}_{A, i}$ in the direction of $\boldsymbol{\theta}_{A, i}$. These components need not be accounted for by $\Delta(\zeta)$, because the part $\hat{\delta} \boldsymbol{\theta}_{A, i}$ is in the nominal model and the part $\delta_{t r, i} \boldsymbol{\theta}_{A, i}$ is taken into account by the structured error component. For any $\boldsymbol{\theta} \in \mathcal{L}_{\boldsymbol{\theta}}\left(\delta_{\boldsymbol{t r}}\right)$, only the part $\boldsymbol{\theta}-$ $\sum_{i=1}^{n_{s}}\left(\delta_{t r, i}+\hat{\delta}_{i}\right) \boldsymbol{\theta}_{A, i}$ needs to appear in $\Delta(\zeta)$. Therefore only vectors $\boldsymbol{\theta} \in \mathcal{L}_{\boldsymbol{\theta}}^{\prime}\left(\boldsymbol{\delta}_{\boldsymbol{t r}}\right)$
are of interest for the determination of $\Delta$ :

$$
\begin{align*}
& \mathcal{L}_{\boldsymbol{\theta}}^{\prime}\left(\boldsymbol{\delta}_{t r}\right):=\left\{\boldsymbol{\theta}-\sum_{i=1}^{n_{s}}\left(\delta_{t r, i}+\hat{\delta}_{i}\right) \boldsymbol{\theta}_{A, i} \mid\right. \\
& \boldsymbol{\theta}\left.\in \mathcal{L}_{\boldsymbol{\theta}} \wedge \forall i \in \boldsymbol{n}_{\boldsymbol{s}} \quad\left(\boldsymbol{\theta}_{A, i} \mid \boldsymbol{\theta}\right)=\delta_{t r, i}+\hat{\delta}_{i}\right\} \tag{4.56}
\end{align*}
$$

$\mathcal{L}_{\theta}^{\prime}\left(\delta_{t r}\right)$ has the same shape as $\mathcal{L}_{\theta}\left(\delta_{t r}\right)$, it is only translated in a direction that is a linear combination of the $\boldsymbol{\theta}_{A, i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$ such, that the translated set contains the origin.

Consequently the transfers $G_{\theta_{\Delta}}\left(e^{j \omega}\right)$ for $\boldsymbol{\theta}_{\Delta} \in \mathcal{L}_{\boldsymbol{\theta}}\left(\boldsymbol{\delta}_{\boldsymbol{t r}}\right)$ account for the uncertainty that remains in the model parameters after the structured components have already been taken care of. However, the uncertainty corresponds to the model error only to the extent that it differs from the nominal model. The error in a transfer $G_{\boldsymbol{\theta}_{\Delta}}\left(e^{j \omega}\right)$ is equal to

$$
\begin{aligned}
& G_{\theta_{\Delta}}\left(e^{j \omega}\right)-\hat{G}\left(e^{j \omega}\right)-\sum_{i=1}^{n_{s}} \delta_{t r, i} A_{i}\left(e^{j \omega}\right)= \\
& \quad G_{\theta_{\Delta}}\left(e^{j \omega}\right)-\hat{G}^{\perp}\left(e^{j \omega}\right)-\sum_{i=1}^{n_{s}}\left(\delta_{t r, i}+\hat{\delta}_{i}\right) A_{i}\left(e^{j \omega}\right)
\end{aligned}
$$

Assuming that the true system is one of the models in $\mathcal{L}_{\boldsymbol{\theta}}\left(\delta_{t r}\right)$ the absolute value of $\Delta\left(e^{j \omega^{\prime}}\right)$ for some frequency $\omega^{\prime} \in \mathbb{R}$ satisfies

$$
\begin{aligned}
\left|\Delta\left(e^{j \omega^{\prime}}\right)\right| & \leq \max _{\boldsymbol{\theta}_{\Delta} \in \mathcal{C}_{\theta}\left(\delta_{\boldsymbol{t} r}\right)}\left|G_{\theta_{\Delta}}\left(e^{j \omega}\right)-\hat{G}\left(e^{j \omega}\right)-\sum_{i=1}^{n_{s}} \delta_{t r_{i},} A_{i}\left(e^{j \omega}\right)\right| \\
& =\max _{\boldsymbol{\theta}_{\Delta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{\prime}\left(\delta_{\boldsymbol{t} r}\right)}\left|G_{\theta_{\Delta}^{\prime}}\left(e^{j \omega}\right)-\hat{G}^{\perp}\left(e^{j \omega}\right)\right|
\end{aligned}
$$

The maximisation involved here is not a linear programming problem. In the same way as presented in section 4.12 .1 this can be approximated by a series of linear programming problems

$$
\begin{equation*}
\hat{\mu}_{\delta_{t_{r}}, \ell}\left(\omega_{i}\right):=\max _{\theta_{\Delta} \in \mathcal{L}_{\theta}^{\prime}\left(\delta_{t r}\right)} \operatorname{Re}\left(\left(G_{\theta_{\Delta}}\left(e^{j \omega_{i}}\right)-\hat{G}^{\perp}\left(e^{j \omega_{i}}\right)\right) e^{-j \phi_{\ell}}\right) \tag{4.57}
\end{equation*}
$$

In practice one can not assume that the true system is in the model set. The same situation occurred in section 4.12.1. The tail of the basis has an effect that can be bounded by $\bar{\mu}(\omega)$ :

$$
\begin{equation*}
\left|\sum_{k=\bar{k}+1}^{\infty} \theta_{t r, k} \operatorname{Re}\left(B_{k}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)\right| \leq \cdots \leq \bar{\mu}\left(\omega_{i}\right) \tag{4.45}
\end{equation*}
$$

This expression can remain unchanged in the presence of structured error components: the tail of the structured error components is always zero, so that the uncertainty in the tail of the true system is not affected at all by the structured errors.

The only problem that now remains, is that the true values for $\delta_{i}, i \in n_{s}$ are not known. Let $P_{A}^{\perp}$ denote the orthogonal projection operator onto the complement of $\left(\boldsymbol{\theta}_{A, i}\right)_{i=1}^{n_{s}}$. As the $\boldsymbol{\theta}_{A, i}$ are an orthonormal set of vectors, $P_{A}^{\perp}$ satisfies

$$
\begin{equation*}
P_{A}^{\frac{1}{A}}=I-\sum_{i=1}^{n_{3}} \boldsymbol{\theta}_{A, i} \boldsymbol{\theta}_{A, i}^{T} \tag{4.58}
\end{equation*}
$$

The projection operator onto the span of $\boldsymbol{\theta}_{A, i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$ is

$$
\begin{equation*}
P_{A}=\sum_{i=1}^{n_{s}} \boldsymbol{\theta}_{A, i} \boldsymbol{\theta}_{A, i}^{T} \tag{4.59}
\end{equation*}
$$

The projection of $\mathcal{L}_{\boldsymbol{\theta}}$ onto the complement of $\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i=1}^{n_{s}}$ shall be denoted $\mathcal{L}_{\boldsymbol{\theta}}^{\prime}$ :

$$
\begin{equation*}
\mathcal{L}_{\theta}^{\prime}:=\left\{\left.P_{A}^{\frac{1}{A}} \theta \right\rvert\, \theta \in \mathcal{L}_{\theta}\right\} \tag{4.60}
\end{equation*}
$$

For any value of $\delta_{t r}$ it holds

$$
\mathcal{L}_{\theta}^{\prime}\left(\delta_{t r}\right) \subset \mathcal{L}_{\theta}^{\prime}
$$

This implies

$$
\begin{equation*}
\hat{\mu}_{\delta_{t r}, \ell}\left(\omega_{i}\right) \leq \hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right):=\max _{\boldsymbol{\theta} \in \mathcal{L}_{\theta}^{\prime}} \operatorname{Re}\left(\left(G_{\theta}\left(e^{j \omega_{i}}\right)-\hat{G}^{\perp}\left(e^{j \omega_{i}}\right)\right) e^{-j \phi_{\ell}}\right) \tag{4.61}
\end{equation*}
$$

Combining the results obtained so far, it can be concluded that the true value of $\Delta\left(e^{j \omega_{i}}\right)$ must lie in the polytope

$$
\begin{equation*}
\overline{\mathcal{P}}_{m}^{\prime}\left(\omega_{i}\right):=\left\{z \in \mathbb{C} \mid \forall \ell \in \boldsymbol{m} \quad \operatorname{Re}\left(z e^{-j \phi_{\ell}}\right) \leq \hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right)+\tilde{\mu}\left(\omega_{i}\right)\right\} \tag{4.62}
\end{equation*}
$$

This polytope corresponds to $\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$ for the case without structured error components.

Remark 4.13 In practice, translating the set of linear constraints $\mathcal{L}_{\theta}$ to another set of linear constraints $\mathcal{L}_{\theta}^{t}$ is cumbersome. Fortunately, this is not needed. Because (4.61) is a linear programming problem, there exists a vector $f$ of proper dimension, such that

$$
f^{*} \theta \equiv \operatorname{Re}\left(G_{\theta}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)
$$

(Actually the causal relationship is the other way round.) (4.61) can therefore be rewritten as

$$
\hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right)+\operatorname{Re}\left(\hat{G}^{\perp}\left(e^{j \omega_{i}}\right) e^{-j \phi_{\ell}}\right)=\max _{\theta \in \mathcal{L}_{\theta}^{\prime}} f^{*} \theta=\max _{\theta \in \mathcal{L}_{\theta}} f^{*}\left(P_{A}^{\perp} \theta\right)=\max _{\theta \in \mathcal{L}_{\theta}}\left(P_{A}^{\perp} f\right)^{*} \theta
$$

where it was used that $\left(P_{A}^{\perp}\right)^{*}=P_{A}^{\perp}$. The change from $\mathcal{L}_{\theta}$ to $\mathcal{L}_{\theta}^{\prime}$ can apparently be accomplished by changing the object function in a straightforward way.

Remark 4.14 Another issue with respect to the optimisations is the fact that we may actually be dealing with a MIMO process. As the results in this section were derived for a subtransfer of such a MIMO process, the parameter vector $\theta$ used in this section was actually a subvector of the parameter vector $\boldsymbol{\theta}$ of the MIMO model.

For the generalisation to the MIMO case, consider the situation that we want to bound the uncertainty in subtransfer $\left(q^{\prime}, p^{\prime}\right),\left(q^{\prime}, p^{\prime}\right) \in \delta$. The value of $\hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right)$ for example would then be found for this subtransfer as

$$
\hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right):=\max _{\theta \in C_{\theta}^{\prime}} \operatorname{Re}\left(\left[G_{\theta}\left(e^{j \omega_{i}}\right)-\hat{G}^{\perp}\left(e^{j \omega_{i}}\right)\right]_{q^{\prime}, p^{\prime}} e^{-j \phi_{\ell}}\right)
$$

where $\boldsymbol{\theta}$ is the full length MIMO parameter vector. Only the parameter values $\theta_{0}^{q^{\prime}, p^{\prime}}$ to $\theta_{\bar{k}, q^{\prime}, p^{\prime}, p^{\prime}}$ determine the value of the object function in this case. In the notation of the previous remark this means, that all elements of $f$ not associated with one of these values are zero.

This does not imply that for example the constraints

$$
\left|\theta_{t r, k}^{\sigma}\right| \leq \bar{\theta}_{k}^{\sigma} \quad \sigma \neq\left(q^{\prime}, p^{\prime}\right)
$$

may be dropped from the calculation of $\hat{\mu}_{\ell}^{\prime}$, even though these constraints seem to bear no relation with $\theta_{0}^{q^{\prime}, p^{\prime}}, \ldots, \theta_{\overline{q^{\prime}}, p^{p^{\prime}, p^{\prime}}}$. In the case of no structured error components this observation would hold true. In the case of structured error components, this is not true any more. This can easily be seen from the previous remark: the fact that only a subvector of $f$ contains non-zero elements does not imply that only the same subvector of $P_{A}^{\frac{1}{A}} f$ contains non-zero elements. Thus the constraints on the parameters of other sub-transfers are relevant to the optimisations for the current sub-transfer as well!

Example (Continued from page 122): In figure 4.16 the effect of removing the first structured error component from $\overline{\mathcal{P}}_{m}\left(\omega^{\prime}\right)$ is shown. Similarly to figure 4.12 on page 120 , the arrows in this figure point in the directions $e^{j \phi \ell}, \ell \in$ $m$. The length of the arrows is equal to $\hat{\mu}^{\prime}\left(\omega^{\prime}\right)+\bar{\mu}\left(\omega^{\prime}\right)$. The polytope $\overline{\mathcal{P}}_{m}^{\prime}\left(\omega^{\prime}\right)$ is determined from these values. The vertices of this polytope are indicated by o's. The vertex having the largest distance to the origin is marked in addition with a ' + '.


Figure 4.16: Determining $\overline{\mathcal{P}}_{m}^{\prime}\left(\omega^{\prime}\right)$ from $\hat{\mu}^{\prime}\left(\omega^{\prime}\right)+\bar{\mu}\left(\omega^{\prime}\right)$

The polytope $\overline{\mathcal{P}}_{m}\left(\omega^{\prime}\right)$ is drawn in the same picture using dash-dotted lines for comparison. Its vertices are marked with 'x' signs. From this picture can be seen that the polytope $\overline{\mathcal{P}}_{m}^{\prime}\left(\omega^{\prime}\right)$ is shifted with respect to $\overline{\mathcal{P}}_{m}\left(\omega^{\prime}\right)$. Moreover, the shape of $\overline{\mathcal{P}}_{m}^{\prime}\left(\omega^{\prime}\right)$ is somewhat different (smaller) than that of $\overline{\mathcal{P}}_{m}\left(\omega^{\prime}\right)$. Both effects contribute to reducing the largest distance of a vertex to the origin. Apparently the choice of the structured error component was a successful one.

## Weighting, frequency domain interpolation error

The procedure of section 4.12 .1 to interpolate between consecutive $\omega \in \Omega$ is basically followed in the case of structured error components as well. The derivate bound for $W(z) \Delta(z)$ needs to be extended, though. Instead of (4.50), the starting point of the derivation is now

$$
\begin{aligned}
\frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}=\frac{d W\left(e^{j \omega}\right)}{d \omega} & G_{t r}\left(e^{j \omega}\right)+W\left(e^{j \omega}\right) \frac{d G_{t r}\left(e^{j \omega}\right)}{d \omega}- \\
& -\frac{d W\left(e^{j \omega}\right) \dot{G}\left(e^{j \omega}\right)}{d \omega}-\sum_{i=1}^{n_{s}} \delta_{i} \frac{d W\left(e^{j \omega}\right) A_{i}\left(e^{j \omega}\right)}{d \omega}
\end{aligned}
$$

(4.52) gets an extra term $\beta_{3}$ :

$$
\forall \omega \in \mathbb{R}\left|\frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}\right| \leq
$$

$$
\leq\left\|\frac{d W(z) \hat{G}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}+\beta_{3}+\left\|\frac{d W(z)}{d z}\right\|_{\mathcal{H}_{\infty}} \beta_{2}+\|W(z)\|_{\mathcal{H}_{\infty}} \beta_{1}=: \beta
$$

$\beta_{3}$ is a bound for

$$
\sum_{i=1}^{n_{s}} \delta_{i}\left\|\frac{d W(z) A_{i}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}
$$

which is obtained easily:

$$
\sum_{i=1}^{n_{s}} \delta_{i}\left\|\frac{d W(z) A_{i}(z)}{d z}\right\|_{\mathcal{H}_{\infty}} \leq \sum_{i=1}^{n_{s}} \max \left\{\bar{d}_{i},-\underline{d}_{i}\right\}\left\|\frac{d W(z) A_{i}(z)}{d z}\right\|_{\mathcal{H}_{\infty}}=: \beta_{3}
$$

Taking in (4.49) for $v_{k}(\omega)$ now the vertices of the polytope $\overline{\mathcal{P}}_{m}^{\prime}(\omega)$ instead of the vertices of $\overline{\mathcal{P}}_{m}(\omega)$, the rest of the procedure in section 4.12.1 can remain unchanged.

## Influence of bad choice of structured errors

Consider the highly simplified situation of figure 4.17. The left part of this fig-


Figure 4.17: Example with bad structured component
ure represents parameter space. The parameter space has only two dimensions in this example. There is one structured error component, $\boldsymbol{\theta}_{A, 1}$. Consequently, the orthoplement of the span of $\boldsymbol{\theta}_{A, 1}$ is spanned by a single vector, $\boldsymbol{\theta}_{A, 1}^{\perp}$. The right part shows, in a more or less worst case situation, the frequency response of $\boldsymbol{\theta}_{A, 1}$ and $\boldsymbol{\theta}_{\boldsymbol{A}, 1}^{\perp}$. The end points of the two vectors, drawn there, are the points in the complex plane corresponding to $G_{\boldsymbol{\theta}_{A, 1}}\left(\omega_{i}\right)$ and $G_{\boldsymbol{\theta}_{\boldsymbol{A}, 1}}\left(\omega_{i}\right)$. Removing the structured error $\boldsymbol{\theta}_{A, 1}$ from $\mathcal{L}_{\boldsymbol{\theta}}$, resulting in $\mathcal{L}_{\boldsymbol{\theta}}^{\prime}$, will increase the frequency response uncertainty region: the polytope $\mathcal{L}_{\theta}$ is roughly aligned with the line $\theta_{A, 1}=\theta_{\boldsymbol{A}, 1}^{\perp}$. Because $G_{\theta_{A, 1}^{\perp}}\left(\omega_{i}\right) \approx-G_{\theta_{A, 1}}\left(\omega_{i}\right)$ this means, that for a $\boldsymbol{\theta} \in \mathcal{L}_{\boldsymbol{\theta}}$
the contribution of $G_{\theta_{A, 1}}$ is to a large extent compensated by that of $G_{\theta_{A, 1}}$. Separating the structured and unstructured part will fail to further recognise this behaviour. Both $G_{\theta_{A, i}}\left(e^{j \omega_{i}}\right)$ and $G_{\theta_{A, i}}\left(e^{j \omega_{i}}\right)$ will become large and it will be hidden that the true value of their sum is actually rather small.

It should be noted, that the structured error direction $\boldsymbol{\theta}_{A, 1}$ has been chosen intentionally very unfortunately in this example.

This figure demonstrates an element of conservatism that is present in the derivation of the bounds $\underline{d}_{i}$ and $\bar{d}_{i}$ on the one hand and $d_{\Delta}$ on the other. The link that may be present between the values of $\delta_{i}$ and the worst-case value $d_{\Delta}$ is lost.

Remark 4.15 Figure 4.17 can also be interpreted in another way: consider $\boldsymbol{\theta}_{A, 1}$ and $\boldsymbol{\theta}_{A, 1}^{\perp}$ as two structured error components. Here the second structured error component becomes needlessly large, because there is a strong correlation with the first that is not used any more. In this picture this is mainly due to bad alignment of $\boldsymbol{\theta}_{A, 1}$ to the dominating direction in $\mathcal{L}_{\boldsymbol{\theta}}$. This situation will be discussed further in section 5.7.

Obviously, any link that is present between the possible values of $\delta_{i}$ gets lost in the uncertainty description (4.4). As soon as this effect becomes important because of a misalignment of the $\boldsymbol{\theta}_{A, i}$ to $\mathcal{L}_{\boldsymbol{\theta}}$, adding extra structured error components increases conservatism. It is therefore of major importance to choose the structured components properly.

To avoid the inflation of the unstructured error due to the structured error components, we may define the polytope $\overline{\mathcal{P}}_{m}^{\prime \prime}\left(\omega_{i}\right)$ containing $\Delta(\omega)$ as follows:

$$
\overline{\mathcal{P}}_{m}^{\prime \prime}\left(\omega_{i}\right)= \begin{cases}\overline{\mathcal{P}}_{m}\left(\omega_{i}\right) & \text { in case } 1  \tag{4.63}\\ \overline{\mathcal{P}}_{m}^{\prime}\left(\omega_{i}\right) & \text { in case } 2\end{cases}
$$

Either case 1 or case 2 applies, whichever makes $\max _{z \in \mathcal{P}_{m}^{\prime \prime}\left(\omega_{i}\right)}|z|$ smallest.
It is assumed that the decision whether case 1 or case 2 applies is made repeatedly for every $\omega \in n_{\Omega}$. Otherwise one could rather decide whether or not to use structured error components at all: deciding for case 1 for all frequencies and still using structured error components is pointless. Making this decision for every frequency separately may seem harmless at first sight. It does have a subtle but important consequence: the value of $\delta_{i}$ in (4.4) on page 78 has become a function of $\omega$ now! This function assumes one of two values for $\omega \in \Omega$. It is either zero or an unknown but constant value in $\left[{\underset{d}{i}}_{i}, \bar{d}_{i}\right]$.

This is unimportant if $\Delta$ and $\delta_{i}, i \in \boldsymbol{n}_{s}$ are used in $\mu$-design. The situation at hand is depicted in figure 4.18. The structured uncertainty block of $\mu$-design is denoted $D$ to avoid confusion with the unstructured error $\Delta$. The uncertainty block $D$ consists of scalar factors $\delta_{1}, \ldots, \delta_{n_{s}}$ and a block $\Delta^{\prime}, \Delta^{\prime}$ is also a block


Figure 4.18: Uncertainty description in $\mu$-design
of scalars:

$$
\Delta^{\prime}=\left[\begin{array}{ccccc}
\Delta_{11}(\omega) & & & 0 & \\
& \ddots & & & \\
& & \Delta_{1 p}(\omega) & & \\
0 & & & \ddots & \\
& & & & \Delta_{q p}(\omega)
\end{array}\right]
$$

$w$ represents external inputs, $z$ controlled outputs. $C$ is the controller to be designed. $\hat{G}^{\prime}$ is a generalised plant containing the nominal model $\hat{G}$ and weighting filters representing the design requirements. As far as "standard" $\mu$-synthesis is concerned, (Packard and Doyle, 1993) the $\delta_{i}$ are $\mathcal{H}_{\infty}$-norm bounded functions of $\omega$, where they are commonly scaled such that their norm-bound is equal to 1 . The uncertainty description (4.4) is described better by the so-called complexreal $\mu$-synthesis, where it is taken into account that $\delta_{i}$ is a real function of $\omega$. The $\Delta_{i j}$ are complex functions of $\omega$. Complex-real $\mu$ does not use the fact that $\delta_{i}$ is not a function of $\omega$ at all, so if this property is lost, it is no a real loss.

The change of $\delta_{i}$ from real constant to real function is important if the frequency domain procedure is transferred to time domain. It also complicates bounding the interpolation error. It can be accomplished by switching between the "worst case interpolating functions" that are constructed during the bounding of the interpolation error with and without structured errors. This
switching would select the smallest of the two worst case functions. This is not further developed here.

### 4.12.3 Time domain procedure



As with the frequency domain bounds, the procedure for time domain bounds is a procedure for SISO transfers. This means, that every element of $\Delta$ is bounded separately. Hence, every transfer in this section should be considered superscripted by a $\sigma \in \mathcal{S}$. Superscripts $w$ for the current operating point are also omitted.

Signals $\underline{s}(t)$ and $\bar{s}(t)$ will be derived, such that

$$
\forall t \in \mathbb{N} \quad \underline{s}(t) \leq W(\zeta) \Delta(t) \leq \bar{s}(t)
$$

(The convention to denote impulse responses by lower case letters is not followed for the unstructured error $\Delta$ to avoid confusion with the gains $\delta_{i}$ of the structured errors.) $W(\zeta)$ is either a stable weighting filter or $z /(z-1)$. In the latter case, the step response parameters of $\Delta$ are bounded. If $W(\zeta)=1$ the impulse response parameters of $\Delta$ are bounded. The signals $s$ are calculated for every $t$ up to some $t_{\max }$. For $t>t_{\max }$ a constant value is given.

Consider first the situation for $t \leq t_{\max }$. This can be compared to the derivation of frequency response uncertainty regions for $\omega \in \Omega$. The problem that these uncertainty regions have many vertices does not occur in time domain: the uncertainty regions are real intervals. The uncertainty region has again two components, one due to the uncertain model parameters, corresponding to $\hat{\mu}_{k}(\omega)$ and one due to the tail of the basis, similar to $\bar{\mu}(\omega)$. Because the uncertainty region is a real interval we only need to consider the directions $\phi_{1}=0$ and $\phi_{2}=\pi$. This will now be worked out in somewhat more detail.

Let $P_{k}(z)=W(z) B_{k}(z)$ with corresponding impulse response $p_{k}(t)$. By convention, $g_{t r}(t), \hat{g}(t), g_{\theta}(t)$ and $a_{i}(t), i \in n_{s}$ are the impulse responses of the transfer functions $G_{t r}(z), \hat{G}(z), G_{\theta}(z)$ and $A_{i}(z), i \in \boldsymbol{n}_{\boldsymbol{s}}$, respectively. It
holds

$$
W(\zeta) \Delta(t)=W(\zeta)\left(g_{t r}(t)-\hat{g}(t)-\sum_{i=1}^{n_{s}} \delta_{t r, i} a_{i}(t)\right)
$$

Following a line of reasoning completely similar to the one leading to (4.61) in section 4.12.2, from this follows

$$
\begin{equation*}
W(\zeta) \Delta(t) \leq \max _{\theta \in \mathcal{C}_{\theta}^{\prime}} W(\zeta)\left(g_{\theta}(t)-\hat{g}^{\perp}(t)\right)+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right| \tag{4.64}
\end{equation*}
$$

One can see that the maximisation is a linear programming problem by realising that

$$
\max _{\theta \in \mathcal{L}_{\theta}^{\prime}} W(\zeta) g_{\theta}(t)=\max _{\theta \in \mathcal{L}_{\theta}^{\prime}} \sum_{k=0}^{\bar{k}} \theta_{k} p_{k}(t)
$$

In case $W(\zeta)$ is a stable filter, the second term on the right-hand side in (4.64) can be bounded by using the fact that for any stable $G(\zeta)$, it holds

$$
\forall \tau \in \mathbb{N} \quad|g(\tau)| \leq\|G(\zeta)\|_{\mathcal{H}_{\infty}}
$$

which can be verified easily by considering the response to the input $\delta(t-\tau)$ and using that the $\mathcal{H}_{\infty}$-norm is the $\ell_{2}$-induced norm. Choosing a truncation value $\tilde{k} \geq \bar{k}$, we thus have

$$
\begin{aligned}
\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right|= & \sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right| \leq \\
\leq & \sum_{k=\bar{k}+1}^{\dot{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left\|P_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \leq \\
& \leq \sum_{k=\bar{k}+1}^{\bar{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\max _{k>\bar{k}}\left\|P_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}
\end{aligned}
$$

Using the sub-multiplicative property of the $\mathcal{H}_{\infty}$-norm:

$$
\max _{k>n}\left\|P_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \leq\|W(\zeta)\|_{\mathcal{H}_{\infty}} \cdot \max _{k>\tilde{k}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}}
$$

which is computable in finite time because of property 4.1. Combination of results gives

$$
\begin{align*}
\forall t \leq t_{\max } W(\zeta) \Delta(t) & \leq \max _{\boldsymbol{\theta} \in \mathcal{L}_{\boldsymbol{\theta}}^{\prime}} W(\zeta)\left(g_{\theta}(t)-\hat{g}^{\perp}(t)\right)+\sum_{k=\bar{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+ \\
& +\|W(\zeta)\|_{\mathcal{H}_{\infty}}: \max _{k>\bar{k}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}=: \bar{s}(t) \tag{4.65}
\end{align*}
$$

For the case $W(z)=z /(z-1)$ the derivation above can not be used because the $\mathcal{H}_{\infty}$-norm of $W(z)$ does not exist; $W(\zeta) \notin \mathcal{H}_{\infty}$. This does not influence the first term in (4.64), but the second term has to be bounded in a different way. The $p_{k}(t)$ are in this case the step response parameters of $B_{k}(\zeta)$. Two bounds can be formulated for the step response $h(t)$ of a general transfer $G(z)$ :

$$
\begin{align*}
\forall t \in \mathbb{N} \quad|h(t)| \leq(t+1) \cdot \max _{\tau \in \mathbb{N}}|g(\tau)| \leq(t+1) \cdot\|G(\zeta)\|_{\mathcal{H}_{\infty}}  \tag{4.66}\\
|h(t)| \leq\|G(\zeta)\|_{\ell_{1}} \tag{4.67}
\end{align*}
$$

Using (4.66) the second term on the right-hand side of (4.64) is bounded by $\tilde{s}_{1}(t)$ as follows:

$$
\begin{align*}
\forall t \leq t_{\max } & \sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right|=\sum_{k=\bar{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right| \leq \\
& \leq \sum_{k=\bar{k}+1}^{\bar{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+(t+1) \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \leq \\
\leq & \sum_{k=\bar{k}+1}^{\bar{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+(t+1) \max _{k>\bar{k}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \sum_{k=\tilde{\tilde{k}+1}} \bar{\theta}_{k}=: \tilde{s}_{1}(t) \tag{4.68}
\end{align*}
$$

(4.67) leads to this bound:

$$
\begin{array}{r}
\forall t \leq t_{\max } \quad \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right|=\sum_{k=\bar{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left|p_{k}(t)\right| \leq \\
\leq \sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}\left\|B_{k}(\zeta)\right\|_{\ell_{1}} \leq \\
 \tag{4.69}\\
\leq \sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|+\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left(c_{1}+c_{2} \sqrt{k}\right)=\tilde{s}_{2}(t)
\end{array}
$$

$c_{1}$ and $c_{2}$ have been chosen such (conjecture 4.2), that

$$
\begin{equation*}
\forall k \in \mathbb{N} \quad\left\|B_{k}(\zeta)\right\|_{\ell_{1}} \leq c_{1}+c_{2} \sqrt{k} \tag{4.70}
\end{equation*}
$$

Combination of results gives in this case

$$
\begin{aligned}
& \forall t \leq t_{\max } W(\zeta) \Delta(t) \leq \max _{\theta \in L_{\theta}^{\prime}} W(\zeta)\left(g_{\theta}(t)-\hat{g}^{\perp}(t)\right)+ \\
& \quad+\min \left\{\tilde{s}_{1}(t), \tilde{s}_{2}(t)\right\}=: \bar{s}(t)
\end{aligned}
$$

For $t>t_{\text {max }}$ a bound is needed not involving $t$. It holds

$$
\begin{equation*}
W(\zeta) \Delta(t)=\sum_{k=0}^{\infty} \theta_{t r, k} p_{k}(t)-W(\zeta) \hat{g}(t)-\sum_{i=1}^{n_{s}} \delta_{i} W(\zeta) a_{i}(t) \tag{4.71}
\end{equation*}
$$

As

$$
\begin{aligned}
\left|\sum_{k=0}^{\infty} \theta_{t r, k} p_{k}(t)\right| & \leq \sum_{k=0}^{\infty} \theta_{t r, k}\left\|W(\zeta) B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \\
& \leq\|W(\zeta)\|_{\mathcal{H}_{\infty}} \max _{k \in \mathbb{N}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \sum_{k=0}^{\infty} \bar{\theta}_{k} \\
|W(\zeta) \hat{g}(t)| & \leq\|W(\zeta) \hat{G}(\zeta)\|_{\mathcal{H}_{\infty}} \leq\|W(\zeta)\|_{\mathcal{H}_{\infty}}\|\hat{G}(\zeta)\|_{\mathcal{H}_{\infty}} \\
\left|\sum_{i=1}^{n_{s}} \delta_{i} W(\zeta) a_{i}(t)\right| & \leq \sum_{i=1}^{n_{s}} \max \left\{\bar{d}_{i},-\underline{d}_{i}\right\}\left\|W(\zeta) A_{i}(\zeta)\right\|_{\mathcal{H}_{\infty}} \\
& \leq\|W(\zeta)\|_{\mathcal{H}_{\infty}} \sum_{i=1}^{n_{s}} \max \left\{\bar{d}_{i},-\underline{d}_{i}\right\}\left\|A_{i}(\zeta)\right\|_{\mathcal{H}_{\infty}}
\end{aligned}
$$

a conservative bound for (4.71) is

$$
\begin{aligned}
&|W(\zeta) \Delta(t)| \leq\|W(\zeta)\|_{\mathcal{H}_{\infty}}\left(\max _{k \in \mathbb{N}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \sum_{k=0}^{\infty} \bar{\theta}_{k}+\|\hat{G}(\zeta)\|_{\mathcal{H}_{\infty}}+\right. \\
&\left.\sum_{i=1}^{n_{s}} \max \left\{\bar{d}_{i},-\underline{d}_{i}\right\}\left\|A_{i}(\zeta)\right\|_{\mathcal{H}_{\infty}}\right)=: \bar{s}(t), \quad t>t_{\max }
\end{aligned}
$$

This bound does not apply in the case $W(z)=z /(z-1)$ as $\|W(\zeta)\|_{\mathcal{H}_{\infty}}$ is not finite then. Observe that for general $g(t) \in h_{2}, t>t_{\max }$

$$
\begin{equation*}
\left|\sum_{\tau=0}^{t} g(\tau)\right| \leq\left|\sum_{\tau=0}^{t_{\max }} g(\tau)\right|+\sum_{\tau=t_{\max }+1}^{\infty}|g(\tau)| \tag{4.72}
\end{equation*}
$$

Assume for a moment that $\sum_{\tau=0}^{t_{\text {max }}} g(\tau) \geq 0$, so that

$$
\begin{aligned}
\left|\sum_{\tau=0}^{t_{\max }} g(\tau)\right|+\sum_{\tau=t_{\max }+1}^{\infty}|g(\tau)|=\left(\sum_{\tau=0}^{t_{\max }} g(\tau)\right. & -|g(\tau)|)+\sum_{\tau=0}^{\infty}|g(\tau)| \\
& =\left(\sum_{\tau=0}^{t_{\max }} g(\tau)-|g(\tau)|\right)+\|g(t)\|_{\ell_{1}}
\end{aligned}
$$

For $\sum_{\tau=0}^{t_{\max }} g(\tau)<0$ it holds

$$
\begin{aligned}
&\left|\sum_{\tau=0}^{t_{\max }} g(\tau)\right|+\sum_{\tau=t_{\max }+1}^{\infty}|g(\tau)|=\left(\sum_{\tau=0}^{t_{\max }}-g(\tau)-|g(\tau)|\right)+\sum_{\tau=0}^{\infty}|g(\tau)| \\
&=\left(\sum_{\tau=0}^{t_{\max }}-g(\tau)-|g(\tau)|\right)+\|g(t)\|_{\ell_{1}}
\end{aligned}
$$

Now define $\mathcal{B}: h_{2} \times \mathbb{N} \rightarrow \mathbb{R} \cup\{0\}$

$$
\mathcal{B}\left(g(t), t_{\max }\right) \mapsto \begin{cases}\sum_{\tau=0}^{t_{\max }} g(\tau)-|g(\tau)| & \text { if } \sum_{\tau=0}^{t_{\max }} g(\tau) \geq 0 \\ \sum_{\tau=0}^{t_{\max }}-g(\tau)-|g(\tau)| & \text { if } \sum_{\tau=0}^{t_{\max }} g(\tau)<0\end{cases}
$$

Substituting the results above in (4.72) gives

$$
\begin{equation*}
\forall t>t_{\max }\left|\sum_{\tau=0}^{t} g(\tau)\right| \leq \mathcal{B}\left(g(t), t_{\max }\right)+\|g(t)\|_{\ell_{1}} \tag{4.73}
\end{equation*}
$$

We now proceed with

$$
\begin{aligned}
\left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} p_{k}(t)\right| & \leq \sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left(\left\|b_{k}(t)\right\|_{\ell_{1}}+\mathcal{B}\left(b_{k}(t), t_{\max }\right)\right) \\
& \leq \sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k}\left(c_{1}+c_{2} \sqrt{k}\right)+\sum_{k=\tilde{k}+1}^{\tilde{k}} \mathcal{B}\left(b_{k}(t), t_{\max }\right), \quad t>t_{\max }
\end{aligned}
$$

where $\tilde{k} \geq \bar{k}$ is some truncation value and $c_{1}$ and $c_{2}$ are as in (4.70). Further it holds for $t>t_{\text {max }}$

$$
|W(\zeta) \hat{g}(t)| \leq\|\hat{g}(t)\|_{\ell_{1}}+\mathcal{B}\left(\hat{g}(t), t_{\max }\right)
$$

$$
\left|\sum_{i=1}^{n_{s}} \delta_{i} W(\zeta) a_{i}(t)\right| \leq \sum_{i=1}^{n_{s}} \max \left\{\bar{d}_{i}, \underline{d}_{i}\right\}\left(\left\|a_{i}(t)\right\|_{\ell_{1}}+\mathcal{B}\left(a_{i}(t), t_{\max }\right)\right)
$$

Combination of results gives the following bound for (4.71):

$$
\begin{aligned}
& |W(\zeta) \Delta(t)| \leq \sum_{k=0}^{\infty} \bar{\theta}_{k}\left(c_{1}+c_{2} \sqrt{k}\right)+\sum_{k=0}^{\tilde{k}} \mathcal{B}\left(b_{k}(t), t_{\max }\right) \\
& \quad+\mid \hat{G}(\zeta) \|_{\ell_{1}}+\mathcal{B}\left(\hat{g}(t), t_{\max }\right) \\
& \quad+\sum_{i=1}^{n_{s}} \max \left\{\bar{d}_{i},-\underline{d}_{i}\right\}\left(\left\|A_{i}(\zeta)\right\|_{\ell_{1}}+\mathcal{B}\left(a_{i}(t), t_{\max }\right)\right)=: \bar{s}(t), \quad t>t_{\max }
\end{aligned}
$$

with $c_{1}$ and $c_{2}$ as in (4.70) and $\tilde{k} \geq 0$ a truncation value.
The derivation of the lower bound $\underline{s}(t)$ is completely analogous. The results for stable $W(z)$ are given first:

$$
\begin{align*}
\forall t \leq t_{\max } W(\zeta) \Delta(t) & \geq \min _{\theta \in \mathcal{L}_{\theta}^{\prime}} W(\zeta)\left(g_{\theta}(t)-\hat{g}^{\perp}(t)\right)-\sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|p_{k}(t)\right|- \\
& -\|W(\zeta)\|_{\mathcal{H}_{\infty}} \cdot \max _{k>\bar{k}}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}} \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}=: \underline{s}(t) \tag{4.74}
\end{align*}
$$

and

$$
\begin{equation*}
\forall t>t_{\max } \quad \underline{s}(t):=-\bar{s}(t) \tag{4.75}
\end{equation*}
$$

For $W(z)=z /(z-1)$ the results are

$$
\begin{align*}
\forall t \leq t_{\max } W(\zeta) \Delta(t) \geq \min _{\theta \in \mathcal{L}_{\theta}^{\prime}} W(\zeta)\left(g_{\theta}(t)\right. & \left.-\hat{g}^{\perp}(t)\right)- \\
& -\min \left\{\tilde{s}_{1}(t), \tilde{s}_{2}(t)\right\}=: \underline{s}(t) \tag{4.76}
\end{align*}
$$

and, again,

$$
\forall t>t_{\max } \quad \underline{s}(t):=-\bar{s}(t) .
$$

### 4.13 Step 11: combining the local results

```
Agrettmm
```









```
L Combine local results.................................... 140
```



```
    ->>
    T-Take numon of all unstructured bounds
```

Combining the local results from section 4.11 and 4.12 is relatively simple to do. The upper and lower bound on the gains $\delta_{i}$ are

$$
\begin{array}{ll}
\forall i \in n_{s} & \bar{d}_{i}:=\max _{w \in \mathcal{W}} \bar{d}_{i}^{w} \\
& \underline{d}_{i}:=\min _{w \in \mathcal{W}} \underline{d}_{i} \tag{4.78}
\end{array}
$$

If $0 \notin\left[\underline{d}_{i}, \bar{d}_{i}\right]$ for some $i \in \boldsymbol{n}_{\boldsymbol{s}}$, the nominal model is falsified by the data (and prior knowledge): in every operating point a non-zero value for the corresponding $\delta_{i}$ is required to reconcile the model with the data.

A similar situation was discussed in remark 4.10 on page 115. However, remark 4.10 applied locally to an operating point. It is not surprising that the nominal model is not consistent with the data and prior knowledge in an operating point, as it may very well be possible that no linear time-invariant process exists that is consistent with the data and prior knowledge in all operating points. If $0 \notin\left[\underline{d}_{i}, \bar{d}_{i}\right]$ holds globally, where the interval $\left[d_{i}, \bar{d}_{i}\right]$ should contain all values of $\delta_{i}$ that are encountered across all operating points, it means that $\delta_{i}$ has the same sign in all operating points, so that, say $\hat{G}+A_{i} \cdot\left(\underline{d}_{i}+\bar{d}_{i}\right) / 2$ may be a better model for all operating points. This is likely to increase the order of $\hat{G}$, which may be a reason for the user not to use this proposed model. One should however be aware, that $\mu$-synthesis can not use the bounds

$$
\underline{d}_{i} \leq \delta_{i} \leq \bar{d}_{i}
$$

as such; it can only handle

$$
\left|\delta_{i}\right| \leq \max \left\{\bar{d}_{i},-\underline{d}_{i}\right\}
$$

which turns out to be particularly conservative in the case discussed here.
In frequency domain one may take two different approaches for the global bound on $\Delta$. The simplest one is to combine the bounds $d_{\Delta}^{w}$,

$$
\forall w \in \mathcal{W} \quad\left\|W(\zeta) \Delta^{w}(\zeta)\right\|_{\mathcal{H}_{\infty}} \leq d_{\Delta}^{w},
$$

to the global bound

$$
\begin{equation*}
\forall w \in \mathcal{W} \quad\left\|W(\zeta) \Delta^{w}(\zeta)\right\|_{\mathcal{H}_{\infty}} \leq d_{\Delta}:=\max _{w \in \mathcal{W}} d_{\Delta}^{w} \tag{4.79}
\end{equation*}
$$

An alternative would be to derive polytopes that are valid for each operating point and bound the interpolation error similarly to section 4.12.2. This could proceed as follows: Let $\tilde{\mu}_{k}^{w}(\omega)$ be defined as

$$
\forall w \in \mathcal{W}, k \in \boldsymbol{m}, \omega \in \Omega \quad \tilde{\mu}_{k}^{w}(\omega):=\left(\hat{\mu}_{k}^{\prime}\right)^{w}(\omega)+\bar{\mu}_{k}^{w}(\omega)
$$

and the global $\tilde{\mu}_{k}(\omega)$ as

$$
\forall k \in \boldsymbol{m}, \omega \in \Omega \quad \tilde{\mu}_{k}(\omega):=\max _{w \in \mathcal{W}} \tilde{\mu}_{k}^{w}(\omega)
$$

The polytope

$$
\bar{P}_{m}^{g}(\omega):=\left\{z \in \mathbb{C} \mid \operatorname{Re}\left(z e^{-j \phi_{k}}\right) \leq \tilde{\mu}_{k}(\omega)\right\}, \quad \omega \in \Omega
$$

will contain $\Delta\left(e^{j \omega}\right)$ in all operating points. One may proceed to bound the interpolation error between the frequencies in $\Omega$ by using the derivative bound

$$
\forall w \in \mathcal{W}\left|\frac{d W\left(e^{j \omega}\right) \Delta\left(e^{j \omega}\right)}{d \omega}\right| \leq \max _{w \in \mathcal{W}} \beta^{w}=: \beta
$$

(4.53) will then give a global bound on $\|W(\zeta) \Delta(\zeta)\|_{\mathcal{H}_{\infty}}$. However, this will result in a bound not less and possibly larger than the one obtained in (4.79): the "worst case polytopes" over all operating points are interpolated by the "worst case derivatives." It may very well be, that the operating point yielding the worst case polytope is not the one giving the worst case derivative. (The reader going through the calculations involved here will probably recognise these somewhat vague statements.)

Yet it may be useful to calculate the polytopes $\bar{P}_{m}^{g}(\omega)$. Contrary to the $\mathcal{H}_{\infty}$-norm bound, they also contain phase information.

The global time domain bounds are calculated similarly. Recall that

$$
\forall w \in \mathcal{W}, t \in \mathbb{N} \quad \underline{s}^{w}(t) \leq W(\zeta) \Delta^{w}(t) \leq \bar{s}^{w}(t)
$$

The globally bounding signals $\underline{s}(t)$ and $\bar{s}(t)$ are defined as

$$
\begin{align*}
\forall t \in \mathbb{N} & \underline{s}(t):=\min _{w \in \mathcal{W}} s^{w}(t)  \tag{4.80}\\
& \bar{s}(t):=\max _{w \in \mathcal{W}} \bar{s}^{w}(t), \tag{4.81}
\end{align*}
$$

so that

$$
\begin{equation*}
\forall w \in \mathcal{W}, t \in \mathbb{N} \quad \underline{s}(t) \leq W(\zeta) \Delta(t) \leq \bar{s}(t) \tag{4.82}
\end{equation*}
$$

### 4.14 Influence of the choice of the basis

Now that all calculations have been presented, it is a good time to consider the influence of the choice of basis generating system on the results we may expect.

In section 4.7 it was mentioned that the poles of the basis generating system should be taken close to the poles of the linear manifestations of the true system. From appendix B follows that this will lead to small values of $\rho^{\sigma}$ in the bounds on the model parameters

$$
\left|\left(\theta_{t r}\right)_{k}^{\sigma}\right| \leq \bar{\theta}_{k}^{\sigma}:= \begin{cases}\theta_{\max }^{\sigma} & k<\left(k^{*}\right)^{\sigma}  \tag{4.5}\\ M^{\sigma}\left(\rho^{\sigma}\right)^{k} & k \geq\left(k^{*}\right)^{\sigma}\end{cases}
$$

These bounds influence the size of the polytopes $\mathcal{L}_{\theta}^{w}, w \in \mathcal{W}$ in various ways. These polytopes in turn determine the size of the local errors and through these of the global errors.

The most obvious way in which (4.5) influences the size of the polytopes $\mathcal{L}_{\theta}^{w}$ is that the constraints on the model parameters that follow immediately from (4.5) are some of the constraints that constitute this polytope. In a well conditioned situation, these constraints should have a minor influence on the size of the error bounds, compared to the influence of the constraints resulting from data related quantities such as noise bounds. Otherwise the rather suspicious situation would occur, that the experimental data hardly contributes to reduction of the model uncertainty.

Also, in the extension of the noise bound (4.7) to the extended noise bound (4.30) and of the cross-covariance bound (4.9) to the extended cross-covariance bound (4.37) the bounds $\bar{\theta}_{k}^{\sigma}$ are used to bound the effect of the tail of the basis and of unknown initial conditions. As a result of this the final model uncertainty bounds are never determined by the data alone: changing the bounds (4.5) will always change the size of the polytopes $\mathcal{L}_{\theta}^{w}$.

Finally, the bounds on the unstructured error $\Delta(\zeta)$ always consist of a component due to the uncertainty of the model parameters that is not covered by the structured error components and of a component due to the uncertainty in the tail of the basis. The latter uncertainty is also bounded by means of (4.5).

One may not expect to obtain small model error bounds if the basis generating system is badly matched to the true system. The values of $\rho^{\sigma}, \sigma \in S$ will be close to one, which will in turn cause a large influence of the tail of the system on the resulting error bounds. This will be the case for example if the standard basis is used to model a system having a largest time constant that is much larger than the sampling interval. In this case a fast system is used to generate a basis for a slow system. In the reverse situation that a slow system is used to generate a basis for a fast system the same problems will occur. (The interested reader may verify this from theorem B. 2 in appendix B.

### 4.15 Summary

In this chapter a coherent algorithm has been presented to separate the model uncertainty for a process operating in different operating points into a detailed, structured part and a less-detailed, unstructured part. The presentation has been such, that a complete algorithm has been described starting after experiments and pre-processing and ending with the aforementioned separation and bounds on both the structured and unstructured part. Several extensions, sidesteps and discussions that seem interesting have not been discussed to make the main line of steps stand out more clearly. These digressions are discussed in the next chapter.

Stripped from all technicalities, the algorithm can be summarised as follows: for every operating point of a process, identify an auxiliary model. Deduce from these auxiliary models the dominating components in model uncertainty, the structured error components. Bound the uncertainty of these components separately. The unstructured error then consists of two parts: the possible variation in model parameters that is not covered by the structured components and a part due to undermodelling. The structured error is expressed as a real interval in which the gain factor for an explicitly determined transfer function varies. The unstructured error can be expressed as a weighted $\mathcal{H}_{\infty}$-norm bound on every subtransfer of the unstructured error or as real intervals in which the, possibly filtered, impulse response parameters or the step response parameters lie.

### 4.15.1 Requirements on the model parametrisation

Many choices have been made in this algorithm. A choice influencing almost every aspect of the algorithm is the model parametrisation used. A lot of technical problems can be expected if one wants to use a different parametrisation than (4.3) in this algorithm:

- It has been argued at several points that the parametrisation itself should be linear (as opposed to a non-linear parametrisation of linear models.) This is required for the translation of prior knowledge to constraints in parameter space and back to constraints in terms of transfer functions, gains or other control-related entities. It would make sections 4.10, 4.11 and 4.12 considerably more difficult if the parametrisation were not linear. It will most likely result in optimisation problems that are not linear programming problems any more but are even non-convex, non-linear problems.
- Linear parametrisation is also required for a sensible interpretation of the principal components in terms of process uncertainty. Although principal component analysis will give a set of dominating directions for any set
of parameter vectors corresponding to the different auxiliary models, one should ask oneself what information these components contain regarding the process uncertainty in case of a non-linear parametrisation.
- Further it was used several times that different subtransfers are parametrised independently. This was used mainly during the translation of the prior knowledge to a set of linear constraints on the model parameters. It was used to break the computational problems into sub-problems. It also made it possible to split the estimation of a MIMO model into the estimation of several MISO models in section 4.8. It would not be too difficult to change the algorithm such that this feature of the parametrisation is not required any more.
- Finally, it was used that

$$
\begin{aligned}
& \sum_{k=\bar{k}}^{\infty} \bar{\theta}_{k}, \\
& \sum_{k=\bar{k}}^{\infty} \bar{\theta}_{k}\left\|B_{k}(\zeta)\right\|_{\mathcal{H}_{\infty}}, \\
& \sum_{k=\bar{k}}^{\infty} \bar{\theta}_{k}\left\|B_{k}(\zeta)\right\|_{\ell_{1}}
\end{aligned}
$$

and

$$
\sum_{k=\bar{k}}^{\infty} \bar{\theta}_{k} \sum_{\tau=0}^{\infty} \tau\left|b_{k}(\tau)\right|
$$

can be calculated or at least bounded in finite time for the chosen model parametrisation.

Remarkably enough, the fact that system based orthonormal basis functions form an orthonormal set was not used, at least not explicitly. (Section 5.6 reveals by looking more closely at one of the arguments of this chapter that it was used implicitly.)

A lot of the machinery used in this chapter was taken from other authors. Nevertheless the algorithm contains many original aspects. By far the most important aspect is the algorithm itself.

The idea to use different models in different operating points and deduce model uncertainty from those models is used by other authors as well. The fact that these models are analysed to reveal structure in model uncertainty, while still allowing for other sources of model uncertainty, is, to our knowledge, new. In many practical applications different models in different operating points
are used as well. Contrary to the algorithm in this chapter, these different models are either translated to an unstructured error bound or translated to a structured model uncertainty in an unsystematic, ad-hoc way.

The combination of structured and unstructured error components has been used before in for example (Zhou and Kimura, 1994). A new aspect of our algorithm is, that the structured components need not be specified a priori. An attempt is made instead to match these components to the observed process uncertainty.

The application of methods from stochastics/statistics on different model estimates was used in (De Vries, 1994) in a way that vaguely resembles the way in which it is used here. The variations in the model parameters however are treated/interpreted completely differently there. This holds a forteriori for (Goodwin et al., 1992).

The construction of a linear set of constraints on the model parameters based on prior knowledge and experimental data is taken from (Hakvoort, 1994). The bounds on the unstructured error are also based on (Hakvoort, 1994), although the procedure to remove the structured error components is new.

Other original ideas with respect to the algorithm are presented in the next chapter.


- Bound errors locally
- Split parameter uncertainty in structured and unstructured parts
$-<\hat{G}, \hat{\boldsymbol{\theta}}:$ nominal model, vector of its first $\bar{k}$ expansion coefficients in basis $b_{k}$
$\longrightarrow\left(\boldsymbol{\theta}_{A, i} \mid \boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \boldsymbol{\theta}_{A, i}: i$ th structured component for arbitrary $\boldsymbol{\theta}^{\prime}$
$\longrightarrow P$ : projection onto span of $\boldsymbol{\theta}_{A, i}$. p. 128
$\longrightarrow P^{\perp}$ : projection onto orthoplement of span of $\boldsymbol{\theta}_{A, i} \quad$ p. 128
$\longrightarrow P^{\perp}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right)$ : unstructured error component in $\boldsymbol{\theta}^{\prime}$
$\longrightarrow \hat{G}^{\perp}=\hat{G}-G_{P \hat{\theta}}$ : nominal model with structured error directions removed p. 126

Bound structured errors........................................... 113
$\longrightarrow \bar{d}_{i}^{w}:=\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}_{A, i} \mid \boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \boldsymbol{\theta}_{A, i}$ : upper bound on $i$ th structured component p. 114
$\longrightarrow \underline{d}_{i}^{w}:=\min _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}_{A, i} \mid \boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \boldsymbol{\theta}_{A, i}$ :
lower bound on $i$ th structured component p. 115
Bound unstructured error in frequency domain ..... p. 116
$-<e^{j \phi_{\ell}}, \ell=1, \ldots, m$ : set of directions in complex plane p. 117
$-<\Omega=\left\{\omega_{i}\right\}$ : discrete set of frequencies p. 117
$\longrightarrow \hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right)=\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}} \operatorname{Re}\left(\left(G_{P^{\perp} \boldsymbol{\theta}^{\prime}}\left(e^{j \omega_{i}}\right)-\hat{G}^{\perp}\left(e^{j \omega_{i}}\right)\right) e^{-j \phi_{\ell}}\right)$
uncertainty in direction $e^{j \phi_{\ell}}$ for frequency $\omega_{i}$ due to param. uncertainty not covered by struct. error
p. 128
$\longrightarrow \bar{\mu}\left(\omega_{i}\right):$ error due to tail of basis
p. 118
$\longrightarrow \overline{\mathcal{P}}_{m}^{\prime}\left(\omega_{i}\right)$ : uncertainty region in complex plane determined by $\hat{\mu}_{\ell}\left(\omega_{i}\right)$ and $\bar{\mu}\left(\omega_{i}\right)$
p. 128

Bound the interpolation error for $\omega \notin \Omega$ p. 130
Bound unstructured error in time domain ............ p. 134
$-<W(\zeta)$ : stable weighting filter or $z /(z-1) \quad$ p. 134
$-<t_{\max }$ : time after which constant bounds are used p. 134
$\longrightarrow \bar{s}(t)$ : upper bound for $W(\zeta) \Delta(t)$ p. 137
$\longrightarrow \underline{s}(t)$ : lower bound for $W(\zeta) \Delta(t) \quad$ p. 139

- Combine local results............................................................. 140
$\longrightarrow \bar{d}_{i}:=\max _{w \in \mathcal{W}} \bar{d}_{i}:$ upper bound for $i$ th structured component
$\longrightarrow \underline{d}_{i}:=\min _{w \in \mathcal{W}} \underline{d}_{i}:$ lower bound for $i$ th structured component
- Take union of all unstructured bounds


## 5

## Extensions to the basic algorithm

5.1 Introduction
5.2 Estimating initial conditions
5.3 Regularisation of Least Squares estimates
5.4 Prior knowledge of static gains
5.5 Prior knowledge of complex gains for arbitrary frequencies

### 5.6 Weighting of parameters

5.7 Other ways to determine the error structure
5.8 Unstable systems and closed loop experiments

### 5.1 Introduction

The algorithm of chapter 4 forms a more or less complete recipe for the identification of structured and unstructured error components. Some of the steps in this framework deserve reconsideration. This will lead to some discussions that are useful in their own right, others are only valuable within the context of the algorithm.

For proper appreciation of the points made in this chapter, a firm understanding of the algorithm of chapter 4 is required. As in chapter 4 , a summary of the algorithm is given at appropriate points in this chapter to indicate with what part of the algorithm the following remarks will be concerned.

### 5.2 Estimating initial conditions

| Bromidam |  |
| :---: | :---: |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| - Construct set of linear constraints ................................ 100 |  |
|  |  |
| - Extend noise bound . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . p. 100 |  |
| - ${ }^{\text {e }}$ (t) ( (unextended) noise bound | p. 82 |
| $\rightarrow \hat{a}(t)$ infuence of basis $\hat{k}+1$ to $k$, no trausient | p. 107 |
| $\rightarrow$ b(t) transient of basis 0 to $k$ | p. 107 |
| $\rightarrow \delta$ infuence of bavis $k+1$ aud higher, incl. trausient | p. 108 |
| $\longrightarrow\left\|v(t)-G_{\theta}(C) u(t)\right\| \leq e_{e}(t):=\bar{\varepsilon}(t)+\hat{a}(t)+\hat{b}(t)+\delta$ <br> Extend cross-covariance bound | p. 108 |
| < $\mathrm{c}_{4} \mathrm{r}_{( }(t)$ : (mextended) cross-covariasice bound | p. 83 |
| $\rightarrow d(l):$ basis function $\bar{k}+1$ to $\hat{L}$, known mputs | p. 111 |
| $\rightarrow f(\ell)$ : basis function $\bar{k}+1$ to $\hat{k}$, irputs u(-i) to $u(-1)$ | p. 111 |
| $\rightarrow \delta_{1}(\theta)$ : basis function $\bar{k}+1$ to $k$, iuputs $n(-1-1)$ to $u(-\infty)$ | p. 111 |
| $\rightarrow \delta_{2}(f):$ basis function $\hat{k}+1$ and bigher, all inputs $\rightarrow \mid \sum_{t} r_{k}(t)\left(\nu(t)-G_{s}(C) u(t)\right]$ | p. 111 |
| $\leq d(f)+f(\ell)+\delta_{1}(\theta)+\delta_{2}(\theta)+\bar{\epsilon}_{t} \sqrt{T^{\prime}}$ $\mathcal{C}_{\theta}^{\prime \prime}:$ set of hinear constraints on $\theta$ for every operating point | p. 112 |
| Bbint mamm butaly | ! ! |
|  | \% l |

### 5.2.1 Motivation

The extended noise bound (4.30) and the extended cross-covariance bound (4.37) are in general quite conservative with respect to the influence of initial conditions. In order to demonstrate this, data was generated by filtering a normally distributed, zero mean white-noise sequence with standard deviation 0.95 by the system

$$
G_{t r}(z)=\frac{0.9}{z-0.9}
$$

No noise was added to the output and a basis was generated using the true system. Theoretically this means that in this case only $\theta_{t r, 0}$ and $\theta_{t r, 1}$ can be non-zero in the expansion

$$
G_{t r}(\zeta)=\sum_{k=0}^{\infty} \theta_{t r_{2} k} B_{k}(\zeta)
$$

This information could have been given to the algorithm through prior values of $\bar{\theta}, M$ and $\rho$, but this was not done; the algorithm estimated these quantities based on the estimated auxiliary model as suggested in section 4.8.3. The input was used as instrumental variable.

The following bounds on the model parameters were found:

$$
\begin{aligned}
& \bar{\theta}_{0}=20.65 \\
& \bar{\theta}_{k}=5.49 \cdot 10^{6} \cdot\left(5.65 \cdot 10^{-6}\right)^{k}, \quad k \geq 1
\end{aligned}
$$

The bound on inputs prior to $t=0$ was set to 2.48 , the maximal absolute value of the input sequence.

Five basis functions were used in the model set. The truncation value $\bar{k}$ in (4.25) was set to ten. In figure 5.1 the output signal is plotted, together with the signal $\hat{a}(t)$, bounding the effect of the known inputs through the basis functions $B_{6}$ to $B_{10}$, and $\hat{b}(t)$, bounding the effect of unknown initial conditions of basis functions $B_{0}$ to $B_{10}$.


Figure 5.1: Output signal and extensions to the noise bound
Due to the very small values of the bounds on the model parameters $\bar{\theta}_{6}$ to $\bar{\theta}_{10}$, the signal $\hat{a}(t)$ is very small. Its standard deviation is of the order $10^{-21}$. The signal $\hat{b}(t)$ is initially very large. Compared to the standard deviation of 1.3 of the output signal, its initial value of about 85 is excessive.

Notwithstanding the large value of $\hat{b}(t)$, the situation is in one respect optimised towards reducing the bound on the initial conditions: due to the exact match of basis generating system and true system, the influence of all but the first two basis functions is negligible in $\hat{b}(t)$. In practice, more basis functions will have a significant contribution to $\hat{b}(t)$. The more basis functions become relevant in $\hat{b}(t)$, the larger it will become and, more importantly, the longer it will take to die out.

This effect does not only occur in the extension of the noise bound. A similar effect occurs for the extended cross-covariance bound. A good relative comparison is in this particular case not possible. The output being noise-free and the system being in the model set, the residuals found by the estimation
routine are very small. The standard deviation of the signal of residuals is about $10^{-15}$. This implies that the non-extended value for the cross-covariance bound $\bar{c}_{\ell}$ in (4.9) will be very small, making comparison to the other contributions to the extended cross-covariance bound meaningless.

The unrealistic inflation of the extended noise bound and the extended cross-covariance bound is caused by the fact that a worst case bound on the transient due to the inputs before $t=0$ is used. The bound is based on $\ell_{1}$ norms, which means that the input is assumed to be against its bound for every $t<0$ with the sign determined such, that the output magnitude for $t=0$ is as large as possible. Based on the prior knowledge, this input is possible before $t=0$, but it is highly unlikely that it occurs. Apparently, the output reaches its $\ell_{1}$-norm based upper bound only for a very specific and unlikely input sequence for $t<0$. For different transfers, this specific input sequence is likely to be different. Some conservatism is therefore introduced by summing $\ell_{1}$ norms in (4.27), (4.28), (4.34) and (4.35).

An alternative to this approach is to use an estimate of the transient and adapt the algorithm to use it properly.

### 5.2.2 Estimate of transient

Estimating the effect of non-zero initial conditions can be incorporated fairly simply into (4.12) in case of a least squares estimation if attention is restricted to the transient of the model. The difference between the transient of the model and the transient of the process will be discussed in remark 5.3 on page 156 . Instead of writing the model output as a weighted sum of inputs filtered by basis functions, it should now be expressed as a weighted sum of filtered inputs and free-run responses to non-zero initial conditions of these filters.

In the case of system based orthonormal basis functions the first basis function - $B_{0}(\zeta)$ - is equal to a constant gain. It has no transient behaviour and need not be considered as far as initial conditions are concerned. Assume the basis generating system has McMillan degree $n_{b}$. In appendix $\mathrm{B}, B_{k}(\zeta)$ will be constructed as one of the sub-transfers of an $n_{b}$-output, 1 -input system of McMillan degree $\lceil k\rceil_{n_{b}}$. (Recall that the notation $\lceil a\rceil_{b}$ was introduced for " $a$ rounded up to the nearest multiple of $b$ " as a generalisation of $\lceil a\rceil$ for " $a$ rounded up to the nearest integer.") This means that the space of signals that can be observed at the output of a filter $B_{k}(\zeta)$ due to non-zero initial conditions is at most $[k]_{n_{b}}$-dimensional. It can be less than this value if not all states are fully observable from this single output.

For a model being a linear combination of $n$ basis functions, the space of all output signals generated by initial conditions can obviously be spanned by $\sum_{k=1}^{n-1}\lceil k\rceil_{n_{b}}$ different output signals. This number is of the order $n^{2}$. Fortunately, the actual dimension of the aforementioned space is of order $n$, see appendix B. The problem of obtaining a (nearly) minimal number of output
signals spanning the space of output signals that can observed as a result of non-zero initial conditions will not be discussed here. In appendix $B$ a procedure is derived by which such a set of signals can be constructed, without generating the full set of order $n^{2}$ signals.

Let $n_{x}^{j i}$ be the number of signals that are produced by the aforementioned procedure to span the space of possible transients in the subtransfer from the $i$ th input to the $j$ th output, $i \in p, j \in \boldsymbol{q}$. It is assumed that the result of the procedure is available in the form of matrices $X_{j i}^{d} \in \mathbb{R}^{T^{d} \times n_{x}^{j i}}$ such that for each data set $d \in \mathcal{D}$, the columns of $X_{j i}^{d}$ span the space of all transients that can be exhibited by the model for the subtransfer.at hand on the time interval $\mathbb{T}^{d}$. In other words, let $x_{j i}^{d}(t): \mathbb{N} \rightarrow \mathbb{R}$ be the transient in data set $d$ due to the initial conditions of the model for subtransfer $(j, i)$, then

$$
\left[\begin{array}{c}
x_{j i}^{d}(0) \\
\vdots \\
x_{j i}^{d}\left(T^{d}-1\right)
\end{array}\right] \in \operatorname{img} X_{j i}^{d}
$$

$X_{j i}^{d}$ depends only on the data set through the value of $T^{d}$ : the data set determines only the number of rows in $X_{j i}^{d}$. If all data sets are the same length, the value of $X_{j i}^{d}$ will not depend on $d$. Further, if the same basis is used for all sub-transfers sharing an output and the model order is also chosen the same, then the value of $X_{j i}^{d}$ will not depend on the input index $i$. This has a straightforward analog in cases where the same basis and model order are used for all sub-transfers sharing an input.

Analogously to section 4.8.1, first the situation with one output will be considered. The subscript $j$ will therefore be dropped. $X^{d}$ is built from the $X_{i}^{d}$ similarly to $U^{d}$ in section 4.8.1:

$$
\forall d \in \mathfrak{D}(w) \quad X^{d}:=\left[\begin{array}{lll}
X_{1}^{d} & \ldots & X_{p}^{d}
\end{array}\right]
$$

$X$ is finally obtained as

$$
X:=\operatorname{diag}_{d \in \mathcal{D}(w)} f^{d} X^{d}
$$

The weighting by $f^{d}$ is applied here for consistency with $U$ and $Y$, it is not strictly necessary. The values of $U$ and $Y$ are as in (4.11).

Let $\phi_{x}$ and $\hat{\phi}_{x}$ be column vectors having as many elements as $X$ has columns. The combined estimation of both the model parameters and the transients due to initial conditions is now performed by solving

$$
\left\|Y-[U X] \cdot\left[\begin{array}{c}
\hat{\theta}  \tag{5.1}\\
\hat{\phi}_{x}
\end{array}\right]\right\|_{2}=\min _{\theta, \phi_{\boldsymbol{x}}}\left\|Y-[U X] \cdot\left[\begin{array}{c}
\theta \\
\phi_{x}
\end{array}\right]\right\|_{2}
$$

for $\hat{\boldsymbol{\theta}}$ and $\hat{\phi}_{\boldsymbol{x}} . \hat{\boldsymbol{\theta}}$ is the estimate of the model parameters similar to (4.12). The vector $X \hat{\phi}_{\boldsymbol{x}}$ contains estimates of the transients for each data set $d \in \mathcal{D}(w)$, stacked on top of each other, weighted by the weighting factors $f^{d}$. For future reference, we introduce $\hat{y}_{x}^{d}, d \in \mathcal{D}(w)$ implicitly through the equation

$$
\operatorname{stack}_{d \in \mathcal{D}(w)} f^{d} \hat{y}_{x}^{d}:=X \tilde{\phi}_{x}
$$

where $\hat{y}_{x}^{d}$ has as many rows as $Y^{d}$ has, so that the partitioning of the matrix on the left hand side is compatible with the partitioning of $X$ into its constituents, $X^{d}, d \in \mathcal{D}(w)$.

Remark 5.1 $X$ is a block diagonal matrix containing the weighted $X^{d}$, whereas $U$ is built from the weighted $U^{d}$ by stacking them on top of each other. This difference is due to the fact, that for every data set of an operating point, the model parameters are taken the same, whereas the transient can be different for every data set.

If the matrix $[U X]$ has full column rank, $\hat{\boldsymbol{\theta}}$ and $\hat{\phi}_{x}$ are unique. Ordinary Least Squares algorithms can be used to solve (5.1). The remainder of this section will be concerned with the situation that $[U X]$ does not have full column rank.

It is assumed that $[U X]$ is a tall matrix. For rank deficient $[U X]$, three different kinds of "deficiencies" can then be distinguished. These may occur simultaneously. Because it was assumed that $[U X]$ does not have full column rank, at least one occurs:

1. $X$ does not have full column rank, i.e. a column of $X$ can be written as a linear combination of other columns of $X$.
2. $U$ does not have full column rank, i.e. a column of $U$ can be written as a linear combination of other columns of $U$.
3. The intersection of the image of $U$ and the image of $X$ has rank one or more, i.e. a column of $U$ can be written as a linear combination of other columns of $U$ and at least one column of $X$.

Situation 1 is merely inconvenient: the actual values of $\hat{\phi}_{x}$ are not required, only the values of $\hat{y}_{x}^{d}(t)$. It is easily verified through a projection argument, that for all $\hat{\phi}_{x}$ satisfying (5.1), X $\hat{\phi}_{x}$ is unique provided situation 3 does not occur. For numerical reasons, an unambiguous value of $\hat{\phi}_{x}$ is obtained by picking the value among all $\hat{\phi}_{x}$ satisfying (5.1) that has minimum norm. Any other selection could have been made instead.

Situation 1 is likely to occur: for a multiple input system with the same basis used for all subtransfers, the space of possible transients of input $i$ is contained by that of input $j$ if the model order for input $i$ is less than or equal to that of
input $j$. (For multiple outputs this argument applies on a per-output basis.) This is related to the non-minimality discussed in remark 4.1 on page 78.

Situation 2 corresponds to a fundamental problem. Apparently the input signal was not rich enough to uniquely identify a model for the experimental data. There is simply insufficient information in the data record. For a multiple input system it might be the case, that the signals applied to each input are sufficiently rich separately, but that there is a relationship between the various input signals making it impossible to attribute the outputs to an unambiguous combination of inputs. Whether or not this applies, additional experiments should be carried out with a better input design.

Situation 3 is strictly speaking as much of a fundamental problem as situation 2 is. The best thing to do is, again, performing additional experiments. As the transients are only relevant in the first part of an experiment of sufficient length, occurrence of this situation means either that the input was only exciting in the very beginning of the experiment, the experiment was too short or that too many basis functions are being used in the model set. If it is not possible to do further experiments, the "explanation" of the measured output has to be distributed over the responses to the inputs on the one hand and the "responses" to the initial conditions on the other hand. Any choice here is arbitrary and it is trivial to construct for any choice an example in which that choice works out particularly badly. In the following, as much of the output signal will be attributed to the inputs as possible. This is inspired by the fact that practical identification in which no initial conditions are estimated implicitly does the same. This is admittedly not a very strong argument, the more so because this situation will not occur in practice for proper experimental data and model orders.

The actual estimation now proceeds as follows. The matrix $[U X]$ is decomposed into an orthogonal matrix $Q$, a (singular) upper-triangular matrix $R$ and a permutation matrix $P$ :

$$
[U X]=: Q R P^{-1}=\left[\begin{array}{ll}
Q_{1} & Q_{2}
\end{array}\right]\left[\begin{array}{cc}
R_{11} & R_{12}  \tag{5.2}\\
0 & 0
\end{array}\right] P^{-1}
$$

where $Q$ and $R_{11}$ are square matrices. By an orthogonal transformation $Z$ this can be transformed into

$$
[U X] P=\left[\begin{array}{ll}
Q_{1} & Q_{2}
\end{array}\right]\left[\begin{array}{cc}
T_{11} & 0  \tag{5.3}\\
0 & 0
\end{array}\right] Z .
$$

The factorisation (5.3) is called a complete orthogonal factorisation of [ $U X$ ], see (Anderson et al., 1995). The solution $\left[\hat{\boldsymbol{\theta}}^{T} \hat{\boldsymbol{\phi}}_{\boldsymbol{x}}{ }^{T}\right]^{T}$ is finally obtained as

$$
\left[\begin{array}{c}
\hat{\boldsymbol{\theta}}  \tag{5.4}\\
\hat{\boldsymbol{\phi}}_{x}
\end{array}\right]=P Z^{T}\left[\begin{array}{c}
T_{11}^{-1} Q_{11}^{T} Y \\
0
\end{array}\right]
$$

This is the minimum norm solution to (5.1), i.e. of all vectors $\left[\hat{\boldsymbol{\theta}}^{T} \hat{\boldsymbol{\phi}}_{\boldsymbol{x}}{ }^{T}\right]^{T}$ satisfying (5.1), the one obtained by (5.4) is the one having the smallest norm.

Situation 2 should not occur, as argued above, so that $U$ is of full column rank. Situation. 1 is properly handled by (5.4). Situation 3 can be handled by requiring that the first $n_{U}$ columns of $P$ are the first $n_{U}$ unit vectors of appropriate length, in any order, where $n_{U}$ is the number of columns of $U$. Because $U$ is of full rank, this still implies $R_{11}$ is of full rank.

The functionality outlined above is present in the LAPACK routine DGELSX.

Remark 5.2 For numerical reasons, the decomposition (5.2) is in general not possible. It is very unlikely that $R_{22}$ is exactly zero in practical cases. This is handled by DGELSx by defining both the permutation $P$ and the partitioning of $R$ such, that $R_{11}$ is the largest leading sub-matrix whose estimated condition number does not exceed a certain threshold value, related to numerical precision. The remaining value of $R_{22}$ is then neglected and the procedure continues as explained above.

A positive side-effect of this is, that " $a$ is a linear combination of $b$ " in the descriptions of the three situations in which $[U X]$ is rank deficient can be reinterpreted as " $a$ is a linear combination of $b$ up to numerical accuracy."

For multiple output systems, the procedure should simply be repeated for each output. In summary, for each operating point we have obtained an estimate $\hat{\boldsymbol{\theta}}$ for the model parameters. Also, for each data set and for each output indexed by $j \in q$, an estimate for the transient in that output is obtained in the form of $\hat{y}_{x, j}^{d}(t)$.

Remark 5.3 Note that the estimate of the transient obtained in this way is always in the space of transients that can be exhibited by the model. The transient of the tail of the basis is not estimated. The transient of the process for a data set $d \in \mathcal{D}$ can therefore be outside the image space of $X^{d}$. The transient of the model is in fact fitted to the experimental data of the process. This "undermodelling" of the transient will result in a bias on the estimated parameters and/or an increase in the residuals. The former effect becomes smaller if the length of the data set increases with respect to the length of the transient.

The undermodelling of the transient can obviously be reduced by increasing the model order. The order that is used to estimate the transient can be chosen independently of the actual model order. The matrices $X^{d}$ can without any problem be chosen such that they span the space of possible transients of a different set of basis functions than the set that is incorporated in the model. In this case the term "transient of the model" is actually not completely correct. For brevity we will nevertheless use this phrase to refer to the quantity that is estimated by $\hat{y}_{x}^{d}(t)$.

### 5.2.3 Adaptation of the transient bounds

In the extended noise bound

$$
\bar{e}_{e}(t)=\bar{e}(t)+\hat{a}(t)+\hat{b}(t)+\delta
$$

defined in (4.30) on page 108, the transient of the model is bounded by the term $\bar{b}(t)$, as can be seen from

$$
\begin{equation*}
\left|\sum_{k=0}^{\tilde{k}} \theta_{t r, k} \sum_{\tau=t^{\prime}+1}^{\infty} b_{k}(\tau) u\left(t^{\prime}-\tau\right)\right| \leq \cdots \leq \hat{b}\left(t^{\prime}\right) \tag{4.27}
\end{equation*}
$$

It is assumed that the transient of the first $\tilde{k}+1$ basis functions is estimated in the estimates $\hat{y}_{x}^{d}(t), d \in \mathcal{D}$ from the previous section. This makes $\hat{y}_{x}^{d}(t)$ an estimate for the quantity that is overbounded by $\bar{b}(t)$. In principle $s \cdot\left|\hat{y}_{x}^{d}(t)\right|$ could be used as an alternative for $\bar{b}(t)$, where $s \geq 1$ is a safety margin. Given the worst case nature of $\vec{b}(t)$, this alternative is likely to be a smaller bound than $\bar{b}(t)$. However, this would even for $s=1$ include the transient $-\hat{y}_{x}^{d}(t)$, which corresponds to an error in the estimate of $200 \%$.

A more realistic approach is to subtract the estimated transient from the output and use $s^{\prime} \cdot\left|\hat{y}_{x}^{d}(t)\right|$ as a bound on the remaining transient of the model. $s^{t} \geq 0$ is another safety margin. This amounts to

$$
\begin{equation*}
\left|y(t)-\hat{y}_{x}(t)-\sum_{k=0}^{\bar{k}} \theta_{t r} \sum_{\tau=0}^{t} b_{k}(\tau) u(t-\tau)\right| \leq \bar{e}(t)+\hat{a}(t)+s^{\prime} \cdot\left|\hat{y}_{x}(t)\right|+\delta \tag{5.5}
\end{equation*}
$$

(5.5) corresponds to the original extended noise bound (4.30) with the term $\vec{b}(t)$ replaced by $s^{t} \cdot\left|\hat{y}_{x}(t)\right|$.

In the extension of the cross-covariance bound the terms $f(\ell)$ and $\delta_{1}(\ell)$ account for the influence of the transient of the model:

$$
\begin{equation*}
\left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{\tau=t+1}^{t+\bar{t}} b_{k}(\tau) u(t-\tau)\right| \leq \cdots \leq f(\ell) \tag{4.33}
\end{equation*}
$$

and

$$
\begin{equation*}
\left|\sum_{k=0}^{\tilde{k}} \theta_{t r, k} \sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{T=t+\bar{t}+1}^{\infty} b_{k}(\tau) u(t-\tau)\right| \leq \cdots \leq \delta_{1}(\ell) \tag{4.34}
\end{equation*}
$$

The true transient of the model, i.e. of the first $\tilde{k}+1$ basis functions in the expansion of the linear manifestation of the true process in an operating point $d \in \mathcal{D}$, will be denoted $y_{x}^{d}(t)$. The original bound was

$$
\left|\sum_{t=0}^{T^{d}-1} r_{\ell}(t) y_{x}^{d}(t)\right| \leq f(\ell)+\delta_{1}(\ell)
$$

Using the estimate $\hat{y}_{x}^{d}(t)$ for $y_{x}^{d}(t)$ and a safety margin $s^{\prime} \geq 0$ this bound can be replaced by

$$
\begin{equation*}
\left|\sum_{t=0}^{T^{d}-1} r_{\ell}(t) y_{x}^{d}(t)-\sum_{t=0}^{T^{d}-1} r_{\ell}(t) \hat{y}_{x}^{d}(t)\right| \leq s^{\prime} \sum_{t=0}^{T^{d}-1}\left|r_{\ell}(t) \hat{y}_{x}^{d}(t)\right|=: \bar{c}_{x, \ell} \tag{5.6}
\end{equation*}
$$

Instead of the extended cross-covariance bound

$$
\begin{align*}
&\left|\sum_{t=0}^{T^{d}-1} y(t) r_{\ell}(t)-\sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{\tau=0}^{t} b_{k}(\tau) u(t-\tau)\right| \leq \\
& \leq d(\ell)+f(\ell)+\delta_{1}(\ell)+\delta_{2}(\ell)+\bar{c}_{\ell} \sqrt{T^{d}} \tag{4.37}
\end{align*}
$$

the bound

$$
\begin{array}{r}
\left|\sum_{t=0}^{T^{d}-1} y(t) r_{\ell}(t)-\sum_{t=0}^{T^{d}-1} \hat{y}_{x}^{d}(t) r_{\ell}(t)-\sum_{t=0}^{T^{d}-1} r_{\ell}(t) \sum_{k=0}^{\bar{k}} \theta_{t r, k} \sum_{r=0}^{t} b_{k}(\tau) u(t-\tau)\right| \leq \\
\leq d(\ell)+\bar{c}_{x, \ell}+\delta_{2}(\ell)+\bar{c}_{\ell} \sqrt{T^{d}} \tag{5.7}
\end{array}
$$

is used.
The linear constraints based on (4.30) and (4.37) are finally removed from the set of linear constraints $\mathcal{L}_{\theta}$ and those following from (5.5) and (5.7) are added. The algorithm of chapter 4 can then proceed as before.

For an example of the potential value of this approach, see section 6.2.

### 5.3 Regularisation of Least Squares estimates

```
ABpmban
-On wapminmax
```




```
- Estimate auxiliary models ................................. }0
    -u(t),y(t),v(t), model parametrisation
    \longrightarrow\mp@subsup{0}{}{w}: parameters of auxiliary model for openatiag point w p. }2
```




```
-Bomu pmar focaly
```



In chapter 4 least squares/instrumental variable estimation was used as an example in all steps where identification of a model by the algorithm was required. In section 4.8 .2 it was shown, that this may lead to numerically ill-conditioned problems and unrealistic estimated models if the length of the experimental data sets is much smaller than the impulse responses of the high order basis functions that are incorporated in the model.

Consider the situation of a SISO system for which one data set is available, as was done in section 4.8.2. The parameter estimate is in this case

$$
\hat{\boldsymbol{\theta}}=\left[\frac{1}{T} U^{T} U\right]^{-1} \frac{1}{T} U^{T} Y
$$

where

$$
\begin{aligned}
& U=\left[\begin{array}{ccc}
{\left[B_{0}(\zeta) u\right](0)} & \ldots & {\left[B_{\bar{k}}(\zeta) u\right](0)} \\
\vdots & & \vdots \\
{\left[B_{0}(\zeta) u\right](T-1)} & \ldots & {\left[B_{\bar{k}}(\zeta) u\right](T-1)}
\end{array}\right] \\
& Y=\left[\begin{array}{c}
y(0) \\
\vdots \\
y(T-1)
\end{array}\right]
\end{aligned}
$$

$T$ is the length of the data set and $\ddot{k}+1$ is the number of basis functions in the model. The $i$ th column of $U$ contains the signal $u(t)$ filtered by $B_{i-1}(\zeta)$. In section 4.8 .2 the matrix

$$
Q=\frac{1}{T} U^{T} U
$$

was introduced and it was shown that for a white input sequence $u Q \rightarrow \sigma_{u}^{2} I$ if $T \rightarrow \infty$, with $\sigma_{u}$ the standard deviation of $u$. It was also demonstrated that for finite $T$ the value of $Q$ can be near singular.

If this effect occurs, it can be reduced by extending the length of the data set. This is likely to be undesirable in a practical situation: it is time consuming, it may be expensive, it will increase the computational load of the identification, etc. However, one can extend the input sequence with zeroes. Let $T$ be the length of the original data set and $T^{\prime}$ the length of the extended data set. Moreover, let $\sigma_{u}^{\prime}$ be the standard deviation of $u$ on the time interval $t=0, \ldots, T-1$. It is obtained

$$
\begin{align*}
T^{\prime}[Q]_{j i} & =\lim _{T^{\prime} \rightarrow \infty} \sum_{t=0}^{T^{\prime}}\left[B_{j}(\zeta) u\right](t)\left[B_{i}(\zeta) u\right](t) \\
& =\lim _{T^{\prime} \rightarrow \infty} \sum_{t=0}^{T^{\prime}}\left(\sum_{\tau_{1}=0}^{t} b_{j}\left(\tau_{1}\right) u\left(t-\tau_{1}\right)\right) \cdot\left(\sum_{\tau_{2}=0}^{t} b_{i}\left(\tau_{2}\right) u\left(t-\tau_{2}\right)\right) \\
& =\lim _{T^{\prime} \rightarrow \infty} \sum_{\tau_{1}=0}^{\infty} \sum_{\tau_{2}=0}^{\infty} b_{j}\left(\tau_{1}\right) b_{i}\left(\tau_{2}\right) \cdot \sum_{t=\max \left\{\tau_{1}, \tau_{2}\right\}} u\left(t-\tau_{1}\right) u\left(t-\tau_{2}\right) \\
& \approx T \sigma_{u}^{\prime} \sum_{\tau=0}^{\infty} b_{j}(\tau) b_{i}(\tau) \tag{5.8}
\end{align*}
$$

The fourth (approximate) equality deserves some further explanation. Due to the assumed whiteness of the noise it holds

$$
\lim _{T \rightarrow \infty} \frac{1}{T} \sum_{t=\max \left\{\tau_{1}, \tau_{2}\right\}}^{\min \left\{\tau_{1}, \tau_{2}\right\}+T-1} u\left(t-\tau_{1}\right) u\left(t-\tau_{2}\right)= \begin{cases}\sigma_{u}^{\prime} & \text { if } \tau_{1}=\tau_{2} \\ 0 & \text { if } \tau_{1} \neq \tau_{2}\end{cases}
$$

For finite $T$ one may expect the values for $\tau_{1}=\tau_{2}$ to be much larger than those for $\tau_{1} \neq \tau_{2}$ and moreover those for $\tau_{1}=\tau_{2}=\tau_{a}$ to be approximately equal to those for $\tau_{1}=\tau_{2}=\tau_{b} \neq \tau_{a}$. How good these approximations are can be influenced by the input design.

The matrix $Q$ can apparently be made approximately equal to the identity by extending the input sequence with zeros. Unfortunately it requires knowledge of the process transfer to tell how the process will respond to this extended sequence. As the aim of the identification was to find an approximate description of this transfer, this information is not available yet, but one may assume that the largest time constant in the process transfer is known already: this knowledge is required to decide how long the experimental data record should be. One may assume that two or three times the largest time constant after the input was fixed to zero the output of the process would have become zero in good approximation as well. This divides the extended data set in three parts: first the original experimental data records, then a change-over part, in which the process inputs are zero and the process outputs settle to zero and finally the steady state part in which both inputs and outputs are (approximately) zero.

$$
U=\left[\begin{array}{ccc}
{\left[B_{0}(\zeta) u\right](0)} & \cdots & {\left[B_{\bar{k}}(\zeta) u\right](0)} \\
\vdots & & \vdots \\
{\left[B_{0}(\zeta) u\right]\left(T^{\prime}-1\right)} & \cdots & {\left[B_{\bar{k}}(\zeta) u\right]\left(T^{\prime}-1\right)}
\end{array}\right]=\left[\begin{array}{c}
U_{\text {exp }} \\
U_{\text {tra }} \\
U_{\text {ss }}
\end{array}\right]
$$

where the partitioning of $U$ in $U_{\text {exp }}, U_{\text {tra }}$ and $U_{s s}$ corresponds to the separation of the extended data set into three parts. Provided the extension of the data set is long enough, it will hold

$$
Q=\frac{1}{T} U^{T} U=\frac{1}{T}\left(U_{\exp }^{T} U_{\exp }+U_{t r a}^{T} U_{t r a}+U_{s s}^{T} U_{s s}\right) \approx I
$$

if the input that is applied to the process during the experiment interval is a white noise sequence with standard deviation one.

Consider $Y$ for the zeroes-extended data set. Let $Y_{\exp }, Y_{\text {tra }}$ and $Y_{s s}$ be a partitioning of $Y$ compatible with that of $U$ in $U_{\text {exp }}, U_{\text {tra }}$ and $U_{s s}$. $Y_{\text {exp }}$ is known, $Y_{\text {tra }}$ is unknown and $Y_{s s}$ is assumed to be (approximately) zero. Leaving the unknown part of the output out of the identification criterion, the optimal model parameter vector should satisfy

$$
\hat{\boldsymbol{\theta}}=\arg \min _{\boldsymbol{\theta}}\left\|\left[\begin{array}{c}
Y_{\exp }  \tag{5.9}\\
Y_{s s}
\end{array}\right]-\left[\begin{array}{c}
U_{\exp } \\
U_{s s}
\end{array}\right] \theta\right\|_{2}
$$

As in the unextended case, a projection argument can be used to show that it holds

$$
\begin{align*}
\hat{\boldsymbol{\theta}} & =\left(\left[\begin{array}{c}
U_{\text {exp }} \\
U_{\mathrm{ss}}
\end{array}\right]^{T}\left[\begin{array}{c}
U_{\text {exp }} \\
U_{\mathrm{ss}}
\end{array}\right]\right)^{-1}\left[\begin{array}{c}
U_{\text {exp }} \\
U_{\mathrm{ss}}
\end{array}\right]^{T}\left[\begin{array}{c}
Y_{\text {exp }} \\
Y_{\mathrm{ss}}
\end{array}\right] \\
& =\left(U_{\text {exp }}^{T} U_{\text {exp }}+U_{\mathrm{ss}}^{T} U_{\mathrm{ss}}\right)^{-1} U_{\text {exp }}^{T} Y_{\text {exp }} \tag{5.10}
\end{align*}
$$

Comparing this expression with the one for the unextended case, it appears that an extra term $U_{\text {ss }}^{T} U_{s s}$ has been added to $U_{\text {exp }}^{T} U_{\text {exp }}$. This term is likely to have a regularising effect, as it was derived previously that

$$
\frac{1}{T} U_{e x p}^{T} U_{\mathrm{exp}}+\frac{1}{T} U_{\mathrm{ss}}^{T} U_{\mathrm{ss}} \approx I-\frac{1}{T} U_{t r a}^{T} U_{t r a}
$$

for white input sequences $u$.
Another interpretation of this estimation procedure can be given based on (5.9). In the original, unextended case there was no penalty for large system outputs after $t=T$. In the new scheme model outputs that are unequal to zero for $t=T, \ldots, T^{\prime}-1$ are penalised. This also indicates how $T^{\prime}$ should be chosen: $T^{\prime}-T$ should be taken such, that the longest of the impulse response sequences $b_{0}(t)$ to $b_{\bar{k}}(t)$ reaches steady state in this period of time.

So called ridge regression replaces the "standard" Least Squares estimate

$$
\hat{\boldsymbol{\theta}}=\left(U^{T} U\right)^{-1} U^{T} Y
$$

with the estimate

$$
\hat{\boldsymbol{\theta}}=\left(U^{T} U+c I\right)^{-1} U^{T} Y, \quad c \in \mathbb{R}_{+} .
$$

Generalised ridge regression uses the estimate

$$
\begin{equation*}
\hat{\boldsymbol{\theta}}=\left(U^{T} U+C\right)^{-1} U^{T} Y, \quad C \in \mathbb{R}^{p \times p}, C \text { diagonal } \tag{5.11}
\end{equation*}
$$

See (Draper and Van Nostrand, 1979) for a review of these and related regularisation methods. These expressions bear some resemblance to (5.10). The estimate (5.10) is more general in the sense that it drops the restriction on $C$ in (5.11) to be diagonal. Moreover, the problem of choosing $c$ or the diagonal elements of $C$ does not occur in (5.10). The only choice that needs to be made is where to draw the line between the transient effect, represented by $Y_{\text {tra }}$ and the (approximate) steady state situation of $Y_{\text {ss }}$. As has been argued already, this can be based on knowledge of the largest relevant time constant of system, which should be known prior to the experiment design.

It should be stressed, that almost nothing was added artificially in this refinement of the original least squares algorithm: the only part that was more or less made up is that $Y_{\mathrm{ss}}=0$ and that the system reaches this steady state


Figure 5.2: Step responses, '-': true system, '--': model estimated with regularised LS


Figure 5.3: Estimated model parameters
after two or three times the largest time constant of the system. These approximations are definitely very sensible.

To get an impression of the capabilities of the proposed modifications, the example of section 4.8 .2 is reconsidered here. The data set was extended to 8500 samples. Of the extra 7500 samples, the first 200 samples were considered a transient effect. Therefore $Y_{s s}$ consists of 7300 samples and $Y_{\text {tra }}$ of 200 samples. Figure 5.2 shows the step response of the model obtained in this way. Comparing these results to those shown in figure 4.4, it is clear that the results have improved significantly by the regularisation. For further comparison, figure 5.3 shows the model parameters. This figure shows also a big improvement over figure 4.6.

### 5.4 Prior knowledge of static gains



In practice it is not uncommon that prior knowledge exists with respect to the static gains of certain transfers. Especially if a process output is controlled by a primary controller that is designed such that there is no final error, the static gain from almost all inputs to the output is equal to zero. The only exception is the setpoint for that specific output, for which the static gain is equal to one. Prior knowledge of static gains may also result from step experiments. These experiments are often carried out to obtain information required for the design of the actual identification experiments.

In formal terms, the prior knowledge consists of values $\bar{h}^{w, \sigma}$ and $\underline{h}^{w, \sigma}$ for some operating point $w \in \mathcal{W}$ and for some sub-transfer $\sigma \in \mathcal{S}$ such that

$$
\begin{align*}
& G_{t r}^{w, \sigma}(z) \leq\left.\bar{h}^{w, \sigma}\right|_{z=1}  \tag{5.12}\\
& G_{t r}^{w, \sigma}(z) \geq\left.\underline{h}^{w, \sigma}\right|_{z=1} \tag{5.13}
\end{align*}
$$

The superscripts $w$ and $\sigma$ will further be dropped. This implies that all transfers do not have their normal MIMO meaning but denote only a SISO subtransfer.

From (5.12) follows

$$
\begin{gather*}
G_{t r}(1)=\sum_{k=0}^{\infty} \theta_{t r, k} B_{k}(1) \leq \bar{h} \Rightarrow \\
\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}(1) \leq \bar{h}+\left|\sum_{k=\bar{k}}^{\bar{k}} \bar{\theta}_{k} B_{k}(1)\right|+\left|\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} B_{k}(1)\right| \tag{5.14}
\end{gather*}
$$

The first term on the right hand side of (5.14) can be calculated without further modifications. The second term can be bounded by

$$
\left|\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k} B_{k}(1)\right| \leq \max _{k>\bar{k}}\left|B_{k}(1)\right| \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}
$$

The maximum can be found easily through property 4.1.
In the same way (5.13) yields

$$
\begin{equation*}
\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}(1) \geq \underline{h}-\left|\sum_{k=\bar{k}}^{\bar{k}} \bar{\theta}_{k} B_{k}(1)\right|-\left|\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} B_{k}(1)\right| \tag{5.15}
\end{equation*}
$$

The prior knowledge (5.12) and (5.13) of the true system translates to the linear inequality constraints (5.14) and (5.15) on the model parameters.

Note that the true static gain is not restricted to the interval $[\underline{h}, \bar{h}]$ by the inequalities (5.14) and (5.15). It can only be guaranteed that

$$
\begin{aligned}
& \underline{h}-2\left|\sum_{k=\bar{k}}^{\bar{k}} \bar{\theta}_{k} B_{k}(1)\right|-2\left|\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k} B_{k}(1)\right| \leq G_{t r}(1) \leq \\
& \leq \bar{h}+2\left|\sum_{k=\bar{k}}^{\bar{k}} \bar{\theta}_{k} B_{k}(1)\right|+2\left|\sum_{k=\bar{k}+1}^{\infty} \bar{\theta}_{k} B_{k}(1)\right|
\end{aligned}
$$

The factors 2 stem from the uncertainty in the sign of the contribution of the tail. In the determination of the upper bound (5.14) a negative sign has to be assumed to make sure that (5.14) always contains the true system. If the actual sign is positive, then the true static gain is larger than $\bar{h}$ by twice the contribution of the tail.

### 5.5 Prior knowledge of complex gains for arbitrary frequencies

```
Asmblmm
-Dovamainembs
```






```
-Construct set of linear constraints ............... p. . }10
    -< \overline{u},\mp@subsup{\overline{B}}{k}{},M,\vec{,},\mp@subsup{b}{k}{\prime}(t)
```





```
    - Add complex gain constraints .......................164
    \longrightarrow\mathcal{L}
```




The case of static gains treated in the previous section corresponds to prior knowledge of the complex gain for $\omega=0$. This situation is easier than the situation for all other frequencies in the range $\omega \in(-\pi, \pi)$, because the complex gain is known to reduce to a real gain.

As in the previous section, only SISO subtransfers will be considered. To fit into the framework of chapter 4 , the prior knowledge has to be translated to linear constraints on the model parameters. Not all prior knowledge can be translated to linear constraints. Also, due to the uncertainty in the tail of the basis, only an approximate description of the prior knowledge can be given.

The kinds of prior knowledge that will be treated are

1. An interval in which the phase of $G_{t r}$ is known to lie for a certain frequency $\omega^{\prime} \in(-\pi, \pi)$ :

$$
\arg G_{t r}\left(e^{j \omega^{\prime}}\right) \in[\underline{\phi}, \bar{\phi}]
$$

for known $\phi$ and $\bar{\phi}$. The size of the interval $[\phi, \bar{\phi}]$ may not exceed $\pi$, or the constraint corresponds to a non-convex region in the complex plane, which can not be described by a set of linear constraints that can be used in a linear programming problem. Using the convex hull instead offers no solution in this case because the convex hull is equal to $\mathbb{C}$.
2. A phase interval as in 1. combined with a lower bound on the absolute value of $G_{t r}$.
3. An upper bound for the absolute value of $G_{t r}$ :

$$
\left|G_{t r}\left(e^{j \omega^{\prime}}\right)\right| \leq \bar{G}
$$

4. An estimate for the gain together with a bound on the uncertainty:

$$
\left|G_{t r}\left(e^{j \omega^{\prime}}\right)-\tilde{G}\right|<\tilde{d G}
$$

The estimate $\tilde{G}$ need not be related to the nominal model or the auxiliary model for the operating point at hand.
3. is a special case of 4 . with $\tilde{G}=0$ and $\overline{d G}=\bar{G}$. Therefore 3 will not be treated separately.

In the following sections it will be shown how each of these types of prior knowledge can be translated - approximately - to linear constraints on the model parameters. These constraints can be added to $\mathcal{L}_{\boldsymbol{\theta}}$, after which the algorithm of chapter 4 can be applied as before.

### 5.5.1 Phase interval

In this section it will be assumed that $G_{t r}\left(e^{j \omega^{\prime}}\right) \neq 0$. This is mostly to ease the presentation, as the argument of $G_{t r}\left(e^{j \omega^{\prime}}\right)$ is undetermined for $G_{i r}\left(e^{j \omega^{t}}\right)=0$.

The actual value of $\arg G_{t r}\left(e^{j \omega^{\prime}}\right)$ is only determined modulo $2 \pi$. However, it is assumed that $\bar{\phi}>\phi$ and $\bar{\phi}-\phi<2 \pi$. If the first inequality does not
hold, the interval of allowable phase shifts is empty, which can not happen in practice. If the second inequality does not hold then any direction in the complex plane is allowed, so the constraints do not form a true restriction and can be ignored. Apart from this fundamental consideration it is also required that $\bar{\phi}-\phi \leq \pi$, due to limits imposed by the translation to linear constraints on the model parameters.


Figure 5.4: Phase bounds on $G_{t r}\left(e^{j \omega^{\prime}}\right)$

In figure 5.4 the situation has been drawn for specific values of $\omega^{\prime \prime}, \bar{\phi}$ and $\phi$. The shaded area indicates values that can not be assumed by $G_{t r}\left(e^{j \omega^{\prime}}\right)$. It holds

$$
\left.\begin{array}{l}
\arg G_{t r}\left(e^{j \omega^{\prime}}\right) \leq \bar{\phi} \wedge  \tag{5.16}\\
\arg G_{t r}\left(e^{j \omega^{\prime}}\right) \geq \phi
\end{array}\right\} \Leftrightarrow\left\{\begin{array}{l}
\operatorname{Im}\left(e^{-j \phi} G_{t r}\left(e^{j \omega^{\prime}}\right)\right) \leq 0 \\
\operatorname{Im}\left(e^{-j \phi} G_{t r}\left(e^{j \omega^{\prime}}\right)\right) \geq 0
\end{array}\right.
$$

It can also be seen from this figure that the equivalence does not hold if $\bar{\phi}-\phi>\pi$. The equivalence (5.16) holds neither in case $G_{t r}\left(e^{j \omega^{\prime}}\right)=0$. The left hand side is undetermined in that case, whereas the right hand side is satisfied.

Restricting attention to the first inequality on the right of (5.16), it holds

$$
\begin{align*}
& \operatorname{Im}\left(e^{-j \bar{\phi}} G_{t r}\left(e^{j \omega^{\prime}}\right)\right) \leq 0 \\
& \Leftrightarrow \operatorname{Im}\left(e^{-j \phi} \sum_{k=0}^{\infty} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right) \leq 0 \tag{5.17}
\end{align*}
$$

$$
\begin{aligned}
& \Rightarrow \operatorname{Im}\left(e^{-j \bar{\phi}} \sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right) \leq \\
& \Rightarrow \operatorname{Im}\left(\sum _ { k = \overline { k } + 1 } ^ { \tilde { k } } \theta _ { t r , k } B _ { k } ( e ^ { j \omega ^ { \prime } } ) \left|+\left|\sum_{k=\bar{k}+1}^{\infty} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right|\right.\right. \\
& \left.\quad \sum_{k=\bar{k}+1}^{-j \bar{\phi}} \sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right) \leq \\
& \bar{\theta}_{k}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right|+\max _{k>\bar{k}}^{\tilde{k}}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right| \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}
\end{aligned}
$$

where $\tilde{k} \geq \bar{k}$ is a truncation value. The second inequality on the right hand side of (5.16) implies in a similar way

$$
\begin{align*}
& \operatorname{Im}\left(e^{-j \phi} \sum_{k=0}^{\tilde{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right) \geq \\
&-\sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|B_{\tilde{k}}\left(e^{j \omega^{\prime}}\right)\right|-\max _{k>\tilde{k}}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right| \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \tag{5.18}
\end{align*}
$$

In figure 5.5 the values for $G_{\hat{\boldsymbol{\theta}}}\left(e^{j \omega}\right)$ satisfying (5.17) and (5.18) are shown. The offsets from the continuous straight lines to the dotted straight lines, indicated by an $=$ sign, are equal to the right hand side of (5.17). The dark shaded area contains those points that do not satisfy either (5.17) or (5.18) or both. The light shaded area contains values of $G_{\hat{\theta}}\left(e^{j \omega^{\prime}}\right)$ that do satisfy (5.17) and ( 5.18 ) but that are not consistent with the prior knowledge and the uncertainty in the tail. By adding additional linear constraints, the size of this extra. included area can be reduced.

### 5.5.2 Gain lower bound

An (absolute) gain lower bound in itself constitutes a non-convex region in complex plain. As already mentioned on page 165 this can not be translated to constraints for a linear programming problem. The convex hull of this region is $\mathbb{C}$; using the convex hull as a basis for the linear constraints required by the LP problem is obviously not a solution. By combining the gain lower bound with a phase constraint as in the previous section, it is possible to arrive at a set of linear constraints that describes approximately the prior knowledge at hand.


Figure 5.5: Translation of phase bounds to permissible values of $G_{\hat{\boldsymbol{\theta}}}\left(e^{j \omega^{\prime}}\right)$

The gain lower bound is taken to be

$$
\left|G_{t r}\left(e^{j \omega^{\prime}}\right)\right| \geq \underline{G}
$$

In figure 5.6 the values of $G_{t r}\left(e^{j \omega^{\prime}}\right)$ that are consistent with the phase bounds


Figure 5.6: Phase bounds and gain lower bound on $G_{t r}\left(e^{j \omega^{\prime}}\right)$
$\bar{\phi}$ and $\phi$ and the gain bound $\underline{G}$ are indicated by the unshaded area.
The derivation concerning the phase bounds is identical to the one of the previous section and will not be repeated here. For the gain lower bound it holds

$$
\left|G_{t r}\left(e^{j \omega^{\prime}}\right)\right| \geq \underline{G}
$$

$$
\begin{align*}
& \Leftrightarrow\left|\sum_{k=0}^{\infty} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right| \geq \underline{G} \\
& \Rightarrow\left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right| \geq \underline{G}-\left|\sum_{k=\tilde{k}+1}^{\tilde{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right|-\left|\sum_{k=\bar{k}+1}^{\infty} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right| \\
& \Rightarrow\left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right| \geq \underline{G}-\sum_{k=\bar{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right|-\max _{k>\bar{k}}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right| \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \tag{5.19}
\end{align*}
$$



Figure 5.7: Translation of phase bounds and gain lower bound to permissible values of $G_{\hat{\boldsymbol{\theta}}}\left(e^{j \omega^{\prime}}\right)$

The lower bound on the absolute value of $\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)$ is not a linear constraint. Figure 5.7 shows how this can be approximated by a linear constraint. Let $\bar{\mu}(\omega)$ be defined as

$$
\begin{equation*}
\bar{\mu}\left(\omega^{\prime}\right):=\sum_{k=\tilde{k}+1}^{\tilde{k}} \bar{\theta}_{k}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right|+\sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k} \max _{k>\tilde{k}}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right| \tag{5.20}
\end{equation*}
$$

for some truncation value $\tilde{k} \geq \bar{k}$. Note that this definition is consistent with the earlier definition of $\bar{\mu}(\omega)$ in (4.45) on page 118. The straight line in figure 5.7 representing the gain lower bound goes through the points

$$
z_{1}:=\left(\underline{G}-\bar{\mu}\left(\omega^{\prime}\right)\right) e^{j \bar{\phi}} \text { and } z_{2}=\left(\underline{G}-\bar{\mu}\left(\omega^{\prime}\right)\right) e^{j \underline{\phi}}
$$

One can verify with elementary mathematics that the set of points on this line


Figure 5.8: Phase bounds and gain lower bound on $G_{t r}\left(e^{j \omega^{t}}\right)$
is

$$
\left\{z \in \mathbb{C} \mid \operatorname{Re}\left(z e^{-j(\bar{\phi}+\phi) / 2}\right)=\operatorname{Re}\left(z_{1} e^{-j(\bar{\phi}+\phi) / 2}\right)\right\}
$$

and all points in the set

$$
\left\{z \in \mathbb{C} \mid \operatorname{Re}\left(z e^{-j(\bar{\phi}+\phi) / 2}\right) \geq \operatorname{Re}\left(z_{1} e^{-j(\bar{\phi}+\phi) / 2}\right)\right\}
$$

satisfy the original bound on the amplitude. Substituting the expression for $z_{1}$ and taking $z=G_{\hat{\theta}}\left(e^{j \omega}\right)$, this yields the following constraint on the process parameters:

$$
\begin{equation*}
\sum_{k=0}^{\bar{k}} \hat{\theta}_{k} \operatorname{Re}\left(B_{k}\left(e^{j \omega^{\prime}}\right) e^{-j(\bar{\phi}+\underline{\phi}) / 2}\right) \geq\left(\underline{G}-\bar{\mu}\left(\omega^{\prime}\right)\right) \cos ((\bar{\phi}-\underline{\phi}) / 2) \tag{5.21}
\end{equation*}
$$

Equation (5.21) together with (5.17) and (5.18) constitute the approximate description of the phase bounds and amplitude lower bound in the form of linear constraints on the model parameters.

### 5.5.3 Uncertain complex gain estimate

The prior knowledge

$$
\left|G_{t r}\left(e^{j \omega^{\prime}}\right)-\tilde{G}\right| \leq \overline{d G}
$$

represents an area in complex plane drawn in figure 5.8. The uncertainty in the
model amplitude due to the unknown tail of the basis is bounded as follows:

$$
\begin{align*}
& \left|G_{t r}\left(e^{j \omega^{\prime}}\right)-\tilde{G}\right| \leq \overline{d G} \\
& \Leftrightarrow\left|\sum_{k=0}^{\infty} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)-\tilde{G}\right| \leq \overline{d G}  \tag{5.22}\\
& \Rightarrow\left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)-\tilde{G}\right| \leq \\
& \quad \overline{d G}+\left|\sum_{k=\bar{k}+1}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right|+\left|\sum_{k=\tilde{k}+1}^{\infty} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right| \\
& \Rightarrow\left|\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)-\tilde{G}\right| \leq \\
& \quad \overline{d G}+\sum_{k=\bar{k}+1}^{\bar{k}} \bar{\theta}_{k}\left|B_{k}\left(e^{j \omega^{\prime}}\right)\right|+\max _{k>\bar{k}}\left|B_{k}\left(e^{j \omega}\right)\right| \sum_{k=\tilde{k}+1}^{\infty} \bar{\theta}_{k}
\end{align*}
$$

The same approach as in section 4.12 .1 is used to approximate this area by a set of linear constraints. An outer bounding polygon with $m$ vertices, $m \geq 3$ will approximate the circle in complex plane. The line partly coinciding with the edge of the polytope bounding the component in the direction $e^{j \phi}, \phi \in[0,2 \pi)$ is given by

$$
\left\{z \in \mathscr{C} \mid \operatorname{Re}\left(e^{-j \phi}(z-\tilde{G})\right)=\tilde{d} G+\bar{\mu}\left(\omega^{\prime}\right)\right\}
$$

with $\mu(\omega)$ as in (5.20). Taking $\phi=2 \pi(\ell-1) / m, \ell \in m$, the set of $m$ linear constraints representing the outer bounding polytope becomes

$$
\begin{equation*}
\forall \ell \in m \quad \operatorname{Re}\left(e^{-j 2 \pi(\ell-1) / m}\left(-\tilde{G}+\sum_{k=0}^{\bar{k}} \theta_{t r, k} B_{k}\left(e^{j \omega^{\prime}}\right)\right)\right) \leq \overline{d G}+\bar{\mu}\left(\omega^{\prime}\right) \tag{5.23}
\end{equation*}
$$

The approximations involved are shown in figure 5.9 for $m=8$.


Figure 5.9: Translation of gain upper bound to permissible values of $G_{\hat{\theta}}\left(e^{j \omega}\right)$

### 5.6 Weighting of parameters



In section 4.9 the model parameters of the structured error components are determined as the left singular vectors of a matrix

$$
\begin{equation*}
X:=\operatorname{sbs}_{w \in \mathcal{W}} \Delta \theta^{w} \tag{5.24}
\end{equation*}
$$

where $\Delta \theta^{w}$ are the differences between the parameters of a central model $\theta^{c}$ and the parameter vectors of auxiliary models for the operating points $w$. In this section a number of situations will be discussed which give rise to weighting of the rows or columns of $X$. Also the meaning of the right singular vectors of $X$ will be investigated.

### 5.6.1 Weighting of model parameters

In chapter 4 no attention has been paid to the preprocessing of the experimental input and output signals. It was assumed that the data sets had already been detrended, outliers had been removed and the different signals had been scaled properly. All of these steps are crucial for obtaining good models, irrespective of whether the models are used as "the" nominal model, as an auxiliary model for an operating point or otherwise.

Many approaches can be used for the scaling of the signals. Some of them are listed below.

1. In practical processes, both actuators and sensors have a limited range. These ranges correspond to ranges on the inputs and the outputs. Signals may be scaled such that these ranges correspond to the interval $[-1,1]$, where it is assumed for simplicity that these ranges are symmetric around zero. As the signals have already been detrended, this zero value corresponds in physical terms to the value of the trend.
2. For identification, signals are often scaled such that they all have a variance of one.
3. Another meaningful scaling in identification is a scaling such, that the noise level becomes the same in all signals. Depending on the identification framework used, this may be applicable to the outputs only or to both inputs and outputs.

The third scaling is often aimed for by applying the second. This uses the implicit assumption that the signal to noise ratio is the same in all signals.

Consider a $q \times p$ model with for each subtransfer a model order $\bar{k}$ :

$$
G_{\boldsymbol{\theta}}(\zeta)=\left[\begin{array}{ccc}
\sum_{k=0}^{\bar{k}} \theta_{k}^{(1,1)} B_{k}^{(1,1)}(\zeta) & \ldots & \sum_{k=0}^{\bar{k}} \theta_{k}^{(1, p)} B_{k}^{(1, p)}(\zeta)  \tag{5.25}\\
\vdots & & \vdots \\
\bar{k} & \\
\sum_{k=0}^{(q, 1)} B_{k}^{(q, 1)}(\zeta) & \ldots & \sum_{k=0}^{\bar{k}} \theta_{k}^{(q, p)} B_{k}^{(q, p)}(\zeta)
\end{array}\right]
$$

The ordering of the elements in the parameter vector $\boldsymbol{\theta}$ is free, as long as the same ordering is used consistently. Let

$$
\theta^{\sigma}=\left[\begin{array}{lll}
\theta_{0}^{\sigma} & \ldots & \theta_{\bar{\sigma}}^{\sigma}
\end{array}\right]^{T}
$$

be a "block" containing the model parameters for subtransfer $\sigma, \sigma \in \mathcal{S}$. A possible ordering for $\boldsymbol{\theta}$ is then

$$
\boldsymbol{\theta}=\left[\begin{array}{llllll}
\left(\boldsymbol{\theta}^{(1,1)}\right)^{T} & \ldots & \left(\boldsymbol{\theta}^{(q, 1)}\right)^{T} & \left(\boldsymbol{\theta}^{(1,2)}\right)^{T} & \ldots & \left(\boldsymbol{\theta}^{(q, p)}\right)^{T}
\end{array}\right]^{T}
$$

Obviously, if the $i$ th input signal, $i \in \boldsymbol{p}$, is scaled by a factor $f_{u, i}$, all model parameters occurring in the $i$ th column of (5.25) should scale by a factor $1 / f_{u, i}$ in order to keep the model equivalent to the unscaled version. If the $j$ th output is scaled by a factor $f_{y, j}, j \in \boldsymbol{q}$, then the model parameters occurring in the $j$ th row of (5.25) scale along by a factor $f_{y, j}$.

Definition 5.1 Two models using different scalings for inputs and/or outputs are called equivalent or physically identical if the sequence of

1. scaling the input signals corresponding to the scaling used by the model,
2. filtering the scaled inputs by the respective model and
3. scaling the model outputs back to unscaled values
produces identical output signals for both models.

Remark 5.4 Given the informal definition 5.1 it is straightforward to formalise the concept of model equivalence. This would involve extra notational clutter because the scaling has to be made explicit in the notation. Notwithstanding that this would remove some ambiguity that is now present in the notation, it is felt that this would not enhance the readability of this exposition.

The definition does not allow for different initial conditions. Again, this can be fixed at the expense of a more involved definition.

In the algorithm of chapter 4 the auxiliary models identified with different scaling factors for inputs and/or outputs will in general not be equivalent. Nonequivalent auxiliary models will result in non-equivalent central models and in different parameter vectors $\Delta \boldsymbol{\theta}^{w}$. The left singular vectors of $X$ in (5.24) will in turn change as well, so that finally the non-equivalent identified auxiliary models result in non-equivalent structured error components.

One can easily make the identified models independent of this scaling by changing to a scaling for both inputs and outputs that is fixed in some way, identifying the models and then transforming the models back to the original scaling. This will make the parameter vectors of the auxiliary models equivalent in the sense of definition 5.1. This in turn means that a matrix $X$ as in (5.24) obtained for one scaling is equivalent to a matrix $X^{\prime}$ obtained for a different scaling, where equivalence of parameter matrices is obtained as a straightforward extension of definition 5.1: a matrix can be transformed into a matrix with which it is equivalent by a scaling of its rows. Which rows scale depends on which inputs and outputs have different scalings. An important observation is, that for two equivalent matrices the matrices of their left singular vectors are not equivalent. So even if the identification procedure for the
auxiliary models is independent of scaling in the sense that it yields equivalent models, the structured error components will not be equivalent.

Apparently scaling is a design parameter for the algorithm. This holds a forteriori if one realises that the scaling used for identification need not be identical to the one used in the principal component analysis. As can already be concluded from the short and incomplete list of scalings at the beginning of this section, there is no such thing as the optimal scaling. This holds for identification and it will turn out to apply to the principal component analysis as well.

In the next section some considerations will be given concerning the interaction between scaling and structured error components. Also the scaling will be generalised to weightings of the rows of $X$ that do not necessarily correspond to scaling of inputs and outputs any more.

## Choice of weighting factors

Scaling on a per input basis or a per output basis, as discussed above, has a clear interpretation. It may be worthwhile to generalise the scaling of the rows of $X$ to scalings, or weightings, per subtransfer. One may have concluded from the results of a robust control design or by physical reasoning that the bottleneck in the robustness of the control design is the uncertainty in one or two subtransfers. By up-scaling the parameters of these subtransfers they will become more pronounced in the structured components and thus will be bounded in a more detailed way. (Another approach would be to explicitly specify the structured error component. This will be treated in section 5.7.) Scaling on a per parameter basis can also be considered. It is difficult to find a reason for doing so.

If the rows of $X$ are weighted differently prior to principal component anal$y$ sis and are scaled back after the analysis, the $\theta_{A, i}, i \in n_{s}$, need not be orthonormal any more. Loosing normality is not so important; it requires normalising some of the formulas, especially those in sections 4.11 and 4.12 . Loosing orthogonality may provide more reason for concern. Orthogonality in parameter space may be interpreted as a form of independence of the structured error components. However, this rather intuitive argument can be applied to both the scaled and the unscaled parameter space, but the conclusions to be drawn are different in physical terms. It is therefore worthwhile to investigate whether orthogonality in parameter space corresponds to some meaningful notion in physical terms.

Consider first the situation of two SISO models having orthonormal parameter vectors $\boldsymbol{\theta}_{1}$ and $\boldsymbol{\theta}_{2}$ in the unscaled case. Because of the orthonormality of the unscaled basis functions it holds

$$
\left(\boldsymbol{\theta}_{1} \mid \boldsymbol{\theta}_{2}\right)=\left(\sum_{k=0}^{\bar{k}} \theta_{1, k} B_{k}\left(e^{j \omega}\right) \mid \sum_{k=0}^{\bar{k}} \theta_{2, k} B_{k}\left(e^{j \omega}\right)\right)=0
$$

The first inner product is the standard inner product in $\mathbb{R}^{\bar{k}+1}$ and the second the standard inner product in $\mathcal{H}_{2}$. So the models $G_{1}\left(e^{j \omega}\right)$ and $G_{2}\left(e^{j \omega}\right)$ are orthogonal to each other and the same holds for their impulse responses $g_{1}(t)$ and $g_{2}(t)$. Now consider a combination of these two models

$$
g^{\prime}(t):=\alpha g_{1}(t)+\beta g_{2}(t)
$$

The energy in the impulse response of $g^{\prime}(t)$ is

$$
\begin{align*}
\left(g^{\prime}(t) \mid g^{\prime}(t)\right) & =\alpha^{2}\left(g_{1}(t) \mid g_{1}(t)\right)+2 \alpha \beta\left(g_{1}(t) \mid g_{2}(t)\right)+\beta^{2}\left(g_{2}(t) \mid g_{2}(t)\right) \\
& =\alpha^{2}\left\|g_{1}(t)\right\|_{h_{2}}^{2}+\beta^{2}\left\|g_{2}(t)\right\|_{h_{2}}^{2} \tag{5.26}
\end{align*}
$$

Apparently, the energy in $g^{\prime}(t)$ due to $\alpha g_{1}(t)$ is never compensated by the term $\beta g_{2}(t)$ and vice versa. This can be interpreted as a way of independence. From the point of view of reduction of conservatism, it seems worthwhile that none of the structural error components $G_{A, i}(\zeta), i \in \boldsymbol{n}_{\boldsymbol{s}}$, gets compensated by another structural error component $G_{A, j}(\zeta), j \in n_{s}, j \neq i$.

This form of independence of the error components is lost if the parameters are scaled on a per parameter basis. This scaling will therefore not further be considered. In (5.26) only the orthogonality of $g_{1}(t)$ and $g_{2}(t)$ was used, not the fact that they were normalised. The independence is therefore maintained if all parameters are scaled by the same factor.

Moving on to MIMO models $G_{1}(\zeta)$ and $G_{2}(\zeta)$, it holds for the parameter vectors $\boldsymbol{\theta}_{1}$ and $\boldsymbol{\theta}_{2}$

$$
\begin{equation*}
\left(\boldsymbol{\theta}_{i} \mid \boldsymbol{\theta}_{j}\right)=\sum_{\sigma \in S}\left(\boldsymbol{\theta}_{i}^{\sigma} \mid \boldsymbol{\theta}_{j}^{\sigma}\right), \quad i, j \in\{1,2\} \tag{5.27}
\end{equation*}
$$

It should be noted that the values per subtransfer scale along with the squares of the parameter scaling. Whether or not two parameter vectors are orthogonal therefore depends on the scalings per subtransfer. As was argued above, this can be used to influence the outcome of the principal component analysis. There is no universal physical interpretation for (5.27) providing an answer as to how to scale the parameters of a MIMO transfer.

The answer to that question should depend on an interaction between prior knowledge and the results of principal component analysis. The scaling is likely to have a corrective nature: if some aspects of the process uncertainty do not come out of the principal component analysis as could be expected from physical reasoning and if this can be tracked down to an unfortunate scaling of the model parameters, an extra scaling applied before the principal component analysis can compensate for this.

To give some extra guidance as to what initial weighting to choose, three options are given with their interpretation:

- No separate weighting can be applied to the parameter vectors. This assumes that the weighting that is suitable for identification of the auxiliary
models is also appropriate for the determination of the structured error components.
- Scale the $\Delta \boldsymbol{\theta}^{w}$ inversely proportionally to the parameters of the central model, $\boldsymbol{\theta}^{\boldsymbol{C}}$. As it was argued before that the weighting should be the same for all parameters of a subtransfer, some averaging has to be carried out over all parameters for a subtransfer of the central model. Let $\boldsymbol{\theta}^{c, \sigma}, \sigma \in \mathcal{S}$, be the sub-vector of $\boldsymbol{\theta}^{c}$ containing only the parameters pertaining to the subtransfer $\sigma$ and let $\Delta \boldsymbol{\theta}^{w, \sigma}$ be the corresponding sub-vector of $\Delta \boldsymbol{\theta}^{w}, w \in$ $\mathcal{W}$. As the arguments above rely heavily on inner products and 2 -norms, the weighting

$$
\begin{equation*}
f^{\sigma}:=\frac{\left\|\boldsymbol{\theta}^{c, \sigma}\right\|_{2}}{\bar{k}^{\sigma}+1} \tag{5.28}
\end{equation*}
$$

is an obvious candidate. To apply this weighting, the values of $\Delta \boldsymbol{\theta}^{w, \sigma}$ have to be divided by $f^{\sigma}$ before building $X$.

The implicit assumption behind this weighting is that the contribution of a subtransfer to the structured error components should be roughly proportional to its contribution to the central model. Large subtransfers are expected to have a large uncertainty. To avoid that these large uncertainties completely dominate the smaller ones in the principal compcient analysis, this weighting can be used.

- Another approach to the previous idea is to assume that the uncertainty per subtransfer is roughly proportional to the variation over the different operating points in the parameters of $\Delta \boldsymbol{\theta}^{w, \sigma}$ A suitable measure for the variation of the parameters is

$$
\begin{equation*}
f^{\sigma}:=\frac{\| \|_{w \in \mathcal{W}}^{\operatorname{stack}} \Delta \boldsymbol{\theta}^{w, \sigma} \|_{2}}{n_{w}\left(\bar{k}^{\sigma}+1\right)} \tag{5.29}
\end{equation*}
$$

Note that this measure is sensitive to a bias component in $\Delta \boldsymbol{\theta}$. This is not a problem if the central model has been chosen according to (4.23), as the bias component will be zero then.
As in the previous weighting, the parameters of $\Delta \theta^{w, \sigma}$ have to be divided by $f^{\sigma}$ before building $X$.

- A combination of both approaches is given by

$$
f^{\sigma}:=\frac{\left\|\begin{array}{l}
\operatorname{stack}  \tag{5.30}\\
w \in \mathcal{W}
\end{array} \boldsymbol{\theta}^{w, \sigma}\right\|_{2}}{n_{w}\left(\bar{k}^{\sigma}+1\right)}
$$

In this case, both the variation in the parameters of a subtransfer and the size of these parameters determine the weighting of that subtransfer in the principal component analysis.

The argumentation above regarding which weightings to choose relies almost entirely on physical interpretation. As is indicated by the different choices that can be made, there is no general notion of optimality that can be specified in terms of, say, only the $\bar{d}_{i}$ and $\underline{d}_{i}, i \in \boldsymbol{n}_{s}$. In the context of identification for robust control design optimality corresponds obviously to optimal performance of the final controller, but this criterion is too complex to be used here. As an alternative it has been tried to provide means to tune the structured error components to what seems reasonable from a physical point of view.

## Incorporation of weightings in the algorithm

It will now be assumed that the weighting of the parameter vector to be applied in principal component analysis is available in the form of a matrix $W$ such that $W \boldsymbol{\theta}$ is the properly weighted form of the unweighted parameter vector $\boldsymbol{\theta}$. In the previous section interpretations have been given that all lead to a diagonal weighting matrices $W$. Note, that the following derivation is valid for any non-singular weighting matrix $W$. The weighting $W$ propagates as follows through the algorithm:

Let $X$ be as in (5.24), i.e. without any weighting. The singular value decomposition of $W X$ is

$$
\begin{equation*}
W X=U_{W} \Sigma_{W} V_{W}^{T} \tag{5.31}
\end{equation*}
$$

where $U_{W}$ and $V_{W}$ are orthogonal matrices and $\Sigma_{W}$ is a diagonal matrix containing the singular values $\sigma_{1} \geq \cdots \geq \sigma_{n_{w}} \geq 0$. In the unweighted case the structured error components $A_{i}(\zeta)$ were taken equal to the models having as their parameter vectors the first $n_{s}$ columns of $U$. In the weighted case, they are to be taken equal to the first $n_{s}$ columns of $W^{-1} U_{W}$. Note that the first $n_{s}$ columns of $U_{W}$ are the parameter vectors for the structured error components in "weighted parameter space" and these are still a set of orthonormal vectors.

In sections 4.11 and 4.12 the difference between a model $G_{\boldsymbol{\theta}^{\prime}}, \boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{\boldsymbol{\theta}}$, and the (partial) representation of the nominal model $G_{\hat{\theta}}$ was decomposed into parts accounted for by the structured error components $A_{i}(\zeta), i \in n_{s}$ on the one hand and parts to be accounted for by the unstructured error $\Delta(\zeta)$ on the other hand. (Recall that the representation $G_{\hat{\theta}}$ of the nominal model is only partial because the tail of $\hat{G}$ is not present in $G_{\hat{\theta}}$.) This decomposition used the orthonormality of the $\theta_{A, i}$ heavily. This should now be revised.

In this revision, the weighting $W$ is interpreted such, that orthogonality in the weighted parameter space corresponds to the kind of independence that is wanted among the $A_{i}(\zeta)$ and also between the $A_{i}(\zeta)$ on the one hand and $\Delta(\zeta)$
on the other. This means that wherever an orthogonal projection occurred in the sections 4.11 and 4.12 , this should be replaced by a transformation to weighted parameter space, an orthogonal projection and a transformation back to unweighted parameter space.

In section 4.11 it was derived for the unweighted case that

$$
\begin{equation*}
\forall i \in n_{s}, w \in \mathcal{W} \quad \bar{d}_{i}^{w}:=-\left(\hat{\boldsymbol{\theta}} \mid \boldsymbol{\theta}_{A, i}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}^{\prime} \mid \boldsymbol{\theta}_{A, i}\right) \tag{4.42}
\end{equation*}
$$

corresponding to a largest component in the direction of $\boldsymbol{\theta}_{\boldsymbol{A}, i}$ equal to

$$
\begin{equation*}
\overline{\boldsymbol{\theta}}_{A, i}:=\bar{d}_{i}^{w} \boldsymbol{\theta}_{A, i} . \tag{5.32}
\end{equation*}
$$

See also figure 4.10. According to the previous paragraph, all parameter vectors in (4.42) and (5.32) should be replaced by their weighted counterparts, yielding a weighted largest component

$$
W \overline{\boldsymbol{\theta}}_{A, i}=\left(-\left(W \hat{\boldsymbol{\theta}} \mid W \boldsymbol{\theta}_{\boldsymbol{A}, i}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(W \boldsymbol{\theta}^{\prime} \mid W \boldsymbol{\theta}_{A, i}\right)\right) W \boldsymbol{\theta}_{\boldsymbol{A}, i}
$$

The unweighted largest component then is

$$
\overline{\boldsymbol{\theta}}_{A, i}=\left(-\left(W \hat{\boldsymbol{\theta}} \mid W \boldsymbol{\theta}_{A, i}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}}\left(W \boldsymbol{\theta}^{\prime} \mid W \boldsymbol{\theta}_{A, i}\right)\right) \boldsymbol{\theta}_{A, i}
$$

so that equation (4.42) should read in case of a weighting $W$ on the model parameters

$$
\begin{equation*}
\bar{d}_{i}^{w t}:=-\left(W \hat{\boldsymbol{\theta}} \mid W \boldsymbol{\theta}_{A, i}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{C}_{\boldsymbol{\theta}}^{w}}\left(W \boldsymbol{\theta}^{\prime} \mid W \boldsymbol{\theta}_{A, i}\right) \tag{5.33}
\end{equation*}
$$

In the same way it is found that the original equation

$$
\begin{equation*}
\forall i \in n_{s}, w \in \mathcal{W} \quad \underline{d}_{i}^{w}:=-\left(\hat{\theta} \mid \boldsymbol{\theta}_{A, i}\right)+\min _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\theta^{w}}}\left(\boldsymbol{\theta}^{\prime} \mid \boldsymbol{\theta}_{A, i}\right) \tag{4.43}
\end{equation*}
$$

for the unweighted case should become

$$
\begin{equation*}
\forall i \in \boldsymbol{n}_{\boldsymbol{s}}, w \in \mathcal{W} \quad \underline{d}_{i}^{w}:=-\left(W \hat{\boldsymbol{\theta}} \mid W \boldsymbol{\theta}_{A, i}\right)+\min _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(W \boldsymbol{\theta}^{\prime} \mid W \boldsymbol{\theta}_{A, i}\right) \tag{5.34}
\end{equation*}
$$

in the weighted case.
The way in which the structured error components are removed from the unstructured error in section 4.12 also needs updating. Basically the structured error components were removed from the unstructured error by projecting all parameter vectors $\theta^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{\boldsymbol{w}}$ onto the orthoplement of the span of the structured error components. For this projection the operator $P_{A}^{\perp}$ was introduced as

$$
\begin{equation*}
P_{A}^{\perp}=I-\sum_{i=1}^{n_{s}} \boldsymbol{\theta}_{A, i} \boldsymbol{\theta}_{A, i}^{T} \tag{4.59}
\end{equation*}
$$

This becomes in the weighted case

$$
\begin{equation*}
P_{A}^{\perp}=W^{-1}\left(I-\sum_{i=1}^{n_{s}} W \boldsymbol{\theta}_{A, i}\left(W \boldsymbol{\theta}_{A, i}\right)^{T}\right) W=I-\sum_{i=1}^{n_{e}} \theta_{A, i} \boldsymbol{\theta}_{A, i}^{T} W^{T} W \tag{5.35}
\end{equation*}
$$

where it was used that $W^{T}=W$. In the middle expression in (5.35) we recognise, from right to left, the transformation to weighted parameter space through the matrix $W$, the original orthogonal projection and the transformation back to unweighted parameters through $W^{-1}$. The projection operator onto $\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i \in n,}$, becomes in the weighted case

$$
\begin{equation*}
P_{A}=\sum_{i=1}^{n_{s}} \boldsymbol{\theta}_{A, i} \boldsymbol{\theta}_{A, i}^{T} W^{T} W \tag{5.36}
\end{equation*}
$$

This change should also be reflected in the definition of $\mathcal{L}_{\theta}^{\prime}$ in (4.60) on page 128.
This concludes the changes that need to be made in the algorithm to allow for a weighting of the model parameters. If frequency domain bounds are required, no changes are required for the interpolation between frequencies for which the uncertainty has been bounded explicitly. The combination of local bounds to global bounds can remain unchanged as well.

### 5.6.2 Weighting of operating point parameters

In this section weighting of the columns of the matrix $X$ from equation (5.24) will be considered. Further, the matrix $V$ of right singular vectors in the singular value decomposition will be interpreted. This interpretation will be presented by means of a numerical example.

Let the matrix $X$ denote again a matrix having the $\Delta \theta^{w}$ as its columns,

$$
X=\operatorname{sbs}_{w \in \mathcal{W}} \Delta \theta^{w}
$$

and let $\tilde{X}$ be the matrix obtained by putting the $\theta^{w}, w \in \mathcal{W}$, next to each other,

$$
\tilde{X}=\operatorname{sbs}_{w \in \mathcal{W}} \theta^{w}
$$

Suppose that $\tilde{X}$ has for example the following numerical value

$$
\tilde{X}=\left[\begin{array}{rrrrrrrr}
1.03 & 1.00 & 0.04 & -0.04 & 0.00 & 0.01 & 0.34 & 0.31 \\
1.02 & 1.05 & -0.02 & 0.03 & 0.02 & 0.03 & 0.32 & 0.33 \\
0.00 & 0.01 & 1.02 & 1.00 & 0.01 & 0.02 & 0.33 & 0.34 \\
0.01 & 0.03 & 0.99 & 0.98 & 0.02 & -0.07 & 0.38 & 0.32 \\
-0.02 & -0.04 & 0.00 & -0.02 & 1.01 & 0.99 & 0.36 & 0.32 \\
0.05 & -0.02 & -0.04 & 0.03 & 0.96 & 0.99 & 0.34 & 0.35
\end{array}\right]
$$

This matrix was generated in MATLAB by adding a random perturbation matrix to the matrix

$$
X_{\text {off }}=\left[\begin{array}{llllllll}
1 & 1 & 0 & 0 & 0 & 0 & 1 / 3 & 1 / 3 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 / 3 & 1 / 3 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 / 3 & 1 / 3 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 / 3 & 1 / 3 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 / 3 & 1 / 3 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 / 3 & 1 / 3
\end{array}\right]
$$

where the entries of the perturbation matrix were normally distributed with zero mean and standard deviation of 0.03 . In this way it was achieved that the columns of $\tilde{X}$ are clustered in four pairs consisting of the first and second column, the third and fourth column, the fifth and sixth column and finally the seventh and eighth column.

After removal of the mean of the columns from $\tilde{X}$ this yields for $X$

$$
X=\left[\begin{array}{rrrrrrrr}
0.69 & 0.66 & -0.29 & -0.37 & -0.34 & -0.32 & 0.00 & -0.02 \\
0.67 & 0.72 & -0.35 & -0.30 & -0.32 & -0.30 & -0.03 & 0.00 \\
-0.35 & -0.33 & 0.69 & 0.67 & -0.32 & -0.31 & -0.01 & 0.01 \\
-0.34 & -0.31 & 0.66 & 0.65 & -0.31 & -0.40 & 0.04 & -0.01 \\
-0.37 & -0.38 & -0.33 & -0.35 & 0.68 & 0.66 & 0.01 & 0.00 \\
-0.30 & -0.36 & -0.37 & -0.30 & 0.62 & 0.66 & 0.00 & 0.02
\end{array}\right]
$$

The clustering in $\tilde{X}$ is maintained in $X$. Omitting for simplicity any weighting of the model parameters as discussed in the previous section, the singular value decomposition of $X$

$$
X=U \Sigma V^{T}
$$

is used to determine the structured error components. The following values are found for $U, \Sigma$ and $V$

$$
\left.\begin{array}{rl}
U & =\left[\begin{array}{rrrrrr}
0.57 & -0.05 & -0.48 & -0.47 & 0.24 & -0.41 \\
0.58 & -0.06 & 0.47 & 0.45 & -0.27 & -0.41 \\
-0.25 & 0.52 & 0.37 & -0.53 & -0.28 & -0.41 \\
-0.23 & 0.53 & -0.36 & 0.54 & 0.28 & -0.41 \\
-0.35 & -0.48 & -0.37 & 0.07 & -0.58 & -0.41 \\
-0.32 & -0.46 & 0.37 & -0.06 & 0.61 & -0.41
\end{array}\right] \\
\Sigma & =\left[\begin{array}{rrrrrrr}
2.0125 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 \\
0 & 1.9991 & 0 & 0 & 0 & 0 & 0 \\
0 \\
0 & 0 & 0.0976 & 0 & 0 & 0 & 0 \\
0 \\
0 & 0 & 0 & 0.0650 & 0 & 0 & 0 \\
0 \\
0 & 0 & 0 & 0 & 0.0385 & 0 & 0 \\
0 \\
0 & 0 & 0 & 0 & 0 & 0.0000 & 0
\end{array}\right]
\end{array}\right]
$$

and

$$
V=\left[\begin{array}{rrrrrrrr}
0.58 & -0.06 & 0.11 & -0.35 & 0.49 & -0.19 & 0.50 & 0.00 \\
0.59 & -0.03 & 0.21 & 0.25 & -0.68 & -0.25 & 0.02 & 0.08 \\
-0.23 & 0.54 & -0.28 & -0.49 & -0.39 & -0.19 & 0.36 & 0.17 \\
-0.24 & 0.52 & 0.64 & 0.35 & 0.19 & -0.26 & 0.16 & -0.11 \\
-0.33 & -0.46 & -0.14 & 0.38 & -0.11 & -0.07 & 0.71 & 0.04 \\
-0.31 & -0.48 & 0.47 & -0.49 & -0.14 & -0.40 & -0.16 & 0.03 \\
-0.01 & 0.00 & -0.37 & 0.24 & 0.26 & -0.69 & -0.26 & 0.45 \\
-0.01 & 0.00 & 0.29 & -0.01 & 0.04 & 0.40 & 0.05 & 0.87
\end{array}\right]
$$

Let $c$ be the number of parameters in $\theta$, in this example 6 . The columns of $\Sigma V^{T}$ or, equivalently, the rows of $V \Sigma^{T}$ specify to what extent the structured error components $\boldsymbol{\theta}_{A, i}=U_{* i}, i \in \boldsymbol{n}_{s}$, are present in $\Delta \boldsymbol{\theta}^{w}, w \in \mathcal{W}$. This stems from the equality

$$
\Delta \boldsymbol{\theta}^{w}=\sum_{i=1}^{c}\left[V \Sigma^{T}\right]_{w, i} U_{* i}
$$

Looking at

$$
V \Sigma^{T}=\left[\begin{array}{rr|rrrr}
1.17 & -0.12 & 0.01 & -0.02 & 0.02 & 0.00 \\
1.20 & -0.06 & 0.02 & 0.02 & -0.03 & 0.00 \\
-0.46 & 1.07 & -0.03 & -0.03 & -0.02 & 0.00 \\
-0.48 & 1.04 & 0.06 & 0.02 & 0.01 & 0.00 \\
-0.66 & -0.91 & -0.01 & 0.02 & 0.00 & 0.00 \\
-0.63 & -0.96 & 0.05 & -0.03 & -0.01 & 0.00 \\
-0.03 & 0.01 & -0.04 & 0.02 & 0.01 & 0.00 \\
-0.01 & -0.01 & 0.03 & 0.00 & 0.00 & 0.00
\end{array}\right]
$$

it is clear that the contribution of the last four columns of $U$ is dominated by that of the first two. (This can be concluded even clearer from $\Sigma$.) Looking at the columns of $V \Sigma^{T}$ corresponding to the two dominating columns of $U$, it is remarkable that the first row resembles the second, the third resembles the fourth and the the fifth resembles the sixth. This corresponds exactly to the clustering of the $\boldsymbol{\theta}^{\boldsymbol{w}}$ and the $\boldsymbol{\Delta} \boldsymbol{\theta}^{\boldsymbol{w}}$.

The clustering could have been discovered without using $V \Sigma^{T}$ by looking for examples at the angles between the vectors $\Delta \boldsymbol{\theta}^{w}, w \in \mathcal{W}$. In fact, because $U$ is an isometry the angles between the $\Delta \theta^{w}$ are identical to the angles between the rows of $V \Sigma^{T}$. But there is more to be seen from $V \Sigma^{T}$. Looking for example at the second column of $V \Sigma^{T}$ it can be seen that $\theta_{A, 2}$ is mainly present in $\left.\Delta \theta^{w}\right|_{w}, w \in\{3,4,5,6\}$. This means in turn, that the second structured error component is determined mainly by the auxiliary models for the third, fourth, fifth and sixth operating point. From the first column of $V \Sigma^{T}$ can be concluded that the first structured error component is determined mainly by the first and


Figure 5.10: Influence of weighting of operating points
second operating point. In this way the rows of $V \Sigma^{T}$ give a rough indication of which operating points are accounted for by the different structured error components. This information can be valuable for a physical interpretation of the structured error components.

It now can occur that a structured error component turns out to correspond mainly to, say, the auxiliary model for a single operating point, that happens to be based on a rather short data set with a relatively high noise level. In figure 5.10a this is illustrated for another set of example values for $\Delta \theta^{*}$ collected in the columns of

$$
X=\left[\begin{array}{rrrr}
-0.9921 & -1.0434 & 1.0374 & 5.0173 \\
1.0262 & 0.9790 & -1.0192 & 1.9892
\end{array}\right]
$$

These values of $\Delta \theta^{w}$ are indicated by ' $x$ ' symbols. The fourth $\Delta \theta^{w}$ has deliberately been chosen somewhat as an outlier. It largely determines the main principal component $U_{* 1}$. There is considerable structure in the remaining three operating point models, they all lie near the line $\theta_{1}+\theta_{2}=0$. Because of the dominating influence of the fourth operating point, this information is hardly present in the principal components. The direction of the line $\theta_{1}+\theta_{2}=0$ is indicated by the dashed line in the figure to facilitate comparison of the principal components to this direction.

The value of $V \Sigma^{T}$ is for this example

$$
V \Sigma^{T}=\left[\begin{array}{rr}
1.9076 & -0.6546 \\
1.9669 & -0.6179 \\
0.3011 & 1.7374 \\
-4.1756 & -0.4648
\end{array}\right]
$$

From the first column it is clear that $U_{* 1}$ is mainly determined by the auxiliary model of the fourth operating point. The first and second operating point seem to have a significant contribution. This is due to the fact that the central model is "pulled towards" $\left.\theta^{w}\right|_{w=4}$. As a result of this, the first and second operating point get a large error in this direction as well, but with an opposite sign. This can be recognised as well from the first column of $V \Sigma^{T}$.

If the faith in the fourth operating point model is low, for example due to a bad signal to noise ratio or a short data set, it seems reasonable to reduce the influence of this model on the structured error components. To achieve this, the weighting matrix

$$
W_{R}=\left[\begin{array}{llll}
1 & & & \\
& 1 & & \\
& & 1 & \\
& & & 0.1
\end{array}\right]
$$

and the scaled matrix

$$
X^{\prime}:=X W_{R}
$$

are introduced and the principal components are determined by a singular value decomposition of $X^{\prime}$ instead of $X$.

The reasoning behind the weighting is, that values of $\boldsymbol{\theta}^{w}$ that have a large influence on the singular value decomposition can have their influence reduced by moving them towards the center, in some sense, of the $\boldsymbol{\theta}^{w}$. The scaling is applied to $X$, so to the $\Delta \boldsymbol{\theta}^{w}$, and not to the $\boldsymbol{\theta}^{\boldsymbol{w}}$. This corresponds to a scaling of the distances between the $\boldsymbol{\theta}^{\boldsymbol{w}}$ and $\boldsymbol{\theta}^{c}: \boldsymbol{\theta}^{c}$ is taken as the center of the $\boldsymbol{\theta}^{\boldsymbol{w}}$. Because of this, this procedure is insensitive to a translation of the origin. This is a desirable feature, as the structure in the model uncertainty should reveal something about the position of the $\boldsymbol{\theta}^{\boldsymbol{w}}$ relative to each other, not relative to the origin.

The result of this weighting is shown in figure 5.10 b . The singular vectors $U_{* 1}$ and $U_{* 2}$ have changed, but $U_{* 1}$ is still not much in line with the structure in the first three operating points. This is due to the fact that after the simple weighting leading from $X$ to $X^{\prime}$ the columns of $X^{\prime}$ are not zero mean any more. The new mean value of the operating point parameter vectors has moved to $\boldsymbol{\theta}^{c i}$. The direction $\boldsymbol{\theta}^{c i}-\boldsymbol{\theta}^{c}$ has a large influence on $U_{* 1}$.

If the mean is subtracted from the columns of $X^{\prime}$ before performing the singular value decomposition the results shown in figure 5.11 are obtained. Now the first singular vector $U_{* 1}$ is dominated indeed by the structure in the first three operating points. The influence of the fourth operating point is still recognisable, so it has not been completely discarded.

It is straightforward to generalise the procedure presented in this section by means of an example to other situations. Basically it boils down to the


Figure 5.11: Influence of weighting and recentering of operating points
following steps: first, reduce the distance between the central model parameter vector and the parameter vector of an operating point that is undesirably dominating the principal component analysis. Then establish a new central model for the new set of operating point models and proceed with the principal component analysis as before. Of course, this procedure need not be restricted to a single auxiliary model. As many parameter vectors can be weighted as seems necessary.

There is insufficient information in the auxiliary model parameter vectors themselves for the algorithm to decide which values should be considered true system behaviour and which ones should be distrusted. It has already been mentioned that noise levels and lengths of data sets can be a reason to suspect models a priori. Also the results of any validation criterion can be used to establish the reliability of the models for different operating points. Different levels of reliability should be reflected in different weighting factors for the operating points.

Remark 5.5 One may have reservations regarding the quality of the new central model $\boldsymbol{\theta}^{c 1}$ that has been obtained by the previous procedure. However, things are not as bad as they may seem. Firstly, the new central model can be shown to be a weighted average of the original auxiliary models, with a lower weighting on models in which there is relatively little faith. Secondly, the central model $\theta^{c t}$ is only a by-product of the determination of the structured error components. Apart from this use, the algorithm never uses it as a model for the process, unless the user decides to use it as the nominal model for the process. In view of the first remark, it still seems better to use the central model $G_{\boldsymbol{\theta}^{c}}$ instead of $G_{\boldsymbol{\theta}^{c}}$.

### 5.7 Other ways to determine the error structure

```
Agormbm
```






```
- Estimate error structure ..................... p. 101
```





Up to this point, principal component analysis has been used to determine the structured error components. In the preceding section means were developed to tune this analysis to various kinds of insight we may have. In this section principal component analysis will be circumvented altogether. Alternative ways to determine structured error components and modifications that should be made to the algorithm will be discussed.

### 5.7.1 Overview of alternatives to principal component analysis

- The most important alternative way to find the dominating parts in the model uncertainty is probably through physical insight. Based on such knowledge of the plant, it can be possible in certain circumstances to indicate a priori what aspects of the plant behaviour will change with changes from one operating point to another. If this prior knowledge is detailed enough, it may be possible to translate it to $n_{s}$ structured error components $A_{i}(\zeta), i \in \boldsymbol{n}_{\boldsymbol{s}}$.
- In section 5.6 .1 it was already mentioned that it can be a sensible assumption for some plants that the uncertainty is roughly proportional to the nominal model: it can be expected that subtransfers that are large in the nominal model also have a large uncertainty. In fact, the previous two statements are not equivalent. Assuming that the uncertainty is proportional to the nominal model requires that the same dynamics should be present in the uncertainty. Assuming that one is large where the other is large does not require this.

In section 5.6.1, more specifically equation (5.28) on page 177 this was less of an issue, because the parameters for a subtransfer were averaged in (5.28) and these averages were used only as a weighting, not as a direct specification of the error components. Another, less important, difference with the situation here is that in section 5.6 .1 the central model was used instead of the nominal model. The nominal model instead of the central model could have been used in section 5.6.1 equally well.

Nevertheless it can be worthwhile to take $n_{s}=1$ and $\boldsymbol{\theta}_{A, 1}=\hat{\boldsymbol{\theta}}$. If the nominal model is in the model set used by this algorithm, in other words, if $\hat{G}(\zeta) \equiv G_{\hat{\theta}}(\zeta)$, the bounds $\bar{d}_{1}$ and $\underline{d}_{1}$ will give a bound on the multiplicative uncertainty:

$$
\begin{aligned}
G_{t r}(\zeta)= & (1+f) \hat{G}(\zeta)+\Delta(\zeta) \\
& f \in\left[\underline{d}_{1}, \bar{d}_{1}\right]
\end{aligned}
$$

Note that $f$ is a scalar, even for MIMO processes.

- A third approach is to let the results of control design dictate which parts of the model uncertainty should be bounded separately as structured error components. This is closely related to the first approach, based on physical insight: it probably requires physical insight to recognise what phenomenon is acting as a bottleneck in the control design. On the other hand, there is a different emphasis in this approach and the first one. The first approach tries to answer the question "Where can the model uncertainty be expected to be large?", whereas this approach tries to answer the more relevant but also more difficult question "Where can the model uncertainty be expected to have a large impact on control performance or robustness?"
- All ways to determine the model uncertainty have relied on information for more than one operating point. This does not mean that the algorithm is completely useless in case of only one operating point. It does mean, that one has to search for alternative ways to find the structured error components.
In principle, one can base the structured error components on the information that is present in the set of linear constraint $\mathcal{L}_{\boldsymbol{\theta}}$. If the polytope represented by this set is large in one direction and much smaller in the others, it is definitely worthwhile to bound the large direction separately, as a structured error component.
In fact, it is not only the size of the polytope itself that is relevant, it is also its position relative to the nominal model. The largest error in parameter space is given by

$$
\begin{equation*}
\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}}\left\|\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right\| \tag{5.37}
\end{equation*}
$$

where the Euclidean norm in parameter space has been used as a measure for the distance between two models. Equation (5.37) seems promising at first: it is a quadratic programming problem with linear constraints, and efficient algorithms exist to solve these problems. However, these algorithms solve the minimisation problem

$$
\min _{\boldsymbol{\theta}} \boldsymbol{\theta}^{T} P \boldsymbol{\theta}+c^{T} \boldsymbol{\theta}
$$

where $P$ is basically required to be a positive definite matrix. Generalisations exist for semi-definite $P$ and indefinite $P$, but especially in the latter case the algorithms get much less efficient. The problem (5.37) is in fact a minimisation with a negative definite $P$.

Using a geometrical argument one can see that the solution of (5.37) must be obtained for a $\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}$ that is a vertex of the polytope $\mathcal{L}_{\boldsymbol{\theta}}$. There seems to be no other solution method to (5.37) than simply traversing all the vertices of $\mathcal{L}_{\theta}$. For a moderately sized plant with reasonable data set lengths, $\mathcal{L}_{\theta}$ already contains thousands of constraints. Traversing all vertices is practically not feasible any more. With computers getting faster and faster, it may nevertheless be worthwhile to pursue this path a little further. What is not feasible at present may become possible in five or ten years. This situation may also benefit from the improvement of algorithms.

One need not restrict oneself to the largest direction only. Once the largest direction is found, the orthogonal complement may be searched for the second largest direction and so on, until the size of the largest vector found in a direction does not seem to qualify for the term "dominating" any more.

As mentioned in remark 4.13 on page 129 it is not necessary to project the polytope $\mathcal{L}_{\theta}$ itself onto the orthogonal complement of the structured error components found so far to find the next largest direction. Let $P_{A}^{\perp}$ denote the projection onto the orthoplement of the span of the structured error components found so far. The next one is obtained by solving

$$
\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}}\left\|P_{A}^{\perp}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right)\right\|=\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right)^{T}\left(P_{\boldsymbol{A}}^{\perp}\right)^{2}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right)
$$

which is a quadratic programming problem like (5.37).
The idea of using a QP-problem to find the dominating error components need not be restricted to the case of one operating point. In case of multiple operating points one should search in the polytope that is the convex hull of the union of the polytopes for different operating points. Finding the former polytope will require at the moment (also) an excessive computational load.

### 5.7.2 Modifications to the algorithm

It this section it will further be assumed that the structured error components are available in the form of $\boldsymbol{n}_{\boldsymbol{s}}$ independent vectors $\boldsymbol{\theta}_{A, i}, \boldsymbol{i} \in \boldsymbol{n}_{\boldsymbol{s}}$. The $\boldsymbol{\theta}_{A, i}$ are neither assumed to be normalised nor orthogonal.

It may seem that this information is sufficient to specify how a certain system $G^{\prime}(\zeta)$ should be written as

$$
\begin{equation*}
G^{\prime}(\zeta)=\hat{G}(\zeta)+\sum_{i=1}^{n_{s}} \delta_{i}^{\prime} A_{i}(\zeta)+\Delta(\zeta) \tag{5.38}
\end{equation*}
$$

This is not the case.
Consider a system $G_{\boldsymbol{\theta}^{\prime}}$ where $\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{\boldsymbol{\theta}}$ for an operating point $w \in \mathcal{W}$. For all $\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{\boldsymbol{w}}$, the difference $\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}$ should partly be accounted for through a sum $\sum_{i=1}^{n_{s}} \delta_{i}^{\prime} \theta_{A, i}$, corresponding to the second term on the right of (5.38), and partly be accounted for by some rest term, corresponding to $\Delta(\zeta)$ in (5.38). This implies that $\boldsymbol{\theta}^{\prime}$ should be written as

$$
\begin{equation*}
\boldsymbol{\theta}^{\prime}=\hat{\boldsymbol{\theta}}+\sum_{i=1}^{n_{s}} \delta_{i}^{\prime} \boldsymbol{\theta}_{A, i}+\boldsymbol{\theta}_{\Delta}^{\prime} \tag{5.39}
\end{equation*}
$$

where $\boldsymbol{\theta}_{\Delta}^{\prime}$ represents the part that should be accounted for by $\Delta(\zeta)$. (Recall that $\Delta(\zeta)$ also accounts for any mismatches in the tails. This means that $\boldsymbol{\theta}_{\Delta}^{\prime}=0$ does not imply $\Delta(\zeta)=0$.) If $\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}} \in\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i \in n_{s}}$ then there is a unique set of values $\delta_{i}^{\prime}, i \in \boldsymbol{n}_{s}$, such that

$$
\boldsymbol{\theta}^{\prime}=\hat{\boldsymbol{\theta}}+\sum_{i=1}^{n_{s}} \delta_{i}^{\prime} \boldsymbol{\theta}_{A, i}, \quad \boldsymbol{\theta}_{\Delta}^{\prime}=0
$$

If $\boldsymbol{\theta}^{\prime} \notin\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i \in \boldsymbol{n},}$ then a value $\boldsymbol{\theta}_{\Delta}^{\prime}$ has to be found, such that

$$
\boldsymbol{\theta}_{\|}^{\prime}:=\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_{\Delta}^{\prime} \in\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i \in n_{n}}
$$

For $\boldsymbol{\theta}_{\|}^{\prime}$ a projection of $\boldsymbol{\theta}^{\prime}$ onto the span of $\boldsymbol{\theta}_{A, i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$, will be used. From section 5.6 .1 it is clear that this projection need not be orthogonal in case of weighting and this is what causes the non-uniqueness of $\boldsymbol{\theta}_{\Delta}^{\prime}$ and the fact alluded to earlier that the directions $\theta_{A, i}$ alone do not determine uniquely the values of $\delta_{i}^{\prime}$ in (5.38). In this section the non-uniqueness will be resolved by using the orthogonal projection. If this is not desired, then the results obtained in this section can be combined with the results of section 5.6.1.

It will now be shown how the algorithm should be updated to reflect the choices made above.

Let the $\boldsymbol{\theta}_{A, i}$ be collected in a matrix $\Phi$ :

$$
\begin{equation*}
\Phi:=\operatorname{sbs}_{i \in n_{s}} \boldsymbol{\theta}_{A, i} \tag{5.40}
\end{equation*}
$$

and let $c$ be the number of entries in the parameter vectors $\theta$. The operator $P_{A}$ mapping an arbitrary vector $\boldsymbol{\theta} \in \mathbb{R}^{c}$ to its orthogonal projection onto img $\Phi$ is

$$
\begin{equation*}
P_{A}: \mathbb{R}^{c} \rightarrow \mathbb{R}^{c}, \quad \theta \mapsto \Phi\left(\Phi^{T} \Phi\right)^{-1} \Phi^{T} \theta \tag{5.41}
\end{equation*}
$$

and the operator $P_{A}^{\perp}$ mapping a $\boldsymbol{\theta}$ to its projection onto the orthoplement of $\operatorname{img} \Phi$ is

$$
\begin{equation*}
P_{A}^{\perp}: \mathbb{R}^{c} \rightarrow \mathbb{R}^{c}, \quad \boldsymbol{\theta} \mapsto\left(I-\Phi\left(\Phi^{T} \Phi\right)^{-1} \Phi^{T}\right) \boldsymbol{\theta} \tag{5.42}
\end{equation*}
$$

Note that these expressions reduce to (4.59) and (4.58) if the $\boldsymbol{\theta}_{A, i}$ form an orthonormal set of vectors. The remarks made above translate to the equalities

$$
\begin{equation*}
\boldsymbol{\theta}_{\|}^{\prime}=P_{A}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \tag{5.43}
\end{equation*}
$$

and

$$
\boldsymbol{\theta}_{\Delta}^{\prime}=P_{A}^{\perp}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) .
$$

Substitution of (5.40) and (5.41) in (5.43) gives

$$
\begin{aligned}
\boldsymbol{\theta}_{\|}^{\prime} & =\left[\begin{array}{lll}
\boldsymbol{\theta}_{A, 1} & \ldots \boldsymbol{\theta}_{A, n_{s}}
\end{array}\right]\left(\Phi^{T} \Phi\right) \Phi^{T}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \\
& =\sum_{i=1}^{n_{s}}\left[\left(\Phi^{T} \Phi\right)^{-1} \Phi^{T}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right)\right]_{i} \boldsymbol{\theta}_{A, i}
\end{aligned}
$$

or

$$
\begin{align*}
\delta_{i}^{\prime} & =\left(e_{i}^{n_{s}}\right)^{T}\left(\Phi^{T} \Phi\right)^{-1} \Phi^{T}\left(\theta^{\prime}-\hat{\boldsymbol{\theta}}\right) \\
& =\left(\Phi\left(\Phi^{T} \Phi\right)^{-1} e_{i}^{n_{s}} \mid \boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \tag{5.44}
\end{align*}
$$

This should be reflected in the calculation of the bounds $\bar{d}_{i}^{w}$ and $\underline{d}_{i}^{w}, i \in$ $\boldsymbol{n}_{s}, w \in \mathcal{W}$. In section 4.11 it was derived for the upper bounds $\bar{d}_{i}^{w}$ for the case of orthonormal $\boldsymbol{\theta}_{A, i}$ that

$$
\begin{equation*}
\forall i \in \boldsymbol{n}_{\boldsymbol{s}}, w \in \mathcal{W} \quad \bar{d}_{i}^{w}:=-\left(\hat{\boldsymbol{\theta}} \mid \boldsymbol{\theta}_{A, i}\right)+\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}^{\prime} \mid \boldsymbol{\theta}_{A, i}\right) \tag{4.42}
\end{equation*}
$$

Based on (5.44), this should now become

$$
\begin{equation*}
\forall i \in n_{\boldsymbol{s}}, w \in \mathcal{W} \quad \bar{d}_{i}^{w}:=\max _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}^{w}}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}} \mid \Phi\left(\Phi^{T} \Phi\right)^{-1} e_{i}\right) \tag{5.45}
\end{equation*}
$$

and for the lower bound we find

$$
\begin{equation*}
\forall i \in n_{s}, w \in \mathcal{W} \quad \underline{d}_{i}^{w}:=\min _{\boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\theta}^{w}}\left(\boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}} \mid \Phi\left(\Phi^{T} \Phi\right)^{-1} e_{i}\right) \tag{5.46}
\end{equation*}
$$

The only change that is required for the calculation of the unstructured error bound is that (5.41) and (5.42) should be used instead of (4.59) and (4.58) for the definition of the projection operators $P_{A}$ and $P_{A}^{\perp}$. The combination of local errors to global errors can proceed as before.

Remark 5.6 In this section it was shown how to use arbitrary structured error components. In section 5.6.1 the direction of the structured error components could be influenced by weighting the model parameters before doing principal component analysis. One may wonder to what extent the use of arbitrary structured error components is more general than weighting.

The example below will demonstrate that through weighting only a limited set of structured error components can be obtained.

Consider the case of two arbitrary vectors $\boldsymbol{\theta}_{1}$ and $\boldsymbol{\theta}_{2}$, both in $\mathbb{R}^{c}$. Let

$$
W=\left[\begin{array}{lll}
w_{1} & & \\
& \ddots & \\
& & w_{c}
\end{array}\right], \quad w_{i}>0, i \in c
$$

be a weighting matrix such that $W \boldsymbol{\theta}_{1}$ and $W \boldsymbol{\theta}_{2}$ would be two of the left singular vectors of $W X$ in (5.31). It would hold

$$
\begin{aligned}
\left(W \theta_{1} \mid W \theta_{2}\right) & =\sum_{i=1}^{c} w_{i}^{2} \theta_{1, i} \theta_{2, i} \\
& \left.=\left(\begin{array}{lll}
w_{1}^{2} & \ldots & w_{c}^{2}
\end{array}\right]^{T} \left\lvert\,\left[\begin{array}{llll}
\theta_{1,1} \theta_{2,1} & \ldots & \theta_{1, c} \theta_{2, c}
\end{array}\right]^{T}\right.\right)=0
\end{aligned}
$$

The first vector in the second inner product has only positive entries. The inner product can now only be zero if the entries of the second vector are either all zero or do not have the same sign. This may not be true. Consequently, structured error components $\boldsymbol{\theta}_{1}$ and $\boldsymbol{\theta}_{2}$ can not be obtained by weighting. Being able to specify structured errors explicitly is truly more general than weighting.

### 5.8 Unstable systems and closed loop experiments

The algorithm is only suitable for stable systems. This is due to the fact that only for stable systems $G(\zeta)$ the expansion

$$
G(\zeta)=\sum_{k=0}^{\infty} \theta_{k} B_{k}(\zeta)
$$

is possible with an exponentially decaying sequence of $\theta_{k}$ 's. This exponential decay was in turn used heavily to bound the effect of undermodelling.

The algorithm itself is not restricted to open loop situations: as long as the prior information can be obtained, the algorithm can be applied. This does not mean, that experimental data obtained in closed loop do not complicate things at all. The complications occur at the level of the identification steps that


Figure 5.12: Feedback configuration
are part of the algorithm, for example for the estimation of auxiliary models. These identification steps have to be able to deal with closed loop data. The least squares algorithm that was proposed by way of example is not suitable for closed loop data.

A technique going back to (Hansen and Franklin, 1988; Hansen et al., 1989; Hansen, 1989) can be used to circumvent both problems. The technique was elaborated further in (Schrama, 1992; Schrama and Van den Hof, 1992) and was applied by (De Vries, 1994) and later by (Hakvoort, 1994) for the purpose of model uncertainty bounding. See De Vries (1994) for more references on the subject and for further motivation of the framework.

The idea is not to identify the possibly unstable plant itself, but (stable) coprime factors of this plant. This will be explained in more detail shortly. Then the implications this scheme has for the application of the algorithm will be considered.

### 5.8.1 Coprime factor representation

First, the following assumptions are made:

1. The linear manifestations $G_{t r}^{w}$ of the plant in the operating point $w \in \mathcal{W}$ are linear time-invariant finite dimensional (LTIFD) systems.
2. For all operating points $w \in \mathcal{W}, G_{i r}^{w}$ is stabilised by the same known LITIFD controller $C$ in the configuration of figure 5.12.
3. A (LTIFD) nominal model $\hat{G}$ for $G_{t r}$ is available that is stabilised by $C$.

Assumption 1 is new only as far as the finite dimensionality of $G_{t r}^{w}$ is concerned. The linear manifestations of the true system were almost by definition considered linear time-invariant processes.

In the feedback configuration of figure $5.12 e(t)$ is a signal representing noise, disturbances and possibly also other effects related to undermodelling,


Figure 5.13: Extended dual Youla parametrisation
see chapter 3. $r_{1}(t)$ and $r_{2}(t)$ are reference signals. They need not both be present for the purposes of this section. Note that the effect of $r_{2}(t)$ can also be achieved by increasing $r_{1}(t)$ with a term $C r_{2}(t)$.

For completeness the definition of right-coprime factorisations is given:

Definition 5.2 A pair $(N(\zeta), D(\zeta))$ is called a right-coprime factorisation over $\mathcal{R} \mathcal{H}_{\infty}$ of a linear, time-invariant, finite dimensional $q \times p$ system $G$ if

1. $G(\zeta)=N(\zeta) D(\zeta)^{-1}$
2. $N(\zeta) \in \mathcal{R} \mathcal{H}_{\infty}^{q \times p}$ and $D(\zeta) \in \mathcal{R} \mathcal{H}_{\infty}^{p \times p}$
3. $\exists U(\zeta) \in \mathcal{R} \mathcal{H}_{\infty}^{p \times q}, V(\zeta) \in \mathcal{R} \mathcal{H}_{\infty}^{p \times p}, \quad U(\zeta) N(\zeta)+V(\zeta) D(\zeta)=I$

Loosely speaking, the coprime factors $N$ and $D$ can be seen as a stable numerator and denominator of $P$ satisfying that there is no cancellation of unstable zeros in the quotient $N D^{-1}$.

Given the assumptions made above, $G_{t r}^{w}, w \in \mathcal{W}, C$ and $\hat{G}$ admit a right-coprime factorisation over $\mathcal{R} \mathcal{H}_{\infty}$, see (Vidyasagar, 1985). Let ( $N_{t r}^{w}, D_{t r}^{w}$ ), $\left(N_{c}, D_{c}\right)$ and $(\hat{N}, \hat{D})$ be right coprime factorisations of $G_{t r}^{w}, w \in \mathcal{W}, C$ and $\hat{G}$ respectively. The assumption that all $G_{t r}^{w}$ are stabilised by a controller $C$ that is known to stabilise $\hat{G}$ is equivalent to the assumption that there exists for all $w \in \mathcal{W}$ a $R_{t r}^{w} \in \mathcal{R} \mathcal{H}_{\infty}$ such that the system drawn in figure 5.13 is equivalent to the part drawn in a dashed box in figure 5.12. In that case it also holds that $S_{t r}^{w} \in \mathcal{R} \mathcal{H}_{\infty}$ for all $w \in \mathcal{W}$. This result is shown in (Schrama, 1991; Schrama, 1992). The interconnection drawn in figure 5.13 is known as the extended dual Youla parametrisation.

In (Schrama, 1991; Schrama, 1992) it is further shown that for $u$ and $y$ obtained in an operating point $w, w \in \mathcal{W}$, it holds

$$
\begin{align*}
& x=(\hat{D}+C \hat{N})^{-1}\left(r_{1}+C r_{2}\right)=(\hat{D}+C \hat{N})^{-1}(u+C y) \\
& z=R_{t r}^{w} x+S_{t r}^{w} e=D_{c}^{-1}(y-N x)  \tag{5.47}\\
& u=D_{t r}^{w} x-N_{c} S_{t r}^{w} e \\
& y=N_{t r}^{w} x+D_{c} S_{t r}^{w} e
\end{align*}
$$

Provided the external references $r_{1}$ and $r_{2}$ are not correlated with the disturbance $e, x$ is uncorrelated with $e$. Note that $x$ and $z$ are computable because $r_{1}, r_{2}, \hat{N}, \hat{D}$ and $C$ are known. This means that the transfer from $x$ to $z$ can be estimated in an open loop fashion. Moreover

$$
\begin{align*}
D_{t r}^{w} & =\hat{D}-N_{c} R_{t r}^{w} \\
N_{t r}^{w} & =\hat{N}+D_{c} R_{t r}^{w} \tag{5.48}
\end{align*}
$$

so that $D_{t r}^{w} \in \mathcal{R} \mathcal{H}_{\infty}^{p \times p}$ and $N_{t r}^{w} \in \mathcal{R} \mathcal{H}_{\infty}^{q \times p}$. These transfers being stable, the algorithm can be used to bound the uncertainty in $D_{t r}^{w}$ and $N_{t r}^{w}$. However, bounding the uncertainty in $D_{t r}^{w}$ and $N_{t r}^{w}$ separately ignores the fact that these transfers are related through (5.48), where $\hat{N}, \hat{D}, N_{c}$ and $D_{c}$ are known transfers. Ignoring the relation that exists between the uncertainty in $D_{t r}^{w}$ and $N_{t r}^{w}$ leads to more conservative error bounds, so it is better to bound the uncertainty in $R_{t r}^{w}$.

Note that the coprime factors $D_{t r}^{w}$ and $N_{t r}^{w}$ and the Youla parameter $R_{t r}^{w}$ that will be found depend on the factors of the nominal model. For a different nominal model or even for a different factorisation of the same model, different factors $D_{t r}^{w}$ and $N_{t r}^{w}$ will be found. (If the identification of $D_{t r}^{w}$ and $N_{t r}^{w}$ is unbiased, the quotient $N_{t r}^{w}\left(D_{t r}^{w}\right)^{-1}$ does not depend on the factors of the nominal model of course.) This may even go so far, that the order of $N_{t r}^{w}$ and $D_{t r}^{w}$ is different for different (factors of) the nominal model. This interesting issue is not investigated further here. The interested reader is referred to (Van den Hof and Schrama, 1995).

### 5.8.2 Implications for the algorithm

The most important implication for the algorithm is, that the model uncertainty is no longer additive uncertainty but "Youla parameter uncertainty." The uncertainty description is

$$
\begin{align*}
& N_{t r}=\hat{N}+D_{c} R_{t r} \\
& D_{t r}=\hat{D}-N_{c} R_{t r}  \tag{5.49}\\
& R_{t r}=R_{0}+\Delta_{R}+\sum_{i=1}^{n_{s}} \delta_{i} A_{R ; i}
\end{align*}
$$

where

$$
\begin{gathered}
N_{t r}, \hat{N}, N_{c} \in \mathcal{H}_{2}^{q \times p}, \quad D_{t r}, \hat{D}, D_{c} \in \mathcal{H}_{2}^{p \times p} \\
R_{t r}, R_{0}, \Delta_{R}, A_{R, i} \in \mathcal{H}_{2}^{q \times p}, \quad \underline{d}_{i} \leq \delta_{i} \leq \bar{d}_{i} \\
\left\|\Delta_{R}\right\| \leq d_{R, \Delta}, \quad d_{R, \Delta}, \underline{d}_{i}, \delta_{i}, \bar{d}_{i} \in \mathbb{R}, \\
i \in n_{s}
\end{gathered}
$$

This structure is visualised in figure 5.14. The uncertainty consisting of $\Delta_{R}$ and $\delta_{1}, \ldots, \delta_{n_{s}}$ can be used by $\mu$ synthesis as well.


Figure 5.14: Coprime factor uncertainty structure
The prior knowledge that is required for successful application of the algorithm is the same as that for the original algorithm, with the modifications that $x$ and $z$ take the role of the original $u$ and $y$. Also $R_{t r}^{w}$ acts like $G_{t r}^{w}$ did in the original algorithm. Nevertheless some further discussion is desirable, as these substitutions may have some consequences for the ways in which this prior knowledge can be obtained.

- The same data sets are required as for the "standard" algorithm. Further the controller that was present in the loop during the experiments has to be known. According to (5.47) this is sufficient to reconstruct the signals $x$ and $z$.
- For some data sets $d \in \mathcal{D}$ an instrumental variable $v^{d}(t), t \in \mathbb{T}^{d}$ may be available. This instrumental variable should be correlated with $x$ but not with the disturbances $e$. This is equivalent to requiring that the instrumental variable is correlated with $r_{1}$ and/or $r_{2}$ but not with $e$.
Similarly to the open loop case, $x$ itself can be used as an instrumental variable if no other qualifying signal is available or desired. As the identif-
cation of the relation between $x$ and $z$ is an open loop problem, there is no strict need for instrumental variables other than $x$.
- For each operating point $w \in \mathcal{W}$, the data is assumed to be affected by noise that is additive to $z$. According to (5.47) and figure 5.13, a sufficient condition for this is, that the noise is additive to the output, as in the open loop case. Being in a closed loop situation, the input $u$ is now affected as well by the noise.

The Youla parameter of the linear manifestations of the true system is expressed in the MIMO basis in the following way:

$$
R_{t r}^{w}(\zeta)=\sum_{\sigma \in S} \sum_{k=0}^{\infty}\left(\theta_{R, t r}^{w}\right)_{k}^{\sigma} E^{\sigma} B_{k}^{\sigma}(\zeta), \quad \sigma \in \mathbb{S}
$$

- For each operating point, and for each subtransfer $\sigma \in \mathcal{S}$ of the Youla parameter $R_{t r}^{w}$, values $M_{R}^{\sigma} \in \mathbb{R}_{+}, \rho_{R}^{\sigma} \in[0,1), \theta_{R, \text { max }}^{\sigma} \in \mathbb{R}_{+}$and $k_{R}^{*} \in \mathbb{N}$ are known, such that

$$
\left|\left(\theta_{R, t r}\right)_{k}^{\sigma}\right| \leq \bar{\theta}_{R, k}^{\sigma}:= \begin{cases}\theta_{R, \max }^{\sigma} & k<\left(k_{R}^{*}\right)^{\sigma}  \tag{5.50}\\ M_{R}^{\sigma}\left(\rho_{R}^{\sigma}\right)^{k} & k \geq\left(k_{R}^{*}\right)^{\sigma}\end{cases}
$$

In the open loop case, the poles of the basis generating system had to be chosen close to the poles of the true system in order to make $\rho^{\sigma}$ small. Conversely, in case of bounding the uncertainty in the Youla parameter the poles of the basis generating system should be taken close to the poles of the Youla parameter $R_{t r}^{w}$.

If the nominal model is used for the factors $\hat{N}$ and $\hat{D}$, then the Youla parameter will be zero for the nominal model. Perturbations of the nominal model correspond to perturbations of the Youla parameter around 0. Although the fact that $R_{t r}^{w}$ is a stable transfer follows from the fact that the true system is stabilised by the controller $C$, it is difficult to be more precise about the pole locations of $R_{t r}^{w}$ in this case. This makes it difficult to generate a basis with a basis generating system having its poles close to those of $R_{t r}^{w}$.

It may therefore be advisable to base $\hat{N}$ and $\hat{D}$ on a model that is known not to coincide with the true system, so that non-zero values for $R_{t r}^{w}$ can be expected. This model has to be stabilised by $C$, but using a preliminary Youla parametrisation it is easy to find such models. The procedure suggested in section 4.8 .3 should probably be used subsequently to estimate the bounds $M_{R}^{\sigma}$ and $\rho_{R}^{\sigma}, \sigma \in \mathcal{S}$.

- For each operating point $w \in \mathcal{W}$ and for each data set $d \in \mathcal{D}(w)$, a signal $\bar{e}_{z}: \mathbb{T}^{d} \rightarrow \mathbb{R}_{+}^{q}$ is available, such that

$$
\forall t \in \mathbb{T}^{d} \quad\left|z^{d}(t)-R_{t r}^{w}(\zeta) x^{d}(t)\right| \leq \bar{e}_{z}^{d}(t)
$$

Basically the signal $\bar{e}_{z}^{d}(t)$ can be derived from the signal $\bar{e}(t)$, the $\ell_{1}$-norm of $S_{t r}^{w}$ and suitable assumptions concerning the extension of $\bar{e}(t)$ for $t<0$. Unfortunately the transfer $S_{t r}^{w}$ is not known exactly. It holds, see for example (Van den Hof and Schrama, 1995),

$$
S_{t r}^{w}=D_{c}^{-1}\left(I+G_{t r}^{w} C\right)^{-1} H
$$

This involves the unknown transfer $G_{t r}^{w}$. Also it was demonstrated in section 5.2 that such a bound based on the $\ell_{1}$ norm may be very conservative. It seems to be a better idea to base the bound $\tilde{e}_{z}$ on the residuals in the identification of $R_{t r}^{w}$ as was proposed in section 4.8.3.

- For each data set $d \in \mathcal{D}$, a vector $\bar{x}^{d} \in \mathbb{R}_{+}^{p}$ is available, such that

$$
\forall t<0 \quad\left|x^{d}(t)\right| \leq \bar{x}^{d}
$$

Let $\bar{r}^{d}$ be a bound for $\left|r_{1}(t)+C r_{2}(t)\right|, t<0$. Then it follows from (5.47) that

$$
\bar{x}^{d} \leq\left\|(\hat{D}+C \hat{N})^{-1}\right\|_{\ell_{1}} \bar{r}
$$

Once again, this is probably a conservative estimate for $\bar{x}^{d}$, see section 5.2 . The procedure suggested in section 4.8 .3 should be preferred.

This concludes our discussion of the main types of prior knowledge that may be or should be available to the algorithm in the closed loop case. The crosscovariance bound (4.9) and the extensions discussed in section 5.4 and 5.5 can also be incorporated into this closed loop procedure with little modifications, similar to the ones discussed above.

Once the prior knowledge is available, the algorithm can be followed unaltered to obtain finally the fully specified uncertainty description given in (5.49)

## 6

## Examples and case studies

6.1 Overview<br>6.2 A simple SISO simulation example<br>6.3 A more complex SISO simulation example

6.4 A MIMO simulation example<br>6.5 Case study:<br>Asynchronous motor<br>6.6 Conclusions

### 6.1 Overview

In this chapter the algorithm of chapter 4 and 5 will be tested by means of a number of examples.

The first example is a SISO simulation example that was constructed in such a way that it fits the assumptions and uncertainty description of the algorithm. This example was already presented throughout chapter 4. The results presented there are briefly repeated and some new results are given. The second example is also a SISO simulation example, but the structure that is present in the model uncertainty can not be represented exactly by the uncertainty description of the algorithm. This example gives an indication of the capabilities of the algorithm to capture in an approximating way the structure that is present in the uncertainty of the transfer of a process if that structure does not fit the uncertainty description (4.4). The third example is a MIMO simulation study. Some of the effects that may occur in MIMO situations, including the effect of weighting as discussed in section 5.6 , will be shown.

The final example uses practical data taken from an asynchronous motor. It is known from physical reasoning and practical experience that the characteristics of this motor change if the magnetic material inside the motor saturates. It will be investigated whether this influence on the dynamics of the motor can be described by structured error components and whether these components can be found by the algorithm.

### 6.2 A simple SISO simulation example

### 6.2.1 System and data

A system with two "operating points" is considered. In the first operating point, denoted operating point $a$, the transfer of the process is

$$
G_{t r}^{a}(z)=\frac{1}{z-0.97}
$$

and for the second operating point, denoted $b$, the transfer is

$$
G_{t r}^{b}(z)=1.5 G_{t r}^{a}(z)=\frac{1.5}{z-0.97}
$$

A basis was generated using the system

$$
G_{b}(z)=\frac{1}{z-0.95}
$$

The poles of the basis generating system were deliberately chosen close to but not equal to the poles of the linear manifestations of the true process. Only one data set per operating point was used. For both operating points a data set was generated containing 1500 samples, with zero initial conditions. The first 500 samples were discarded, the remaining 1000 samples were taken as the experimental data sets. The experimental data sets therefore had unknown, non-zero initial conditions. 10 basis functions were incorporated in the model. This gave sufficient freedom in the model set to contain an accurate approximation of the true process.

The inputs were random white noise sequences having a standard normal distribution. Additive noise was added to the output. The signal to noise ratio at the output was 40 dB in the first operating point. In the second operating point the absolute noise level was taken the same, so that the signal to noise ratio was better in this operating point. The noise was uncorrelated with the inputs, was not coloured and had a zero-mean Gaussian distribution. The fact that the noise was not coloured was not used by the algorithm itself, nor in the identification of auxiliary models.

No prior information was given to the algorithm concerning the decay rate of the model parameters. As the poles of both the true system and the basis generating system are known in this example, it can be calculated that there exist $M_{a}, M_{b} \in \mathbb{R}_{+}$, such that

$$
\begin{aligned}
& \left|\theta_{t r, k}^{a}\right| \leq M_{a} 0.255^{k} \\
& \left|\theta_{t r, k}^{b}\right| \leq M_{b} 0.255^{k}
\end{aligned}
$$

where $\theta_{t r, k}^{a}$ and $\theta_{t r, k}^{b}$ are the true expansion coefficients of $G_{t r}^{a}$ and $G_{t r}^{b}$ expressed in the basis generated by $G_{b}$. No noise levels or bounds on past inputs were
given either. The inputs were used as instrumental variables in the crosscovariance bound (4.9). The value of $\bar{c}_{\ell}$, which is the actual cross-covariance bound, was estimated by the algorithm.

### 6.2.2 Results

First the results obtained using one principal component are discussed. The algorithm was run once without estimating initial conditions, using the extended noise and cross-covariance bounds (4.30) and (4.37) and once while estimating initial conditions, using the extended noise and cross-covariance bounds (5.5) and (5.7).

The extended noise bound limits the influence of noise and transients on the output. In figure 6.1 the bounds are shown using dashed lines and the sum of the true transient and the true noise are plotted with a continuous line. The left part of the picture shows the results obtained with the bound (5.5),


Figure 6.1: Extended noise bound, with (left) and without (right) estimating initial conditions
i.e. while estimating initial conditions. The right part shows the results using the bound (4.30), without estimating initial conditions. The areas between the dashed lines are the areas that are consistent with the respective extended noise bounds. It is clear that estimating the transient improves the extended noise bound considerably.

The truncation value $\tilde{k}$ was taken equal to 19 in the calculation of the extended bounds. In this way the same number of basis functions is used for the model and for the first part of the tail of the basis. The undermodelling due to the first part of the tail is bounded in a less conservative way than that due to the rest of the tail. It seems reasonable that if ten basis functions are enough to get sufficiently accurate models, then ten extra basis functions should be enough to get a sufficiently non-conservative bound on the influence of the tail.

Given the above value of $\tilde{k}, \hat{a}(t)$ in (4.26) accounts for the possible influence of known inputs through basis function 10 to 19 and $\hat{b}(t)$ in (4.27) bounds the possible influence of unknown inputs through basis function 0 to $19 . \delta$ in (4.28) accounts for the influence of all inputs, known and unknown, through basis function 20 and higher.

In the left hand picture, the mean and standard deviation of $\hat{a}(t)$ were 0.0011 and 0.0007 , respectively. The value of the noise bound $\bar{e}(t)$ was estimated to be 0.14 for all $t$ and the value of $\delta$ was $0.59 \cdot 10^{-5}$. It is clear that the contribution of $\hat{a}(t)$ and $\delta$ is negligible. This is mainly due to the fast decay of the estimated model parameters. This will be discussed in more detail shortly.

In the right hand picture, two effects occur. Most visible is the large value of $\hat{b}(t)$. It is the contribution of $\hat{b}(t)$ that causes the excessive initial values for the extended noise bound. The other effect is that the residuals of the estimation become larger, because they also have to account for an unmodelled transient in the data. This manifests itself in the larger estimated noise bound $\bar{e}(t)$ : the value 3.05 was estimated for all $t$.

The lack of modelling of the transient results in less accurately estimated model parameters. This leads to worse values for $M$ and $\rho$ and in turn to larger values for $\hat{a}(t)$ and $\delta$. The mean value and standard deviation of $\hat{a}(t)$ are in this case 0.0049 and 0.0028 . The value of $\delta$ is $0.12 \cdot 10^{-3}$. These values are larger than in the case that the transient is estimated, but they are still negligible.

The absolute values of the estimated model parameters and the parameter bounds $\bar{\theta}_{k}, k=0, \ldots, 9$ that were estimated from these are shown in figure 6.2. If the transient is not estimated, the parameters of the high order basis functions become less exact, which means in this case that they become larger. Therefore the estimated bounds $\bar{\theta}_{k}$ become larger as well.


| transient estimated |  |
| :---: | :--- |
| $-\mathrm{x}:$ | Model param. <br> $-:$$\overline{\bar{\theta}}_{k}$ |
| transient not estimated |  |
| $--\mathrm{o}:$ | Model param. |
| $--:$ | $\bar{\theta}_{k}$ |

Figure 6.2: Estimated model parameters and their bounds

Not estimating the transient that is present in the data inflates the polytopes $\mathcal{L}_{\theta}^{w}, w \in \mathcal{W}$ containing the true model parameters: the extended noise bound, the extended cross-covariance bound and the bounds $\bar{\theta}_{k}$ all increase (except for $\bar{\theta}_{0}$, see figure 6.2 ). The slower decay of the bounds $\bar{\theta}_{k}$ also causes a larger influence of the tail of the basis on the uncertainty bounds. In figure 6.3 the impact this has on the uncertainty regions due to the unstructured error is shown for a number of frequencies. The uncertainty regions apply globally, i.e. for both operating points. There is clearly a dramatic increase in the size of the uncertainty regions if no transients are estimated.


Figure 6.3: Frequency uncertainty regions with one structured error component; light areas: uncertainty without estimated transient, dark areas: uncertainty with estimated transient, $\mathbf{x}$ : models estimated for an operating point

With this the comparison of results with and without estimating initial conditions is ended. Based on these results only the algorithm that estimates the transients along with the model will be used further.

Now the results with a structured error component are compared to those without one. Figure 6.4 shows the uncertainty for a number of frequencies. The x's in this plot show the transfer of the estimated auxiliary models for the two operating points. The line roughly halfway between the two lines connecting the x's corresponds to the transfer of the nominal model. The light areas represent the uncertainty that is found without a structured error component. The dark areas show the uncertainty when one structured error component is removed


Figure 6.4: Frequency uncertainty regions with estimated transient; light areas: no structured error, dark areas: one structured error component
from the unstructured error. The size of this structured error component as bounded by $\underline{d}_{1}$ and $\bar{d}_{1}$ is shown by dashed lines. The dark areas in figure 6.3 and figure 6.4 are the same.

Figure 6.5 shows the process transfers and the uncertainty bounds in a Bode amplitude plot. The light shaded area represents the nominal model together with the uncertainty due to the unstructured error if no structured error components are used. The two continuous lines near the edges of this area represent the true transfers in the two operating points, the one on top in operating point $b$, the other in operating point $a$. The unstructured error, which is the only model uncertainty if no structured error is used, appears to be large enough to contain the difference with the nominal model for both true transfers.

The darker shaded area represents the nominal model and unstructured error if one structured error component is used. Adding a structured error component has clearly shrunk this error. The dashed line is the amplitude plot for

$$
\max \left\{\bar{d}_{1},-\underline{d}_{1}\right\} A_{1}\left(e^{j \omega}\right)
$$

The dotted line right below it is the amplitude of the true difference between the process and the nominal model. Actually there are two dotted lines, one for


Figure 6.5: Uncertainty in Bode amplitude plot, global results
each operating point, but they are so close that they can not be distinguished in this plot. As far as one can tell from the Bode amplitude plot, the structured error component is consistent with the nominal model and the two true process transfers.

Comparing the Nyquist plot in figure 6.4 with the Bode plot in figure 6.5 , the Nyquist plot looks probably more dramatic than the Bode plot. This is due to the logarithmic scaling that is employed in the Bode plot.

### 6.2.3 Discussion

It is not surprising that the algorithm performs satisfactory in the situation analysed in this section: the undermodelling is small and so are the noise levels. The uncertainty in the process transfer is clearly dominated by the difference between the transfers of the two operating points.

With two operating points and one structured error component, one can expect the algorithm to find a nominal model equal to the mean of the two transfers. The structured error component is then "aligned" with the difference between the two transfers. This is exactly what happens. In this academic example there is no point in asking whether the structure that was found in the uncertainty is correct. Having only two operating points, the structure is correct if the uncertainty description (4.4) is consistent with the true transfers.

In a practical example much more care should be taken. It is likely that a practical process does have more than two operating points. In fact, the set of
operating points is likely to be an uncountable set in many cases. If one wants to "extrapolate" the validity of the uncertainty bounds to more operating points than those for which data was obtained, it is wise to gather data for more than two operating points. One should also select these operating points carefully to make sure that they cover the relevant changes in process behaviour that one wants to capture by the uncertainty description.

Nevertheless it can be concluded that the algorithm performs as could be expected in a situation that fits the algorithm. In the next section a situation that fits the algorithm less well is analysed.

The other conclusion to be drawn from this section is that modelling the transient can have a large beneficial effect on the size of the extended noise bound. How large this effect is depends on the noise level and the level of undermodelling in an operating point.

### 6.3 A more complex SISO simulation example

### 6.3.1 System and data

In this section a system will be considered that can be described by the following set of transfers

$$
G_{t r}^{w}(z)=\left(0.003-0.01 \alpha^{w}\right) \frac{\left(1-\alpha^{w}\right) z}{\left(z-0.97-0.1 \alpha^{w}\right)(z-0.9)}
$$

where $\alpha^{w}$ takes the values listed in table 6.1. Different values of $\alpha^{w}$ correspond to a different operating point. Data was generated for all operating points in table 6.1. The DC-gain of $G_{t r}^{w}(z)$, i.e. $G_{t r}^{w}(1)$, is equal to $1-\alpha^{w}$. The Bode amplitude plots for the system in different operating points are collected in figure 6.6. The changes in process transfer are highly structured, they can be described with a single parameter $\alpha^{w}$, but this structure can not be represented by the uncertainty description (4.4) used by the algorithm.

For each operating point, a data set of 2000 samples was generated, where the input was normally distributed, zero-mean white noise. The first 1000 samples were then discarded, so that reasonably sized transients may be expected in the data sets. Additive noise was added to the output. The signal to

Table 6.1: Operating points and their values of $\alpha$

| $w$ | 1 | 2 | 3 | 4 | 5 |
| :---: | :---: | :---: | :---: | :---: | :---: |
| $\alpha^{w}$ | -0.2 | -0.1 | 0 | 0.1 | 0.2 |



Figure 6.6: Bode amplitude plots for different operating points of the example of section 6.3
noise ratio at all outputs was 40 dB . The output noise was white, zero mean, normally distributed and uncorrelated with the inputs.

10 basis functions were used in the model. The truncation value $\tilde{k}$ in the calculation of the extended noise and cross-covariance bounds was taken equal to 19. A basis was generated with the system

$$
G_{b}(z)=\frac{z}{z-0.975}
$$

The basis generating system was deliberately not tuned too much to the true system to investigate the influence this has. The single pole in the basis generating system is in particular relatively far away from the pole in 0.9 in the true system. Better results would be obtained if a basis was generated by a second order system having an additional pole near $z=0.9$ compared to $G_{b}(z)$ above.

### 6.3.2 First results, discussion

Before presenting the results, figure 6.6 is first considered somewhat closer. Although the plots may seem to be close to each other, they are very different as far as additive errors are concerned. The largest transfer, corresponding to $\alpha^{w}=-0.2$, is for high frequencies an order of magnitude larger than the smallest transfer, which is obtained for $\alpha^{w}=+0.2$. One may also verify, that the time constant corresponding to the pole that moves for differing operating points varies by a factor 5 .

This is reflected in figure 6.7 in which Bode amplitude plots of the nominal model and of the true errors between the nominal model and the process in different operating points are shown. The nominal model was taken equal to the central model in the algorithm. The true errors are, especially for high frequencies, of the same order of magnitude as the nominal model itself.


Figure 6.7: Bode amplitude plot of true additive errors for different operating points ('-') and nominal model ('--')

In figure 6.8 the structured and unstructured errors that were found by the algorithm are shown in a Bode amplitude plot. The dotted lines give the unstructured errors, the dashed lines the structured errors. The dotted line with no additional marker is the unstructured error if no structured error components are used. The dotted line marked with *'s is the unstructured error for one structured error component. On very close inspection it can be seen that there is also a dotted line marked with +'s. This line can be distinguished between $\omega=10^{-2}$ and $\omega=10^{-1}$. It is a little bit below the line marked with *'s. It gives the unstructured error if two structured components are used.

The unstructured error is significantly larger than the true error, especially for large frequencies and for small frequencies. In the remainder of this section, this is analysed further, thereby showing the capability of the algorithm to indicate what effect actually determined the final noise bounds.

For the analysis of the noise bound, attention is restricted to the single frequency $\omega=10^{-3}$. The case for one structured error component is considered, the cases for zero or two structured error components would given comparable


Figure 6.8: Structured and unstructured uncertainty in Bode amplitude plot
results. Figure 6.9 on page 213 shows the transfer function uncertainty regions for this frequency for different operating points. It turns out that the operating point for $\alpha^{w}=-0.1$ has the largest uncertainty region, containing all the others. Consequently the global uncertainty region, i.e. the one that is valid for all operating points, is in this case determined solely by the uncertainty for $\alpha^{w}=-0.1$. (This situation is not only encountered for $\omega=10^{-3}$, but also for other low frequencies. For higher frequencies the size of the uncertainty region for $\alpha^{w}=-0.2$ becomes comparable in size to that for $\alpha^{w}=-0.1$ and both uncertainty regions then determine the global uncertainty region.)

Figure 6.10 on page 213 separates the transfer function uncertainty regions for $\alpha^{w}=-0.1, \omega=10^{-3}$ into a part due to the uncertainty in the model parameters and a part exclusively due to the tail of the basis. The part induced by the model parameter uncertainty is enclosed by a dashed line, the remainder of the area enclosed by a continuous line is caused by the tail of the basis.

As far as the uncertainty due to the tail of the basis is concerned, there are basically two ways to reduce this uncertainty. The first is to use more basis functions in the model. This may increase the size of the uncertainty due to the model parameters, but the reader may verify in section 4.12 that this increase will never be larger than the decrease of the uncertainty due to the tail of the basis.

The second way to reduce the uncertainty caused by the tail of the basis is to use a set of basis functions for which the expansion coefficients for the basis
functions in the tail are smaller. This can be achieved by choosing the poles of the basis generating system closer to the poles of the linear manifestation of the system in this operating point. This may deteriorate the decay rate of the expansion coefficients for the tail of the basis in other operating points. As the poles of the linear manifestation of the system in the operating point that we are considering are not known in a practical setting, matching the poles of the basis generating system better to those of the true system involves either an extra identification of the true system in this operating point or a model reduction step of the corresponding auxiliary model. Recall that the poles of the auxiliary models coincide with the poles of the basis generating system (up to multiplicity).

In this particular example, the decay rate of the tail could be improved by generating a basis with a second order system. It was already mentioned that the pole in $z=0.9$ of the true system is not accounted for in the basis generating system. Leaving this pole out of the basis generating system has apparently a significant influence on the size of the tail of the basis.

Remark 6.1 The previous discussion may suggest that the area enclosed by dashed lines in figure 6.10 is independent of the tail of the basis. This is not true. This area is determined, among other things, by the extended noise and cross-covariance bounds. In the extension of these bounds, the tail of the basis is involved as well.

Attention is now focussed on the uncertainty due to the model parameter uncertainty. For $\omega=10^{-3}$ this corresponds to the area within dashed lines in figure 6.10. The size of this part of the uncertainty is determined by the set $\mathcal{L}_{\boldsymbol{\theta}}^{w}$ of linear constraints. $\mathcal{L}_{\theta}^{w}$ consists in this case of ten upper and ten lower bounds for each of the model parameters, 1000 upper and 1000 lower bounds corresponding to the extended noise bound and one upper and one lower bound originating from the extended cross-covariance bound. It turns out, that the only constraints that were ever active during the linear programming problems that were solved in this operating point are constraints that come from the extended noise bound. The cross-covariance bound was never active; if it were omitted from the problem, the same uncertainty regions and structured error bounds would have been obtained! (One can not tell in advance whether this situation will occur: in some examples it did and in others it did not.)

To keep the memory requirements of the algorithm reasonably small, it was only recorded which constraints were active at least once in all linear programs in an operating point. This means that the constraints that were reported to be active may not have been active during the bounding of the unstructured error for $\omega=10^{-3}$. They may have been active instead while bounding the unstructured error for other frequencies or while bounding the structured error component.

Figure 6.11 on page 214 shows the residuals together with the extended noise upper and lower bounds. The noise bounds that were active are marked with a ' $o$ '. There seems to be a tendency to avoid making the noise bound active for samples that correspond to a local peak in the noise bound. This is to be expected, as these constraints are less strict than those for samples for which the noise bound does not peak. Surprisingly enough, for some of the samples for which the noise bound peaks, the corresponding constraints have been active as well.

To continue the search of the effects that eventually determine the unstructured error in figure 6.8, the extended noise bound is separated in figure 6.12 on page 214 into its constituents. The straight line at a constant level of 0.0069 is the unextended noise bound. The line that is for small $t$ a bit above the unextended noise bound is the sum of the unextended noise bound and the bound on the transient as derived in section 5.2. The "noisy" line above that is the sum of unextended noise bound, transient bound and $\hat{a}(t)$. In this example, $\hat{a}(t)$ is a bound on the response of basis functions 10 to 19 to the inputs in the experimental data set. In the last line in the top of the plot the term $\delta$ is finally added. $\delta$ is a bound on the response of basis functions with index 20 and higher, both to the inputs in the experimental data set and to the inputs preceding the experiments.

Clearly the contribution of $\delta$, i.e. the response of basis functions 20 and higher, is the largest one in this example. From equation (4.28) on page 108 can be seen that, given a set of basis functions, the value of $\delta$ is determined by $\max \{\bar{u},|u(0)|, \ldots,|u(T-1)|\}$ and $\bar{\theta}_{k}, k=20, \ldots$. In this example it holds

$$
\begin{gathered}
\max \{\bar{u},|u(0)|, \ldots,|u(T-1)|\}=\bar{u}, \\
\bar{\theta}_{k}=M \rho^{k}, \quad k=20, \ldots
\end{gathered}
$$

In table 6.2 the values of $\delta, \bar{u}, M$ and $\rho$ that were found for different operating points are listed. The change in magnitude of $\delta$ over different operating points is in good agreement with the change in magnitude that is observed for the uncertainty regions in frequency domain. It is interesting to compare the values found for $\alpha^{w}=-0.2$ with those for $\alpha^{w}=-0.1$. Both $\bar{u}$ and $M$ are larger for $\alpha^{w}=-0.2$ than for $\alpha^{w}=-0.1$, which leads in principle to a larger value of $\delta$. However, this is more than compensated by the decrease in $\rho$ from 0.681 to 0.666 : the value of $\delta$ appears to be very sensitive to the value of $\rho$. This also explains the small values of $\delta$ for $\alpha^{w}=0.1$ and $\alpha^{w}=0.2$. These can be attributed to the small value of $\rho$ that is estimated in these operating points.

### 6.3.3 Further results

The example in this section was designed to get insight into the capabilities of the algorithm to describe structure in a process uncertainty that does not fit the uncertainty description (4.4). Figure 6.8 on page 209 seems to indicate

| $\alpha^{w}$ | $\delta$ | $\bar{u}$ | $M$ | $\rho$ | $\delta / \bar{u}$ |
| ---: | ---: | ---: | ---: | ---: | ---: | ---: |
|  | $\cdot 10^{3}$ |  |  |  | $\cdot 10^{3}$ |
| -0.2 | 20.1 | 3.79 | 0.324 | 0.666 | 5.3 |
| -0.1 | 26.1 | 3.62 | 0.266 | 0.681 | 7.2 |
| 0.0 | 14.0 | 3.82 | 0.220 | 0.666 | 3.7 |
| 0.1 | 0.8 | 3.88 | 0.184 | 0.587 | 0.2 |
| 0.2 | 0.1 | 3.27 | 0.176 | 0.542 | 0.0 |

Table 6.2: Values of/determining $\delta$ for different operating points
that these capabilities are very restricted: using one or two structured error components hardly changes the unstructured error. Moreover, the unstructured error bound is much larger than the true model error.

The analysis in the previous part of this section suggests that the unstructured error is dominated by contributions which can be traced back to the influence of undermodelling and the way in which this is bounded. The tail of the basis inflates the extended noise bound such, that the uncertainty in the process transfer is mainly determined by this effect. This, in turn, was caused by leaving out a pole of the true system from the basis generating system.

To verify whether this is indeed the case, the algorithm was run again, but this time the tail of the basis was ignored. This was achieved by setting

$$
\bar{\theta}_{k}=0, \quad k>9
$$

For $k=0, \ldots, 9$ the original values for $\bar{\theta}_{k}$ were maintained. This gives the results shown in figure 6.13 on page 215 . The meaning of the plots in this figure is the same as in figure 6.8. The dotted line with no markers is the unstructured error that is obtained without structured error components. The dotted line marked with *'s is the unstructured error that remains after one structured error component is removed and the dotted line marked with +'s the unstructured error with two structured error components removed. The dashed line with $x$ 's is the first structured error component, that with o's is the second.

A number of things can be observed from this figure. First of all, the unstructured error that is obtained if no structured errors are used is in much closer agreement with the true size of the model errors in different operating points than was the case in figure 6.8. Secondly, the first structured error component is very close to the unstructured error. It leads indeed to a significant reduction of the unstructured error if this component is removed from the unstructured error. Removing a second error component has a much smaller influence on the size of the unstructured error. This is in correspondence with the size of the singular values that are found in the singular value decompo-


Figure 6.9: Frequency uncertainty regions in different operating points for $\omega=10^{-3}$


Figure 6.10: Total uncertainty region (-) and uncertainty region not due to the tail of basis (--) for $\omega=10^{-3}$


Figure 6.11: Residuals and extended noise bounds. Active bounds are marked with a ' $o$ '


Figure 6.12: Constituents of the extended noise bound (see text)


Figure 6.13: Structured and unstructured uncertainty in Bode amplitude plot, effects of undermodelling ignored (see text)


Figure 6.14: Residuals and extended noise bounds. Active bounds are marked with a ' 0 '. Effects of undermodelling are ignored.
sition involved in the principal component analysis, see table 6.3. The first principal component has an associated singular value that is about six times as large as that of the second principal component.

Table 6.3: Singular values of the four structured error components

| $\sigma_{1}$ | $\sigma_{2}$ | $\sigma_{3}$ | $\sigma_{4}$ |
| :---: | :---: | :---: | :---: |
| 0.097 | 0.015 | 0.006 | 0.003 |

For comparison the extended noise bound has been drawn in figure 6.14 for the same operating point as in figure 6.11 on page 214. The active constraints are again marked with a ' $o$ '. This time the extended noise bound consists only of the unextended noise bound and a bound on the transient in the data. All other contributions to the extended noise bound appearing in figure 6.12 on page 214 are zero in figure 6.14 because the tail of the basis is forced to zero through the values of $\vec{\theta}_{k}$. The extended cross-covariance bound has been active as well, both at its upper bound and its lower bound. As in figure 6.11, there seems to be no pattern in the active constraints, such that (most) active constraints could have been selected beforehand from the total set of linear constraints. This issue will be further considered in section 6.5.3.

By plotting the local uncertainty regions for all operating points (not done here), it can be seen that the unstructured error is no longer determined by a single operating point. The uncertainty regions now overlap mostly. In general an uncertainty region for one operating point is no longer fully contained in those for other operating points.

The first structured error component is drawn in figure 6.15 for a number of frequencies. The normal continuous lines are the Nyquist plots of the system in different operating points. The dotted lines connect points on different Nyquist plots for the same frequency. The thick continuous lines represent the uncertainty due to the first structured error component for these frequencies. The structured error seems to give indeed a good account of the changes that occur if the system moves between different operating points.

The uncertainty bounds were further tested by confronting both the structured and unstructured error bounds with the transfers in two "intermediate" operating points, namely for $\alpha^{I}=-0.15$ and $\alpha^{I I}=+0.15$. As in this example the process is completely known in these operating points, the true values of the structured error gains $\delta_{i}, i=1,2$ can be calculated. The results of these calculations together with the lower and upper bounds $\underline{d}_{i}$ and $\bar{d}_{i}$ are summarised in table 6.4. The results do not falsify the structured error bounds $\underline{d}_{i}$ and $\bar{d}_{i}, i=1,2$.

Figure 6.16 on page 218 shows the unstructured errors for both operating points, together with their bounds for the three cases of no structured error


Figure 6.15: First structured error component in Nyquist plot (see text)

Table 6.4: Structured error bounds and true structured error gains for operating points $I$ and $I I$

| $i$ | $\underline{d}_{i}$ | $\bar{d}_{i}$ | $\delta_{i}^{I}$ | $\delta_{i}^{I}$ |
| :---: | :---: | :---: | :---: | :---: |
| 1 | -0.0634 | 0.0639 | -0.0456 | 0.0465 |
| 2 | -0.0113 | 0.0115 | 0.0021 | -0.0005 |

component, one structured error component and two structured error components. Here too, the results do not falsify the error bounds that were found earlier.

An interesting thing happens in the left plot in figure 6.16 for $\omega=0.088$. For this frequency the unstructured error increases if two instead of one structured error components are removed from the unstructured error. Apparently a situation as discussed in figure 4.17 on page 131 occurs for this frequency: by separating the second structured error component from the unstructured error the relation that exists between these two is lost. This relation must have been such, that for $\omega=0.088$ a large unstructured error (with two structured errors removed) can only occur if at the same time the second structured error component has a value that partly compensates the unstructured error. Because this
relation is still present in the unstructured error bound if only one structured error component is removed, the unstructured error with one structured error component removed is for this frequency smaller than that with two structured error components removed.

Bode amplitude plots


Figure 6.16: Validation of unstructured error bounds. left: $\alpha=-0.15$, right: $\alpha=+0.15$; '-': error bound, ':': true error; no marker: no structured error, ' $x$ ': one structured error component, ' $o$ ': two structured error components

### 6.3.4 Conclusions

In this example it turned out, that the size of the extended noise bound is very sensitive to the decay rate of the model parameters. A slightly smaller decay rate leads to a significantly larger contribution of the tail of the basis in the extended noise bound and the extended cross-covariance bound. Therefore it is of major importance for obtaining good results that the basis generating system has its poles close to those of the linear manifestations of the process.

It was possible to pinpoint undermodelling as the cause of rather large unstructured errors because the algorithm can provide very detailed information with respect to the separate components of the error bounds it generates.

Even for structure in the errors that can not be represented by the uncertainty description used by the algorithm, the algorithm is still able to reduce the
unstructured error significantly by using a number of structured error components. In this case using one structured component already gave good results. For the example in this section at least, the algorithm can approximate a large part of the structure in the error by its structured uncertainty description. This is promising, especially because the variation in process transfer was relatively large for different operating points.

### 6.4 A MIMO simulation example

### 6.4.1 System and data

In this simulation example, the set of systems

$$
\begin{aligned}
& G_{t r}(z)=\left[\begin{array}{cc}
\frac{(0.035-\alpha)(1.1+10 \alpha)}{z-0.965-\alpha} & \frac{0.1 \cdot(0.025-\beta)(1+15 \beta)}{z-0.975-\beta} \\
\frac{0.1 \cdot(0.02-\beta)(1+17 \beta)}{z-0.98-\beta} & \frac{(0.03-\alpha)(1+10 \alpha)}{z-0.97-\alpha}
\end{array}\right] \\
& (\alpha, \beta) \in\{(0.01,0.01),(0.01,-0.01),(-0.01,0.01),(-0.01,-0.01)\}
\end{aligned}
$$

is considered. The DC-gain of the system is given by

$$
G_{t r}(1)=\left[\begin{array}{cc}
1.1+10 \alpha & 0.1 \cdot(1+15 \beta) \\
0.1 \cdot(1+17 \beta) & 1+10 \alpha
\end{array}\right]
$$

In every operating point, all sub-transfers are first order transfers with no zero. The off-diagonal terms are smaller in magnitude than the diagonal terms. As in the previous example, the structure that is present in the changes in process transfer over the different operating points can not be described exactly by the uncertainty structure of the algorithm. The relative changes in the offdiagonal transfers are larger than those of the diagonal transfers, both in terms of DC-gain and of time constant.

A basis was generated by the system

$$
G_{b}(z)=\frac{1}{z-0.9725}
$$

$G_{b}(z)$ could have been determined through identification but this involves the risk that an (un)fortunate identification result will turn out to be the explanation of the main features of the results obtained by the algorithm. To rule out this risk, a basis generating system was chosen by hand. The transfer of the basis generating system was chosen close to all subtransfers. The pole location is the mean of the largest and smallest pole that can be found in any of the subtransfers of the system in any of the four operating points. For none of the subtransfers there is an operating point in which the pole location of the basis generating system matches the pole of the subtransfer exactly.

As $G_{b}(z)$ is a SISO transfer, the same set of basis functions was used for every subtransfer. Seven basis functions were used in the models for every subtransfer of this system. This number seems large enough to obtain accurate models while not being so large that it has large repercussions in terms of computing time required by the algorithm. To make the points that are to be raised in this section stand out more clearly, the tail of the basis is consequently ignored in this example!

As in the previous examples, the input sequences in the identification experiments were taken to be white noise sequences having a Gaussian distribution with zero mean and standard deviation one. The length of each data set was 1000 samples. Realistically sized initial conditions were obtained as in the previous two examples. The output signals were disturbed with additive white noise. The output noise was uncorrelated with the inputs and had a Gaussian distribution with zero mean and standard deviation $10^{-2.5}$.

Figure 6.17 shows the Bode amplitude plots for the four subtransfers of the system in different operating points.


Figure 6.17: Bode amplitude plots. no marker: $\alpha=0.01$; ' $x$ ': $\alpha=-0.01$; 'o': $\beta=0.01 ; ~ ‘+’: \beta=-0.01$

### 6.4.2 Results and discussion

Figure 6.18 on page 222 shows the uncertainty regions that were obtained for the different subtransfers using zero and one structured error component. A weighting as discussed in section 5.6 .1 was applied. Recall that this weight-
ing means, that the differences $\Delta \boldsymbol{\theta}^{w}$ between the auxiliary model parameters for different operating points and the central model are pre-multiplied by a weighting matrix $W_{L}$ before performing the principal component analysis:

$$
W_{L} X=W_{L} \operatorname{sbs}_{w \in \mathcal{W}} \Delta \boldsymbol{\theta}^{w}=U_{W} \Sigma_{W} V_{W}^{T}
$$

where $U_{W} \Sigma_{W} V_{W}^{T}$ is a singular value decomposition of the weighted matrix $W_{L} X$. In section 5.6 .1 it was argued, that $W_{L}$ should be a diagonal matrix, having the same weighting for model parameters belonging to the same subtransfer. In this example, $W_{L}$ is therefore a block diagonal matrix, where each block on the diagonal consists of a scalar weighting $w^{\sigma}$ times the identity matrix.

The weighting $w^{\sigma}$ was chosen according to equation (5.30) on page 177. This means that subtransfers with parameters that are either large or have a large variation (in absolute sense) are weighted less in the derivation of the principal components. In this way, the principal components are no longer completely determined by the large transfers. This gave the singular values listed in table 6.5.

Table 6.5: Singular values of the principal component decomposition

| $\sigma_{1}$ | $\sigma_{2}$ | $\sigma_{3}$ |
| :---: | :---: | :---: |
| 0.0636 | 0.0071 | 0.0010 |

The most striking fact that can be seen in figure 6.18 is probably the relatively large uncertainty regions for the off-diagonal subtransfers. As far as the transfers on the diagonal are concerned, results are not as good as in the previous SISO examples: the uncertainty regions are small compared to the nominal transfer but removing a structured error component hardly reduces them further.

The singular values shown in table 6.5 suggest that removing a second structured error component has little effect. The results obtained with two structured error components confirm this (not shown).

Because the tail of the basis was ignored in this example, the large uncertainty regions in the off-diagonal subtransfers can not be due to a slow decay rate of the tail-coefficients for these subtransfers. Ignoring the basis corresponds to an infinitely fast decay rate for the coefficients beyond those in the model parametrisation. The large uncertainty regions must be induced by the set of linear constraints $\mathcal{L}_{\boldsymbol{\theta}}^{w}$ for the operating points $w$.

Having a closer look at the constraints in $\mathcal{L}_{\theta}$, it becomes clear why the offdiagonal uncertainty regions turn out so large. In this example, $\mathcal{L}_{\boldsymbol{\theta}}$ basically contains three kinds of constraints:


Figure 6.18: Uncertainty regions in Nyquist plot. Dark areas, ' - .': one principal component removed; Light areas, '--': no principal component

1. The explicit bounds on the model parameters

$$
\begin{equation*}
-\bar{\theta}_{k} \leq \theta_{t r, k} \leq \bar{\theta}_{k}, \quad k=0, \ldots, \bar{k} \tag{6.1}
\end{equation*}
$$

2. The extended noise bounds

$$
\begin{equation*}
\forall j \in \boldsymbol{q}, \forall t \in \mathbb{T}| | y_{j}(t)-\sum_{i=1}^{p} \sum_{k=0}^{\bar{k}^{(j, i)}} \theta_{t r, k}^{(j, i)} \sum_{\tau=0}^{t} u_{i}(\tau) b_{k}^{(j, i)}(t-\tau) \mid \leq \bar{e}_{e, j}(t) \tag{6.2}
\end{equation*}
$$

3. The extended cross-covariance bounds

$$
\begin{align*}
& \forall \ell \in \boldsymbol{n}_{\boldsymbol{r}}, \forall j \in \boldsymbol{q} \\
& \left|\sum_{t \in \mathbb{T}} r_{\ell}(t) y_{j}(t)-\sum_{t \in \mathbb{T}} r_{\ell}(t) \sum_{i=1}^{p} \sum_{k=0}^{k^{(j, i)}} \theta_{t r, k}^{(j, i)} \sum_{\tau=0}^{t} u_{i}(\tau) b_{k}^{(j, i)}(t-\tau)\right| \leq \bar{c}_{\ell, j} \tag{6.3}
\end{align*}
$$

In previous examples it has already been found, that the extended noise bounds and the extended cross-covariance bounds are the bounds that eventually determine the size of the uncertainty regions. The explicit bounds $\bar{\theta}_{k}$ do not
contribute to a further reduction of the model uncertainty, they are too weak. However, the extended noise bounds and also the extended cross-covariance bounds provide basically a bound on the model parameters per output. They do not discriminate between the contributions of different inputs. Therefore, the extended noise bound can be small compared to the contribution to the output of one input, but large compared to that of another input. This is the case in this example. The extended noise bound is small compared to the contribution of input one to output one, but large compared to the contribution of input two to output one. Because the extended noise bound does not discriminate between different inputs, one may expect unstructured error regions of roughly the same size for all subtransfers of an output. This is confirmed by figure 6.19. The unstructured error is roughly the same size in all sub-transfers. Therefore the unstructured error is relatively much larger in the off-diagonal subtransfers.


Figure 6.19: Structured and unstructured errors in Bode amplitude plot. '-': nominal model; ' $:$ ': unstructured error, no princ. comp.; ':*': unstructured error, one princ. comp.; '--x’: structured error component

## Influence of other inputs

Things are made slightly worse by the following effect. To determine the unstructured error for, say, the subtransfer from input 2 to output 1 , linear pro-
gramming problems of the form

$$
\text { Maximise }\left(f \mid \boldsymbol{\theta}^{\prime}\right) \quad \text { subject to } \boldsymbol{\theta}^{\prime} \in \mathcal{L}_{\boldsymbol{\theta}}
$$

are solved, where $f$ is a vector containing the coefficients of the object function. In this example and for this subtransfer, $f$ is of the form

$$
f=\left[\begin{array}{lllllllll}
0 & \ldots & 0 & f_{0}^{(1,2)} & \ldots & f_{6}^{(1,2)} & 0 & \ldots & 0 \tag{6.4}
\end{array}\right],
$$

only the parameters in $\boldsymbol{\theta}$ pertaining to subtransfer $(1,2)$ are involved in the object function. This means that in the extended noise bounds, the parameters in $\theta$ belonging to different subtransfers to the same output can (and will) be manipulated such, that the contribution of the other inputs to the output compensate the contribution of the input at hand as far as possible. Provided the input signals are independent of each other, there is fortunately not much room to manoeuvre. In this example there is only one subtransfer from different inputs to the same output, subtransfer $(1,1)$. If the transfer from the first input to the first output is manipulated such, that it compensates the contribution from the second input to the first output for a sample instant $t_{1}$, then it is likely that this manipulation has an adverse effect for another sample instant $t_{2} \neq t_{1}$. Nevertheless, the more inputs a system has, the more freedom there is in this respect. Due to this effect, the uncertainty regions will increase with the number of inputs.

The exact structure of (6.4) is lost if one or more structured error components are used, see remark 4.14 on page 129. This means that there is not complete freedom any more to manipulate the parameters of other subtransfers. However, a lot of freedom is still left, so that essentially the same effect as discussed above will occur also if structured error components are used.

## Influence of weighting

The results above were obtained using weighting during the principal component analysis. The influence of this weighting can best be demonstrated by comparing the principal components that were obtained in this way by those obtained without weighting. Figure 6.20 shows in the upper plot the first two principal components with weighting and in the lower plot without weighting. The dotted vertical lines separate the parts of the principal components belonging to different subtransfers.

In the lower plot, the first principal component is only concerned with subtransfers $(1,1)$ and $(2,2)$, while the second has only significant non-zero values for subtransfer $(1,2)$ and $(2,1)$. At first sight this may seem a good result. After all, the uncertainty was constructed such, that uncertainty in the diagonal terms was governed (in a non-linear way) by a parameter $\alpha$ and the off-diagonal terms by another parameter $\beta$. However, the principal component


Figure 6.20: Principal components. (Top: with weighting; Bottom: without weighting; $\left.{ }^{\prime \prime}: \boldsymbol{\theta}_{A, 1} ;{ }^{*}: \theta_{A, 2}\right)$
analysis does not "know" this, it just happens to find this separation because the parameters that are related to $\alpha$ are an order of magnitude larger than those related to $\beta$. Moreover, the variation in subtransfers $(1,2)$ and $(2,1)$ (measured in $\ell_{2}$-sense) is relatively larger than that in subtransfers $(1,1)$ and $(2,2)$. We would therefore like to see that variation expressed in the first principal component and not in the second.

Things are better for the off-diagonal subtransfers in the upper plot. In this case the first principal component is largest ${ }^{\text {in }}$ the subtransfers ( 1,2 ) and $(2,1)$. This is achieved at the expense of some cross-coupling with subtransfers $(1,1)$ and $(2,2)$.

Figure 6.21 is the unweighted counterpart of figure 6.19. Comparing these two figures, it can be seen that the unweighted structured errors indeed ignore the off-diagonal terms. The structured errors in these subtransfers are much smaller than the unstructured errors now. The structured errors in the weighted case seem much more realistically sized. However, as far as the unstructured errors are concerned, this does not make much difference. In the weighted case they were about equally large with and without a structured error component and in the unweighted case the same applies.

If we look at the subtransfers on the diagonal, the unweighted results are


Figure 6.21: Structured and unstructured errors in Bode amplitude plot, no weighting. '-': nominal model; ':': unstructured error, no princ. comp.; ': *': unstructured error, one princ. comp.; '--x': structured error component
better than the weighted results. Because the structured error is in fact only concerned with these two transfers, it fits better to the error in these subtransfers. There is indeed a clear reduction of the unstructured error if one structured error component is removed.

In figure 6.22 the structured errors are plotted together with the Nyquist plot of the two diagonal entries. The Nyquist plots of the true transfers for $\alpha=0.01$ and $\alpha=-0.01$ are drawn using continuous lines. Some frequency points are marked with x's on these plots. For these frequencies, both the structured errors with and without weighting are plotted too. Those with weighting are drawn using dotted lines, those without using dashed lines. The dashed lines are on top of the dotted lines, but it can be seen that the dotted lines protrude a bit further.

For the diagonal entries weighting hardly changes the direction of the first structured error component in the Nyquist plane. Weighting of the principal components does change the bounds on the structured errors. As the dotted lines are longer than the dashed lines, $\bar{d}_{1}$ is larger and $\underline{d}_{1}$ is smaller with weighting than without. It is not exactly known what determined these bounds in both cases. All that can be said is that the different directions in parameter space of the structured error components with and without weighting must be



Figure 6.22: Nyquist plot of true transfers and structured error component (see text)
responsible for these different bounds. The Nyquist plot for the diagonal terms do not show much difference in direction for these two cases, but figure 6.20, which also includes the off-diagonal terms, clearly does.

In both cases the direction in the Nyquist plane of the first structured error is in close agreement with the true change in dynamics that occurs in the diagonal entries. Despite the large unstructured uncertainty, the structured error describes indeed the changes in dynamics that can be attributed to operating point changes!

## Sensitivity to noise level and data length

In this example the data sets consisted of 1200 samples. Reducing this to 900 samples has far reaching consequences. In figure 6.23 the central model parameters for subtransfer ( 2,1 ) are shown. The two parameter vectors are very close. There is a large relative error in the last parameter, but compared to the size of the other parameters, this error is not so big. Nevertheless, this error can have a large influence on the uncertainty regions. If the decay rate of the model parameters is estimated from these parameters, the decay rate for $N=900$ will be much less than that for $N=1200$. In the example above, that would not make such a big difference, because all tail effects are ignored there. However, if tail effects are taken into account, as they probably should, then the slower estimated decay rate will give a much larger influence of the tail. Recall that in the previous example it was found that the size of the tail effects is very sensitive to the decay rate of the model parameters.

To demonstrate that the data length of 900 samples is not unreasonably short, figure 6.24 shows the impulse response of subtransfer $(2,1)$ for $\beta=0.01$. The time constant of this impulse response is about 100 , so that a data length of 900 samples should not be considered too short to identify this subtransfer


Figure 6.23: Model parameters for subtransfer (2,1)
reliably.
One of the causes of the inaccuracy in the estimated model parameters for smaller data sets is, again, the influence of the other input on the output. The influence of input 2 on output 2 is much larger than that of input 1. A relatively small error in the estimated parameters of subtransfer (2,2) may therefore cause a significantly larger error (in a relative sense) in the parameters of subtransfer $(2,1)$. Asymptotically this mechanism will not be present: for infinite data length, the parameters estimated for subtransfer ( 2,1 ) are independent of those estimated for subtransfer ( 2,2 ), but for finite data lengths the cross-coupling is observed.

In the same way does an increase of the noise level in the outputs lead to problems. The higher noise level will lead to less accurately estimated model parameters. This effect will be relatively largest in the off-diagonal terms.


Figure 6.24: Impulse response of subtransfer (2,1), $\beta=0.01$

### 6.4.3 Conclusions

The example in this section demonstrated that relatively large errors can be expected in a subtransfer if the contribution of the input of the subtransfer to its output is small compared to the contribution of other inputs to that output. This is mainly due to two effects:

- The information that mainly determines the size of the error bounds is the extended noise bounds and the extended cross-covariance bounds. These bounds discriminate only between outputs, not between the effect of different inputs on the same output. Therefore these bounds are relatively large for small subtransfers.
- The auxiliary models will have a larger relative error for small subtransfers than for large subtransfers. This translates to a less accurately estimated decay rate for the model parameters of a small subtransfer. The size of the effects of the tail of the basis are very sensitive to this decay rate.

It was also shown, that small subtransfers are ignored by the first structured error components if no weighting is applied. If this is not desired, it can be easily avoided by weighting the model parameters before applying principal component analysis: scaling up these parameters before doing the principal component analysis and scaling them down afterwards will have the desired effect. However, this can not solve the problem of the relatively large uncertainty in small subtransfers.

### 6.5 Case study: Asynchronous motor

### 6.5.1 System and data

A schematic view of an asynchronous machine is shown in figure 6.25. In this scheme, four windings are drawn through which electrical currents can flow. The three large windings labeled $a, b$ and $c$ represent the stator windings of the machine. These are fixed to the house of the machine. If current or voltage sources are connected to these windings, stator currents $I_{a}, I_{b}$ and $I_{c}$ will flow. The fourth winding, labeled $r$, is the rotor winding. This winding is mounted on a shaft that may rotate with respect to the rest of the machine. The rotor winding is short-circuited.

The stator windings act as inductances: currents $I_{a}, I_{b}$ and $I_{c}$ will generate a magnetic field inside the machine. This field is indicated here by its magnetic flux density vector $B$. By proper manipulation of the currents $I_{a}, I_{b}$ and $I_{c}$ a vector $B$ can be realised that rotates around the main axis of the machine. This rotating magnetic field causes a changing magnetic flux through the rotor winding $r$. This induces an electric field in the rotor winding. As the rotor
winding is short-circuited, a rotor current $I_{r}$ will flow. The rotor current $I_{r}$ and the magnetic flux density $B$ lead to a Lorentz force $F_{L}$ on the rotor. As a result of this force, a torque is exercised on the rotor, which will start to rotate.


Figure 6.25: Schematic representation of an asynchronous motor

In case the rotor rotates as fast as the magnetic flux density $B$, the magnetic flux through the rotor winding is constant. No rotor current $I_{r}$ will flow any more and consequently there will be no Lorentz force and no torque will be exercised on the rotor. This situation corresponds to the no-load situation. As the motor is normally not operated in a way in which no torque is exercised on the rotor, this situation will hardly ever occur. Consequently it will hardly ever be the case that the rotor rotates synchronously with the magnetic field inside the machine. This is why this kind of motor is called an asynchronous motor.

Figure 6.25 is highly simplified. The stator inductances do not consist of a single winding but have multiple windings. Hereby a larger magnetic field is induced inside the machine for the same stator currents $I_{a}, I_{b}$ and $I_{c}$. In the motor used for the experiments reported in this section, the rotor consists of three inductances, each having multiple windings. These inductances are mounted under angles of $60^{\circ}$ relative to each other. This improves the rotational symmetry of the machine. The inductance of the stator windings is further increased by the presence of iron in the construction of the machine.

This too aids in generating a large magnetic flux density from stator currents $I_{a}, I_{b}$ and $I_{c}$. Apart from the fact that the drawing is too simple, also many side effects have not been mentioned in the discussion above. One of these side effects, saturation of the magnetic material for high flux densities, will be encountered later.

In the test setup used in this section, voltages $U_{a}, U_{b}$ and $U_{c}$ are applied to the stator windings. These voltages are generated by an inverter. This device transforms a DC voltage into a triple of AC voltages. These AC voltages have the same frequency but are shifted in phase. This is done in a way that is suitable for their application as driving voltages for the three stator windings. The stator voltages $U_{a}, U_{b}$ and $U_{c}$ are used as the input signals for the machine, the resulting currents in the windings are considered the output signals of the machine. The interested reader is referred to (Gorter, 1997, Appendix A) for a more thorough description of the test-setup: the experimental data described in this section are taken from this thesis.

As the machine under test has three stator windings, the machine is called a three-phase machine. Under certain symmetry conditions, the three stator voltages and currents can be transformed to two stator voltages and currents of an equivalent two-phase orthogonal machine. This approach is used here. The actual machine considered in this section is therefore a fictitious two phase machine. The two stator windings of this machine are referred to as $\alpha$ and $\beta$. This fictitious machine is related in a well-defined way to the physical threephase machine. By considering the equivalent two-phase machine, the original three-input three-output problem reduces to a two-input two-output problem. In this two-phase machine, the two stator windings are perpendicular to each other. Neglecting leakage fluxes and assuming linear magnetics, this means that there is no magnetic coupling between these two phases. Therefore the two-input two-output two-phase motor can be split into two decoupled SISO systems. As practical conditions are not completely ideal, it appears that there is a small coupling between the two SISO systems. Therefore the system is regarded here still as a two-by-two MIMO system.

The experimental data used in this section were collected with a locked rotor: the rotor position was fixed so that it could not rotate any more with respect to the stator. Under this condition and assuming full decoupling between the two stator windings, the electrical circuit drawn in figure 6.26 is "seen" in each stator port. $R_{s}$ represents the resistance of the stator winding. The ideal transformer in the middle of the figure models the magnetic coupling between the stator windings and the rotor windings. $L_{\sigma}$ is the so-called leakage inductance and $L_{m}$ the main inductance. $R_{r}$ is the resistance of the rotor winding. A more detailed discussion can be found in (Gorter, 1997).

The relation in frequency domain between any of the two stator voltages


Figure 6.26: Locked-rotor equivalent circuit


Figure 6.27: Bode amplitude plot of locked rotor equivalent circuit
$U_{t}(j \omega)$ and the corresponding stator current $I_{t}(j \omega)$ is

$$
\frac{U_{t}(j \omega)}{I_{t}(j \omega)}=R_{s}+j \omega L_{\sigma}+\frac{j \omega R_{r} L_{m}}{j \omega L_{m}+R_{r}}
$$

or, equivalently,

$$
\frac{I_{t}(s)}{U_{t}(s)}=\frac{L_{m} s+R_{r}}{L_{m} L_{\sigma} s^{2}+\left(R_{s} L_{m}+L_{\sigma} R_{r}+L_{m} R_{s}\right) s+R_{r} R_{s}}
$$

which is the transfer function from the input $U_{t}(s)$ to the output $I_{t}(s)$. This transfer function has two real poles and one zero. Figure 6.27 shows the Bode amplitude plot for this transfer based on the values of $R_{r}, R_{s}, L_{m}$ and $L_{\sigma}$ as specified by the manufacturer of the motor. For completeness, these values are listed in table 6.6. The locations of the poles and the zero are indicated by vertical dotted lines. The zero is located between the two poles.

Table 6.6: Machine parameters

| $R_{s}$ | 0.220 | $\Omega$ |
| :---: | ---: | :--- |
| $R_{r}$ | 0.348 | $\Omega$ |
| $L_{m}$ | 97.4 | mH |
| $L_{\sigma}$ | 9.0 | mH |

The inputs of the motor were excited by generalised pseudo-random binary noise sequences. A generalised PRBS differs from a normal PRBS in that its spectrum is smoother for higher frequencies. The frequency spectrum was designed such, that much energy was put in the frequency region where the poles and the zero are located. In this way, the experimental results will contain more energy in the frequency region that is of interest for good estimation of the physical parameters of the machine. The inverter normally supplying the machine was used to supply it with the designed generalised PRBS signals.

As a result of saturation of the magnetic material inside the machine, the value of the main inductance $L_{m}$ decreases for large magnetising currents $I_{m}$. The magnetising current has been selected with a DC-current value of $I_{t}$. To investigate the influence of saturation and in particular whether this influence can be described by a structured error component, experiments were done with different DC levels for $I_{t}$. Different DC levels are interpreted as different operating points for the asynchronous motor. These levels gave ten operating points, labeled $A$ through $J$. Table 6.7 gives the DC levels. The DC levels for $I_{t}$ were translated to corresponding DC levels for $U_{t}$ and the excitations discussed in the previous paragraph were superimposed on these DC levels.

The experimental results in operating points $A$ and $C$ turned out to be unsuitable for identification of accurate models: the residuals obtained in these operating points were so large that the model uncertainty would be completely dominated by these operating points. This was explained by the fact that the power electronics in the inverter is operating in these operating points only just above the threshold values required for proper operation. The eight remaining operating points were used during the estimation and model uncertainty bounding.

The clock frequency of the generalised PRBS was 250 Hz . Measurements were taken at 1 kHz and were later downsampled to 250 Hz . As a standard part of data preprocessing, inputs and outputs were scaled such, that their standard deviations were approximately one in all operating points. The physical ranges of these signals are such, that no numerical problems are to be expected if this scaling would not have been applied.

Table 6.7: $D C$ level of $I_{s}^{S}$ in $\alpha$-axis and $\beta$-axis for different operating points

| wp. | $\alpha$-axis <br> $[\mathrm{A}]$ | $\beta$-axis <br> $[\mathrm{A}]$ |
| :---: | :---: | :---: |
| $A$ | -0.0473 | -0.0177 |
| $B$ | 4.0178 | -0.2477 |
| $C$ | -0.1031 | 3.6220 |
| $D$ | $3.6400 / \sqrt{2}$ | $3.8884 / \sqrt{2}$ |
| $E$ | 12.2533 | -0.2759 |
| $F$ | -0.1270 | 11.6856 |
| $G$ | $11.3079 / \sqrt{2}$ | $11.3677 / \sqrt{2}$ |
| $H$ | 20.0661 | -0.3854 |
| $I$ | -0.1682 | 19.4153 |
| $J$ | $19.4526 / \sqrt{2}$ | $20.0339 / \sqrt{2}$ |

### 6.5.2 Results and discussion

The two-phase asynchronous motor is very similar to the MIMO simulation example in the previous section. Here too the subtransfers on the diagonal are much larger than the off-diagonal transfers. Based on the results in the previous section, no weighting was applied before determining the structured error components. The reasoning behind this is as follows. Because the extended noise and extended cross-covariance bounds do not distinguish between different inputs, the unstructured error can be expected to be relatively large in the off-diagonal transfers. There is not much that the structured error can do about this, so we might as well concentrate on the subtransfers on the diagonal while determining the structured errors. In these subtransfers, the structured errors do have the potential to reduce the unstructured error. Tuning the structured error to the subtransfers on the diagonal is achieved automatically if no weighting is applied, as was demonstrated in the previous section.

The unstructured errors and the structured errors were determined without taking the influence of the tail of the basis into account. The reason for this will become clear shortly. A basis was generated by the system

$$
G_{b}(z)=\frac{1}{z-0.945}
$$

The pole location of the basis generating system was based on preliminary identification runs.

The singular values that were obtained in the principal component analysis are give in table 6.8.

The Bode amplitude plots of the central model and of the unstructured error bounds for zero, one and two structured error components are given in

Table 6.8: Singular values for the structured error components

| $\sigma_{1}$ | $\sigma_{2}$ | $\sigma_{3}$ | $\sigma_{4}$ | $\sigma_{5}$ | $\sigma_{6}$ | $\sigma_{7}$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 0.1554 | 0.1114 | 0.0456 | 0.0320 | 0.0247 | 0.0165 | 0.0157 |

figure 6.28. There is so little difference between the three unstructured error bounds that it is not really relevant which is which. Clearly the unstructured errors are too large to have any hope for performance if they are used in a robust control design.


Figure 6.28: Central model ('-') and unstructured error bounds ('--')

Because the tail of the basis was ignored, many causes for the large unstructured errors can already be excluded beforehand. The only possibly reasons for the large unstructured error are large noise bounds and large cross-covariance bounds. Note that the extended noise bound differs in this example only from the unextended noise bound by the estimated bound on the transient. The transient only takes significant values during the first 400 samples. As the data sets consist of over 2000 samples, the transient can not be the explanation for
a large extended noise bound.
Analysing the local unstructured errors it turns out that the global unstructured error for the transfer from $U_{\alpha}$ to $I_{\alpha}$ is determined for low frequencies only by the unstructured error for operating point $I$. For higher frequencies operating points $F$ and $H$ become relevant as well. For the transfer from $U_{\beta}$ to $I_{\alpha}$ the global unstructured error is also mainly determined by operating point $I$. In this case operating point $G$ also plays a minor role. For the subtransfers to the other output, $I_{\beta}$, the uncertainty in the transfer from $U_{\alpha}$ is determined solely by operating point $J$ and the uncertainty in the transfer from $U_{\beta}$ is determined by operating points $D, E, G, H$ and $J$.

Table 6.9 provides further insight into the causes of this. In this table the rms-value of the residuals is listed, together with the value of the estimated noise bound $\bar{e}$ and the signal to noise ratio at the output. The latter is estimated as the ratio between the energy in the output signal and the energy in the residuals. If attention is restricted to the subtransfers to output $I_{\alpha}$, the unstructured error is mainly determined by operating point $I$. According to table 6.9 the rms-value of the residuals is not excessively large in this operating point. The rms-value of the residuals in operating point $H$ is larger, but operating point $H$ plays a much smaller role in the unstructured error bounds for this output. However, the estimated noise bound is not based on the rms-value of the residuals but on the maximum absolute value. Apparently there have been some unfortunate samples in the residuals of operating point $I$, because its noise bound $\bar{e}$ is considerably larger than that of operating point $H$, which had even a larger rms-value for the residuals than operating point $I$ !

Table 6.9: Noise levels, noise bounds and signal to noise ratios

| $\alpha$-axis |  |  |  | $\beta$-axis |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| wp. | rms $\epsilon$ | $\bar{e}$ | $\mathrm{~S} / \mathrm{N}[\mathrm{dB}]$ | wp. | rms $\epsilon$ | $\bar{e}$ | $\mathrm{~S} / \mathrm{N}[\mathrm{dB}]$ |
| $B$ | 0.1113 | 0.4180 | 19.1 | $B$ | 0.0506 | 0.2494 | 25.9 |
| $D$ | 0.0641 | 0.2880 | 23.9 | $D$ | 0.0762 | 0.3712 | 22.4 |
| $E$ | 0.1647 | 0.5485 | 15.7 | $E$ | 0.0851 | 0.3969 | 21.4 |
| $F$ | 0.1284 | 0.6952 | 17.8 | $F$ | 0.0644 | 0.2865 | 23.8 |
| $G$ | 0.1759 | 0.6308 | 15.1 | $G$ | 0.1123 | 0.4471 | 19.0 |
| $H$ | 0.2057 | 0.6230 | 13.7 | $H$ | 0.1173 | 0.5114 | 18.6 |
| $I$ | 0.1754 | 0.8051 | 15.1 | $I$ | 0.0778 | 0.3076 | 22.2 |
| $J$ | 0.1706 | 0.5808 | 15.4 | $J$ | 0.2311 | 0.7513 | 12.7 |

Another conclusion that can be drawn from table 6.9 is that the signal to noise ratio is considerably worse than in the previous simulation examples. At signal to noise ratios of 20 dB or worse, the unstructured error becomes very
large. The structured error components can only be expected to reduce the unstructured error if the error induced by operating point changes is large or at least comparable to other error sources. This explains why removing one or two structured error components from the unstructured error hardly influences the size of this error in figure 6.28 .

There is another effect that increases the unstructured error: the unstructured error describes for every frequency the largest possible uncertainty that can be encountered for that specific frequency. It does not contain information of the sort "if the uncertainty for this specific frequency is at its bound, the uncertainty for all remaining frequencies can only be this large." Any correlation that exists between the uncertainty at different frequencies is lost in the algorithm. More specifically, the linear constraints on the model parameters $\mathcal{L}_{\theta}$ still contain information of this type but using this information in different, independent linear programming problems to bound the unstructured error looses this information.

It is exactly this type of information that is present in the structured error components. However, if no "uncertainty directions" dominate the others, (too) many structured error components are needed before the unstructured errors start to decrease. Moreover, a similar loss of information occurs here as well: the correlation that exists between the structured error components and also between the structured errors on the one hand and the unstructured error on the other hand is lost. This is not a real problem if the structured components rapidly decrease in size, but it can become a significant source of conservatism if the structured components are about the same size.

For completeness, the unstructured error that is obtained if the tail of the basis is not ignored is plotted in figure 6.29. As could be expected, the unstructured error is increased for all subtansfers by incorporating the influence of the tail of the basis.

### 6.5.3 Computational load

The unstructured errors were bounded in this example for 30 frequency points. For every frequency point, the error in eight directions in the complex plane was bounded and this was done for all four subtransfers. Having eight operating points, this gives a total of 7680 linear programming problems that need to be solved to bound the unstructured errors. If a different number of structured error components is removed from the unstructured errors, another set of 7680 linear programming problems needs to be solved. The data sets for each operating point were not exactly the same size. On average they contained 2125 samples. This gives for every output 4250 linear constraints because of the extended noise bounds. The cross-covariance bounds account for another eight constraints, and the explicit bounds on the model parameters represent another 40 constraints. On average there were a total 8588 constraints in each


Figure 6.29: Central model ('-') and unstructured error bounds ('--'), tail effects included
linear programming problem.
If no structured errors are used, solving the 7680 linear programming problems took about 35.5 hours (on a 90 MHz pentium running no other tasks and with sufficient memory to make sure that no swapping occurs.) If one or more structured errors are used, computing time increases to about 91 hours. This increase is probably related to remark 4.14 on page 129: if no structured error is used, the unstructured error in a subtransfer is independent of the extended noise bounds for other outputs than the one corresponding to that subtransfer. If one or more structured error components are used, this is no longer true.

After solving all linear programming problems, the algorithm sets off to collect all results in a binary output file. Intermediate results are saved to disk to keep total memory requirements of the algorithm reasonable. Saving the results is therefore mainly a matter of copying data from one file to another. In the configuration that was used for these calculations, this took another eight hours. The configuration was not optimised towards minimising this time.

The computing time required to perform the other steps in the algorithm are neglegible compared to that required for the two steps mentioned above.

Therefore it takes about 300 hours or $12 \frac{1}{2}$ days of computing time to get results for zero, one and two structured error components.

Literature reports, see (Murtagh, 1981), that the time required to solve one linear programming problem is roughly proportional to the cube of the number of constraints in the problem. The number of constraints is mainly determined by the number of extended noise bounds. For $q$ outputs and $N$ samples in a data set, $2 N q$ constraints due to the extended noise bounds are found. The number of linear programming problems that needs to be solved is proportional to $m$, the number of directions that are bounded in the complex plane. It is also proportional to $n_{\Omega}$, the number of frequencies for which the unstructured error is bounded, to the number of subtransfers and to $n_{w}$, the number of operating points. If $p$ denotes as before the number of inputs in the system, the computing time $T_{l p}$ to solve all linear programming problems satisfies

$$
T_{l p} \sim m n_{\Omega} n_{w} p q(N q)^{3}
$$

The asymmetry between inputs and outputs results from the fact that the number of constraints in the linear programming problem is proportional to the number of outputs only.

Taking into account that the number of parameters that need to be estimated is proportional to the number of subtransfers, and that this should have its influence on the length of the experimental data sets, it is clear that the required computing time gets out of hand very quickly if the number of inputs and/or outputs of the system grows.

The computational load can be reduced if it is possible to select a priori most samples from the extended noise bound that will correspond to active constraints. To investigate this, consider the histogram in figure 6.30. In this histogram the distribution of residuals for output 1 in operating point $J$ is plotted. Also the distribution in the subset of residuals corresponding to active constraints is shown. No structured error components were used in this example. Judging from this histogram, a considerable number of constraints can be skipped a priori: if all constraints for which the residual is small are skipped, the number of constraints will decrease significantly, but hardly any constraints that would become active will be dropped.

It is interesting to see that also constraints for which the residuals are very small can become active. Making such a constraint active involves manipulating the model parameters such, that the mismatch between process output and model output becomes so large for that particular sample that it hits the noise bound, without violating any of the other noise constraints. Apparently there is sufficient freedom in the model parametrisation and in the extended noise bound to make it possible that this happens for some of the small residuals.

The situation gets slightly worse if the tail of the basis is taken into account. This gives the histogram shown in figure 6.31 on page 241. In this case, where the difference between the extended noise bound and the residuals gets larger,


Figure 6.30: Histogram of all residual values (light) and of those corresponding to active constraints (dark)
more constraints corresponding to small residuals get active. This shows that it is not a trivial exercise to formulate a selection rule selecting the majority of active constraints but skipping as many inactive constraints as possible.

Even if such a selection rule can be found, it will not solve the problem of the heavy computational load, it will only alleviate it to some extent.

### 6.5.4 Conclusions

As this example is very similar to the previous simulation example, most observations that were made for that simulation example can also be made for this example. For example, the unstructured error in the off-diagonal subtransfers is of the same order of magnitude as that in the diagonal subtransfers. This makes the unstructured error in the off-diagonal subtransfers relatively very large.

In this example the unstructured errors can also be called large in the subtransfers on the diagonal. This is due to the fact that the signal to noise ratio at the output is 20 dB or worse. As the unstructured errors resulting from a consequently large noise bound are truly unstructured, removing one or two structured error components hardly makes any difference. For such noise levels, no significant gain can be expected from the use of structured error components. This is a serious limitation for practical application of the algorithm.


Figure 6.31: Histogram of all residual values (light) and of those corresponding to active constraints (dark), influence of tail of basis not ignored

Another effect that inflates the unstructured error and that becomes more important if the noise levels become higher is that the correlation that may exist between the unstructured error at different frequencies is lost. Similarly the correlation that may exist between different structured error components is lost. This effect is not restricted to this example, it just becomes more important due to the higher noise level.

Another practical problem with application of the algorithm is that the computational load gets out of hand very quickly if the number of inputs and/or outputs increases. For this relatively small example already four days computing time is required to bound the unstructured error using one or two structured error components. Bounding the uncertainty for systems with say sixty inputs and ten outputs is therefore presently completely out of the question.

### 6.6 Conclusions

The algorithm gives good results for the SISO simulation examples in this chapter. This is not restricted to the example that "fits" the uncertainty description of the algorithm. Also for structure in the model uncertainty that can not be described exactly by the algorithm good results are obtained. In this case the real structure is obviously not found by the algorithm, but it finds a sufficiently good approximation of it to reduce the unstructured error significantly.

By tracing back through the intermediate results, the different contributions
to the uncertainty bounds can be separated well. In this way it was found that the uncertainty bounds are very sensitive to the exponential decay rate that is estimated for the model parameters. A small decrease in the decay rate can cause a large increase in the model uncertainty bounds.

For MIMO systems it turned out that the prior knowledge determining the size of the uncertainty bounds does not discriminate between different subtransfers to the same output. Therefore the unstructured error is of roughly the same order of magnitude for all subtransfers to the same output. If the influence of one input on an output is much smaller than that of another input on the same output, the unstructured error will be relatively much larger for the subtransfer of the former input-output pair.

Weighting can be applied successfully to make small subtransfers contribute to the structured error components. Without weighting, only large subtransfers will determine the structured error components. However, the usefulness of this is decreased by the previous observation.

For signal to noise levels of 20 dB or worse, the noise bounds get so large that unstructured errors get of the same order of magnitude as the nominal transfer. There is very little structure in these errors, so that removing structured error components does not improve this situation. It is not quite clear to what extent this result is restricted to the example worked out in section 6.5. Nevertheless it seems clear that application of the algorithm is only useful in cases where there is a good signal to noise ratio.

The computational load of the algorithm increases rapidly with the number of inputs and outputs of the system and with the length of the data sets. For very patient users, two by two systems are within the reach of the algorithm. For larger systems the computing time on a 90 MHz pentium gets currently in the order of weeks.

## 7

## Software components for CASID

### 7.1 Introduction

7.2 Object-oriented aspects of C++
7.3 Matrices and related data structures

7.4 Building on vectors and matrices<br>7.5 Keeping all data of an identification session<br>7.6 User interface<br>7.7 Performance

### 7.1 Introduction

### 7.1.1 Overview

In the previous chapter some results obtained with the algorithm of chapters 4 and 5 were presented. Obviously the algorithm has been implemented in a computer program of some form. The initial idea was to implement the algorithm in Matlab. However, thinking about how to arrange the data operated on and generated by the algorithm, it became clear that Matlab's programming language would provide insufficient support to complete this task in a structured and maintainable way. These considerations are given in section 7.1.2 below. In this section it is also argued why C++ was chosen as the alternative to Matlab.

The first task that was faced after deciding to implement the algorithm in $\mathrm{C}++$ was to implement matrices and related data structures in $\mathrm{C}++$, so that in $\mathrm{C}++$ equally powerful expressions operating on matrices are possible as in Matlab. This is described in section 7.3. As soon as these structures are available, more complicated structures can be built from these, which was the original reason for implementing the software in C++. These structures are discussed in section 7.4. They were designed with the goal of being more generally applicable in the field of Computer Aided System IDentification than just for the algorithm described in this thesis. Finally, section 7.5 describes the way all these components were arranged in a single data structure that keeps all relevant information of an identification session.

The sections on the implementation of matrices and more complicated data structures are preceded by a section on the object-oriented aspects of C++. The discussion of this subject is kept to the minimum required to appreciate some of the design decisions that were made in the implementation. The chapter concludes with user interface aspects and some remarks concerning the performance of Matlab and C++.

### 7.1.2 Comparison of programming in Matlab and programming in C++

Matlab has become a very popular tool in the fields of control engineering and systems theory. Compared to programming languages as C or Pascal, Matlab has a number of important advantages:

1. Matlab supports dynamically sized real and complex matrices. The user has the freedom to change at will the dimension of a matrix contained in a variable, and he/she does not have to bother with allocating a new chunk of memory to store the new matrix and releasing the memory containing the previous value. Moreover, if one wants to multiply a matrix a with a matrix $b$ and store the result in variable $c$, one can simply write in Matlab
```
c = a * b;
```

In C this would require a subprogram of the form

## Code fragment 7.1

```
for (rowCnt = 0; rowCnt < m; rowCnt++)
    for (colCnt = 0; colCnt < n; colCnt++)
    {
            c[rowCnt][colCnt] = 0;
            for (inCnt = 0; inCnt < p; inCnt++)
                c[rowCnt][colCnt] += a[rowCnt][inCnt] * b[inCnt][colCnt];
        }
```

The cryptic $C$ syntax probably looks daunting at first, but that is not the real problem with code fragment 7.1. The C -code requires five lines of code for the same operation that is carried out by one line of code in Matlab. Moreover, the C code requires that integer variables rowCnt, colCnt and inCnt have previously been defined and the variables or constants $\mathrm{m}, \mathrm{n}$ and p have to be set up previously such that they contain the number of rows in $a$, the number of columns in $b$ and the "inner dimension," respectively.
2. Matlab provides easy access to high quality routines for matrix inversion, QR-decomposition, Schur-decomposition, etc. Standard C provides no routines for this.
3. Matlab has flexible and powerful functions for plotting and 3D visualisation.
4. C is a compiled language. This means that the C-code is converted to a series of machine instructions. Execution of these machine instructions will have the same effect as execution of the C-code would have, provided we had a machine that knew how to execute C-code. The process of conversion is called compilation. A program can be divided into different modules and these modules can be compiled independently. Combining these different modules to a program is done in the linking-phase.
Every tiny change in a program requires a recompilation of at least the unit that changed. To reflect the change in the compiled unit in the final program, the program has to be relinked. This can be quite timeconsuming.
Matlab has an interpreted language. This means that the Matlab interpreter figures out during execution of a user-written function, what the function is supposed to do. One can start testing a new function as soon as it has been saved to disk. This saves a lot of time during the prototyping phase of an algorithm. The price that has to be paid for this is that execution of the function is somewhat slower, see below.
5. Despite the existence of an ANSI standard for C, most C-programs not specifically designed to be platform independent tend to be platform dependent, as anyone will know who has ever gone through the trouble of porting, say, a C-program written for a UNIX machine to DOS. This problem can be alleviated to some extent by using libraries of routines that are explicitly supported by its developers on many platforms. For example, several software libraries exist to support writing graphical user interfaces on different windowing systems. The software library hides all differences between the various windowing system and provides a uniform interface on all supported platforms. However, high quality software that is available on as many platforms as Matlab is often both scarce and expensive.

There is another advantage of Matlab that follows from the previous advantages and perhaps clever marketing of "the Mathworks," its manufacturer. By now Matlab has become a de-facto standard in the field of control design and system identification, at least as far as research institutions in this field are concerned: someone who has developed a new algorithm and who wants to share the code he has written with the rest of the world is likely to do so in the form of Matlab routines.

All disadvantages of C apply in principle also to $\mathrm{C}++$, but $\mathrm{C}++$ provides more tools to work around the disadvantages. For example, it is possible in C++ to define a new type, say matrixCls, such that variables $a, b$ and $c$ behave like matrices. (The postfix -Cls stands for "class," see below.) Code fragment 7.1 may then read in $\mathrm{C}++$ :

```
matrixCls a, b, c;
\(c=a * b ;\)
```

The only drawback with respect to the Matlab-code is, that the variables a, $b$ and $c$ need to be declared explicitly.

This example already illustrates the first of a number of important advantages of C++ over Matlab:

1. The Matlab-language is restricted to variables of type matrix. Scalars and vectors are considered special cases of matrices. In C++ a variable that is used as a counter in a loop can be declared to be an integer variable. Calculating with a single integer value is much faster than calculating with a matrix that may be either real or complex and which can only be found to be a scalar during the execution of the routine.

Not only can C++ use simpler types than Matlab in situations where that is convenient, it can also define types that are more complex than matrices. In Matlab all objects have to be described in terms of matrices. For example, a state-space description of a transfer function $G(z)$ is represented in Matlab by a quadruple of matrices, say a, b, c, d, where

$$
G(z)=\mathrm{c}(z I-\mathrm{a})^{-1} \mathrm{~b}+\mathrm{d} .
$$

Adding two transfers $G_{1}(z)$ and $G_{2}(z)$ can be achieved with

## Code fragment 7.2

[asum, bsum, csum, dsum] =parallel (a1, b1, c1, d1, a2, b2 , c2, d2);
Surely this is awkward. In C++ a type SStransferCls can be defined, such that one could write instead of fragment 7.2

## Code fragment 7.3

SStransferCls G1, G2, Gsum;

Gsum $=\mathrm{G} 1+\mathrm{G} 2$;
(In Matlab this problem is recognised in several toolboxes. These toolboxes provided packed representations for systems, signals, etc. Unfortunately the representations used by different toolboxes are incompatible.)
A number of benefits follow from C++'s capabilities to define new types and operators for these types:
(a) Much more detail concerning the representation of a type can be hidden from the user. For example, discrete time signals are represented in Matlab using matrices where different scalar signals appear in different columns of the matrix and different sample instances in different rows. In C++ a type signalcls can be defined such that the user does not need to know. This is generally referred to as information hiding.
The importance of information hiding is that if it is decided for whatever reason to interchange the role of rows and columns in the implementation, none of the code using variables of this type needs to be changed. This makes maintaining the code much easier.
Obviously the code defining the type needs to undergo some changes, but this too can in some cases be a matter of only a few lines of code. The concept of information hiding can be applied not only at the level of using the type, but also at the level of implementing the type.
(b) Code can be implemented more efficiently. In the implementation of the state-space type SStransferCls it need not constantly be checked whether the dimensions of the system-, input-, output- and input/output matrices are compatible. This is checked only when these matrices are changed or first created. From then on it can be relied on that they are compatible.
(c) Provided sufficient tools have been implemented to support this, the developer can code his algorithm at the level of abstraction at which he designed it. For example, if p1, p2 and p3 are variables of type polynomialCls that behave like polynomials, one could write in $\mathrm{C}++$

$$
\mathrm{p}^{3}=\mathrm{p} 1+\mathrm{p} 2 ;
$$

In Matlab polynomials are represented by vectors containing the coefficients of the polynomials. Before two polynomials can be added in Matlab, one has to make sure that the two vectors representing them have the same length. This may involve padding one of the two vectors with zeros.
2. As $\mathrm{C}++$ is a compiled language, the translation of $\mathrm{C}++$-source code to instructions that the machine understands occurs in the separate steps of compiling and linking. During the actual execution of the code, the machine is not concerned with this any more. There is less overhead during the execution of the code and consequently the code executes faster.
Note that the same difference between Matlab and C++ was earlier considered a disadvantage of $\mathrm{C}++$ during the prototyping of an algorithm.
3. Related functions can be grouped in a single file in C++. A file can contain several routines and the name of these routines can have long, meaningful names. If a programmer gives one of his routines the same name as another, this is most likely detected during compilation or linking. Also it is possible to make some auxiliary routines available only inside the module in which they are used. Outside the module such routines are unknown. Different routines with the same names can coexist without problems in this way. (This does not apply to routines that need to be available in all modules.)

In Matlab every separate function is stored in a separate file. If one wants to run these routines on all platforms supporting Matlab, the MS-DOS/MS-Windows platform restricts the length of the name of the routine to eight characters. (The recent successor of this platform, Windows 95, supports file names up to 256 characters, but there is no MatLAB version yet that supports this.) Moreover, to avoid conflicts with routines with the same name in other toolboxes it is good practice to start all routines in a toolbox with the same sequence of two or three characters. This further restricts the effective length of a routine name. These effects make the code both more difficult to read and more difficult to maintain.
4. More diagnosing tools are available for C++ compilers than for Matlab. Although Matlab has limited debugging facilities, the debugging tools for $\mathrm{C}++$ are in general much more powerful. Finding out which parts of a program are time-critical is also better supported in C++.

Comparing the advantages and disadvantages of C++ and Matlab, the advantages of $\mathrm{C}++$ seem to get more important for larger projects with more complex data structures. For smaller projects whose data can be represented sufficiently well using matrices only, Matlab seems to be the proper tool.

### 7.1.3 Aims of the software structures in this chapter

It was felt that the data structures that are required for the implementation of the algorithm of chapters 4 and 5 are insufficiently supported by Matlab. It was therefore decided to implement the algorithm in C++. At the time when this was decided, no routines had been written to make matrices, polynomials, signals, state-space realisations of transfer functions etc. available in C++. In this respect this decision was a big step backward. However, filling this gap between C++ and Matlab is an effort that one needs to go through only once. Provided the tools that are developed are powerful and general enough, they can be applied also for other projects. This has been the aim in the development of the data structures presented in this chapter.

Due to time constraints and the fact that developing the software tools discussed above was only a derived goal of the research reported in this thesis, some of the functionality that was considered desirable in a general library for Computer Aided System IDentification was not implemented. In these cases, care has been taken to make sure that the overall structure permits relatively easy extension in this respect.

This chapter is not concerned with numerics. The difficult problem of implementing efficient, robust and reliable numerical algorithms to operate on the data was circumvented by using software libraries for these purposes that were available on the internet. Routines for QR-decomposition, singular value decomposition and other operations from linear algebra were taken from the Lapack library. This library is documented in (Anderson et al., 1995). The linear programming problems that are an important part of the algorithm were solved by version 2.0 of the LP_SOLVE package by Michel Berkelaar and Jeroen Dirks, (Berkelaar, n.d.). These libraries were treated as black boxes as far as possible.

### 7.2 Object-oriented aspects of C++

In the previous comparison between Matlab and C++ a big difference between the two has not been mentioned yet: C++ supports object-oriented programming, while the Matlab programming language does not.

The phrase "object-oriented" is used in many different contexts these days and in each of these contexts it is usually presented as something advantageous. Indeed, in the field of software development object-orientation is expected to give faster development and maintenance of code, more natural ways of translating a real-world problem into a program solving it, more reuse of existing code, etc. In this light it would have seemed reasonable to mention objectorientation in the previous section as an advantage of $\mathrm{C}++$.

This section does not consider whether the promises of object-oriented design have been fulfilled in recent years and if so to what extent. It is restricted to explaining as little as possible about object-oriented programming to understand the design of the software that will be outlined in the next sections.

In (Pree, 1994) the following basic concepts of object-orientation are distinguished:

## 1. Data abstraction

2. Inheritance
3. Polymorphism and dynamic binding

These will be discussed below.

## Data abstraction

Data abstraction is achieved by the use of objects. In (Pree, 1994) an object is described as follows: "Objects are instances of abstract data types. Thus an object is an entity that has attributes (= data representing an object's state) and provides certain operations that are defined for the particular object."

An example may clarify this. Complex numbers all share the same abstract data type, the type of complex numbers. A variable of type complex number is an instance of this type. If a program contains three complex variables $z 1$, $z 2$ and $z 3$, the program contains three instances of type complex.

It may be that the complex variables are represented as a pair of real numbers, corresponding to the pair of real and imaginary part. This pair of real numbers are in this case the attributes of the data. To make a useful type of the type of complex numbers, variables (or expressions) of this type should "know" how to add to or subtract from another complex number, multiply by a real number or another complex number, etc. These are the operations that are provided for the object. They are called the methods of an object.

A complex number may equally well be defined as a pair of real numbers corresponding to its modulus and its argument. The same operations should be provided for the object as in the previous implementation using real and imaginary part, but these operations should now operate differently on the internal variables. It should be completely invisible which of the two representations for complex numbers are used internally by the object. This is how data abstraction is achieved: a complex number "behaves" according to the usual axioms of complex numbers, but how this is realised is unimportant.

## Inheritance

A data type can be derived from another data type through inheritance. The derived data type is called the subclass and the data type derived from is the superclass. The subclass is a descendant of the superclass. A subclass has all the attributes and methods of its superclass. It may have some extra attributes or methods and it may change some of the methods of the superclass, but all operations that are defined on the superclass are "automatically" defined on the subclass as well. This terminology reflects the convention to call a data type a class. With this terminology we can say that objects are instances of a class.

The power of inheritance lies in the reuse of code: if a certain existing data type matches already most of the requirements one has, one may derive a new data type from it. The aspects of the data type that already match the requirements can then be used straightaway. The mismatching parts need to be changed or supplemented, but this may require a lot less work than designing the new type from scratch.

Consider for example a program manipulating publications. We may require that every publication has a title, an author and a year of publication. Different kinds of publications may add different information to this: an article is published in a journal or in proceedings, a book may be part of a series, etc.

To present relations between classes, the Object Model Notation proposed in (Rumbaugh et al., 1991) will be used. The example above could be represented in a diagram as shown in figure 7.1. Each class is represented by a rectangle divided into three parts. The upper part shows the name of the class in bold. The middle part lists the attributes of a class and the lower part has room for the methods of the class. In the remainder of this chapter, the middle and lower part will often be omitted. The triangle pointing up on the lines connecting


Figure 7.1: Example class hierarchy
the classes InProceedings and Proceedings with Publication indicates that InProceedings and Proceedings are derived from Publication. The line connecting InProceedings with Proceedings and terminated with a diamond at the side of InProceedings indicates that an instance of InProceedings refers to one instance of Proceedings.

Figure 7.2 finally shows relations between classes that are not present in the hierarchy of figure 7.1 but that will be needed later on in this chapter. The line connecting A with B in the left part of figure 7.2 , terminated with a diamond and a solid circle, indicates that an instance of class A refers to zero or more instances of class $B$. If the solid circle is replaced by an open circle, A refers to zero or one instance of $B$.

A class can inherit from multiple other classes. In this case it will have all the properties (attributes and methods) of all classes it derives from.


Figure 7.2: A refers to zero or more B (left) or zero or one B (right)

## Polymorphism and dynamic binding

Polymorphism means that a variable can contain values of different types. In the example above this would mean that a variable of type Publication actually contains a value of type Proceedings. The type with which the variable was introduced to the program is called its static type, the type of the value it actually contains is called its dynamic type. The dynamic type of the variable is always equal to its static type or a descendant of it.

In the example above one could think of a list of publications. Such a list would be designed to refer to zero or more instances of class Publication. In reality it would not contain instances of class Publication but Articles, Books, Proceedings, etc. Such a list is made possible through polymorphism.

If a method of an object is called, one would want that the method corresponding to the dynamic type of the object is called: the dynamic type of the object may be a descendant of the static type of the object and the descendant may have changed the method of the static type. This means that only during the execution of the program, when it is known what the dynamic type of the object is, can be decided which method should really be called, that of the static type of the object or maybe that of one of its descendants. This is called dynamic binding.

In C++ polymorphism and dynamic binding can not be applied in all cases. A programmer has to take special precautions to make sure that these concepts are available at a certain point in his program. This technicality will be ignored further, although it gives rise to a lot of design decisions that need to be made during the coding of a new class.

### 7.3 Matrices and related data structures

### 7.3.1 Vectors

Near the bottom of the class hierarchy that will be described in this chapter is the class of vectors, the vectorCls. A vector is a one-dimensional array of real
or complex numbers on which a number of operations are defined: addition of another vector, a complex scalar or a real scalar, multiplication with or division by a scalar and element-by-element multiplication or division. Moreover, one vector may be assigned to another.

Before looking into the details of this, lets assume for now that this is implemented. It is now possible to write

```
c=a+b;
```

to add the instances $a$ and $b$ of type vectorCls and store the result in another instance c of type vectorCls. This line of code is executed in the following steps.

1. Create a temporary instance of class vectorcls to hold the result of the addition.
2. Perform the addition, storing the result in the temporary object.
3. Discard the old contents of $c$ and make sure that $c$ is of the same dimension as a and b.
4. Copy the contents of the temporary instance to $c$.
5. Discard the temporary instance.

Step 4 is inconvenient: memory needs to be allocated for $c$ to receive a copy of the information in the temporary instance and then this copy has to be made. At this point in the program there exist two copies of the same data. The next step then throws away one of these copies. It would be both faster and less memory consuming if $c$ and the temporary instance would share the same data instead of making a separate copy for both.

This can be taken even a step further. After

## Code fragment 7.4

$\mathrm{a}=\mathrm{b}$;
$a$ and $b$ might as well share the same data. Only if the value contained in a or b changes the two instances need to be given their own copies: if the value of b after execution of code fragment 7.4 changes it is not intended that the value of a changes along!

This is a wide-spread idea in class design. It is called deferred copying. It is implemented here by using an extra class called vectorData. An instance of vectorData contains the actual data of a vector. A vectorcls refers to an instance of vectorData. After the assignment in code fragment 7.4 the instances a and b refer to the same instance of class vectorData. This gives the preliminary class hierarchy shown in figure 7.3. An instance of vectorCls may also refer to no instance of vectorData, which is indicated in figure 7.3


Figure 7.3: Preliminary hierarchy for the implementation of vectorCls
by the open circle. This special case is interpreted as the empty vector: the instance of vectorCls represents a vector of dimension zero.

A lot of the functionality of a vector is actually implemented by vectorData. vectorData "knows" whether its data is real or complex, can switch between these two cases, can add two instances of vectorData, etc. It is also concerned with allocating sufficient memory to store all data in the vector and releasing this memory again if the instance of vectorData ceases to exist. In principle, vectorData hides all details concerning how the vectors are stored, how a real vector can be told from a complex one and how many elements the vector has.

Now suppose a is a vector containing four elements and b is a vector containing three elements. If we want to assign the value of $b$ to the last three elements of a, we would like to be able to write:

## Code fragment 7.5

$$
\mathrm{a}[1: 3]=\mathrm{b} ;
$$

Some technical details of $\mathrm{C}++$ need to be introduced now: in C++ one indices into a vector using square braces, in Matlab this is done using parentheses. Moreover, in $\mathrm{C}++$ the first element of a vector has index 0 , where in Matlab it has index 1. Finally the notation 1:3 has been borrowed from Matlab. This notation can not be used in C++. The problem of specifying ranges of indexes is discussed later. The equivalent Matlab code for the previous example would be

$$
a(2: 4)=b ;
$$

Code fragment 7.5 is implemented by letting a[1:3] return an instance of type vectorClsRef. An instance of vectorClsRef refers to (part of) an instance of vectorCls. For now assume that an instance of class indexRange specifies what part of the vectorCls is to be used. Assigning an instance b of class vectorCls to an instance of vectorClsRef referring to an instance a of class vectorCls is now implemented by overwriting the part of a with $b$. The vectorClsRef does not know how to do that, it has to ask vectorData to do that.

Now let us take the example in code fragment 7.5 a step further. Consider the following example:

Code fragment 7.6
$a[1: 3]=a[0: 2] ;$

The intention of this line of code is that after the assignment the elements 1,2 and 3 of the vector a contain the values that the elements 0,1 and 2 had before the assignment. $a[1: 3]$ plays a different role in this example than $a[0: 2]$. a[1:3] refers to the second to fourth element of a but a[0:2] should refer to a copy of the first to third element of a. Fortunately this can be handled relatively easily. In the implementation of vectorClsRef assigning a vectorClsRef $b$ to $a$ vectorClsRef $a$ is implemented by converting $b$ first to a vectorCls and then assigning that to a.

Converting a vectorClsRef to a vectorCls could be implemented by letting the new vectorCls refer to a new vectorData that contained a copy of the subvector referred to by the vectorClsRef. In figure 7.4 this is shown schematically for the example given above. In this figure the convention to represent instances of a class by rounded rectangles and put the name of the class in parentheses is used. The upper part of the figure shows an instance of vectorClsRef referring to the second, third and fourth element of a vectorCls containing the value $[10,11,12,13]$. The lower part shows the instance of vectorCls that is made from this vectorClsRef.


Figure 7.4: Conversion of vectorClskef to vectorCls by explicit copying

It was chosen to arrange the actual implementation somewhat differently. To avoid making unnecessary copies of vectorData instances it was decided to make it possible that a vectorCls instance refers to part of a vectorData instance. What part it refers to is indicated by an optional indexRange. If no indexRange is associated to an instance of vectorCls, the vectorCls is assumed to refer to all of its associated vectorData. Figure 7.5 shows how a vectorCls can now be created from a vectorClsRef without the need to copy the vectorData instance.

It may seem that one problem has now been replaced by the other. Instead of copying the vectorData instance an instance of type indexRange needs to be copied. However, most indexRanges can be represented in a much more


Figure 7.5: Conversion of vectorClsRef to vectorCls without copying of a vectorData
compact way than general vectors of the same length. Therefore copying an indexRange is in general much faster than copying a vectorData containing the same number of elements. On the other hand, checking whether a vectorCls has an indexRange and getting the elements out of such a range one by one implies a lot of overhead. However, this overhead occurs also during the copying of a vectorCls. It depends on the actual sequence of actions in a program whether using an indexRange in the way described above speeds up the program or slows it down.

The assignment in code fragment 7.6 now consists of the following steps.

1. $\mathbf{a}[1: 3]$ is evaluated, thereby creating a vectorClisRef that refers to the second, third and fourth element of a.
2. a[0:2] is evaluated, resulting in a vectorClsRef referring to the first, second and third element of a.
3. The vectorClsRef created in step 2 is converted to a temporary vectorCls. This involves making a copy of the indexRange of the vectorClsRef created in step 2 and sharing an instance of vectorData with a.
4. The vectorClsRef created in step 1 requests permission from its vectorCls (in this case instance a) to change the vectorCls's data. The vectorCls will request this permission in turn from its vectorData and this will report that it can not grant this permission, because other instances of vectorcls are referring to this instance of vectorData, expecting that the vectorData instance remains constant. This is resolved
by creating a new copy of the vectorData that only a will refer to. The vectorData instance that $a$ is now referring to is shared by no other vectorCls instance, so a grants permission to change its data to the vectorClsRef that requested it.
5. The actual change of the data of a is carried out.
6. The vectorClsRef that had permission to change a's data informs a that it no longer needs this permission.
7. The temporary vectorCls created in step 3 is discarded. To clean things up, the temporary vectorCls first deletes its indexRange. Then it announces to its vectorData that it will no longer use it. The vectorData responds that it is now no longer used by any vectorCls instance, so that the vectorcls instance knows that it is safe to delete the vectorData instance.
8. Both vectorClsRef instances are deleted.

If the assignment to be performed had been of the form

## Code fragment 7.7

$a[1: 3]=b[0: 2] ;$
where $b$ is a vectorCls instance independent of $a$, the creation of a new vectorData in step 4 would not have occurred, provided no other vectorCls instances are referring to a's vectorData instance. Saving this copy operation was the reason to create the possibility that a vectorCls refers to part of a vectorData.

The hierarchy of vectorcls-related objects is finally given in figure 7.6.


Figure 7.6: Class hierarchy implementing vectors

Remark 7.1 Figure 7.6 is still a simplification of the true hierarchy, The exchange of information between a shared object and other objects that share it is implemented by means of two classes, one encapsulating the actions to be performed by the shared object, the other encapsulating those to be performed
by the sharing object. Therefore vectorData is derived from the former class, vectorCls from the latter.

Also vectorCls objects are able to print a representation of themselves to a file or on the screen and they know how to convert input from a file or from the keyboard to a value of type vectorcls. The interaction with files, terminals or any other form of a stream is arranged by deriving from a class streamablecls.

These and other details are not further discussed here.

### 7.3.2 Specifying sets of indices

In the previous section subvectors of a vector were considered. It was discussed how to handle subvectors assuming that it was known what part of a larger vector to consider. The task of specifying that part was left to a class called indexRange. In this section the class indexRange and its descendants are described.

The class indexRange represents in fact nothing else than a collection of integer values. Several ways to specify such a range come to mind:

- A collection containing all values at equidistant steps between given starting and ending values. In Matlab these indices are specified by the colon operator: $1: 5$ or $10:-2: 1$ are examples of such sets.
- A trivial set containing only a single integer value.
- A collection containing an explicit list of all its elements.
- A collection containing an explicit sorted list of all its elements. Sorting the elements in the list can be useful to remove duplicated elements or to check whether a value is a member of the collection.
- A combination of other collections, representing the union of the respective collections.

For every item in the previous list a different class is derived from indexRange. This gives the hierarchy of figure 7.7. In this hierarchy a sorted collection of integers is implemented as a special case of an explicit list of integers.
indexRange does not have the full functionality of a set of integers. It only specifies how to interact with such a set no matter what its actual form is. Due to polymorphism a vectorCls or vectorClsRef can think that it is dealing with an indexRange where it is in fact dealing with, say, an explicitRange. This makes it possible to define new forms of sets of integers without the need to inform vectorCls or vectorClsRef about this. As long as the new set is derived from indexRange, vectorCls and vectorClsRef will already know how to interact with it.


Figure 7.7: Hierarchy of sets of indices

An explicitRange is vaguely similar to a vectorCls. It refers to integer values similarly as vectorCls refers to real or complex values. Indeed, explicitRange is implemented like vectorCls, referring to an auxiliary class explicitData that takes care of allocating memory for the list, etc. Two instances of explicitRange can share the same instance of explicitData in the same way as two instances of vectorCls can share a vectorData.

Every descendant of indexRange is required to define an auxiliary class called an iterator. Every instance of an iterator class is associated with an instance of an indexRange descendant. The task of the iterator is to return one by one the elements of its associated indexRange. This functionality could have been implemented in indexRange and its descendants, but the scheme outlined here makes it possible to have two simultaneous iterations over a collection without the need to copy the collection. This scheme violates the principle of information hiding a bit. An iterator for a class needs to know how the class it belongs to stores its information and furthermore it needs to have access to this information. This is not considered a problem as long as the programmer implementing or maintaining these classes is aware of it. To a user of these classes it is still invisible how a certain set is implemented.

### 7.3.3 Matrices

Matrices are derived from vectorCls. They are considered an extension of vectors. The extensions involve

- specifying how many rows or columns the matrix has. The total number of elements in the matrix should be a multiple of this number.
- defining two dimensional indexing. The expression

```
m(2,1)
```

refers to the element in the third row and the second column of a matrix m . Note that here too indexing starts at index zero.

- implementing multiplication of a matrix with another matrix or with a vector.

The actual calculations of matrix addition and of all operations involving a scalar can be performed by the corresponding vectorCls operations. Nevertheless separate methods have to be defined for the matrix class, because the type of the result of these operations is no longer a vector but a matrix. The matrix functionality is present in the class matrixCls. As already mentioned, this class is derived from vectorcls.
vectorCls being the superclass of matrixCls it is clear that the entries of matrixCls are somehow stored as a long vector. Whether the row vectors are concatenated to one vector or the column vectors are stacked on top of each other is irrelevant for the use of matrixCls. A user of the class can find out if he insists without looking at the code, but it would be unwise to write any code that relies on the choice that has actually been made.

## Submatrices

Taking submatrices of a matrix is arranged in the same way as taking subvectors of a vector is. If $m$ is an instance of matrixCls and ra and $r b$ are indexRanges, the expression $m$ ( $r a, r b$ ) returns an instance of type matrixClsRef. This time parentheses instead of square brackets are used because C++ does not support the syntax $m[r a, r b]$. Using the syntax $m[r a][r b]$ would cause other problems.

There are some subtleties involved in the implementation of these aspects in matrixCls and matrixClsRef. Two indexRanges are involved in selecting a submatrix from a matrix, one referring to the row indices of the matrix, the other referring to the column indices. In the superclass vectorCls there is only the possibility to refer either to all elements of a vectorData instance or to a subset of it indicated by a single indexRange. The problem of where to store the other indexRange can not be solved by creating a new attribute for matrixClses that can contain this extra indexRange. Suppose for example that a matrix consists of three rows and three columns and that it refers internally to all elements of a vectorData. Because a matrixCls is derived from a vectorCls it is a vectorCls and can be treated as such. As the matrix contains nine elements, the vectorData instance that the vectorCls superclass refers to must also have nine elements in this case; it was assumed that all elements of the vectorData were used to represent the matrix. The matrixCls interpreted as a vectorCls will appear to be a vector with nine elements. Now
consider the same three by three matrix, but now represented as a submatrix of a larger matrix, say a four by four matrix. The vectorData referred to by the matrixCls now contains sixteen elements, only nine of which are used by the matrixCls. If one of the indexRanges, say the one selecting three out of four possible rows, is stored in the vectorCls and the other indexRange selecting the column indices is stored in the matrixCls, the vectorCls will "think" that it refers to only three of the elements of the vectorData instead of nine, because the extensions to vectorCls defined by matrixCls are invisible to vectorCls. Not only is this inconsistency confusing for the user, it also makes that all element by element operations that the vectorcls superclass is supposed to perform, such as multiplication by a scalar, operate only on a fraction of the data on which they should operate. To make things even worse, the reader may verify that the fraction of the vectorData that the vectorCls is referring to may not even be part of the submatrix that is actually intended.

The solution of this problem is to derive a class from indexRange that can store two indexRanges. This functionality is present in the twoDRange class. The twoDRange needs to know whether a matrixCls stores its data as a concatenation of rows or as a concatenation of columns. In the first case it also needs to know how many columns the matrix has it is supposed to select a submatrix from. In the second case the number of rows needs to be known. The twoDRange can now calculate which elements of the vectorData instance a matrixCls instance is referring to and present itself as a collection of indices corresponding exactly to those elements.

The difference between a matrixClsRef and a vectorClsRef is that a matrixClsRef always refers to a matrixCls, never to its superclass vectorCls. A vectorClsRef refers to a vectorCls, which may or may not actually be a matrixCls. Also a matrixClsRef refers to a collection of indices which is required to be of type twoDRange. A vectorClsRef refers to a collection of indices that may be represented by any descendant of indexRange.

Figure 7.8 shows the extension of the hierarchy described in this section.

## Matrix functions and decompositions

A matrix class is not very useful if only multiplications and additions are defined. What is really needed is a set of functions to calculate a matrix's eigenvalues and eigenvectors, its Schur decomposition, its singular value decomposition, its QR decomposition, its LU decomposition and its inverse. For a description of these decompositions and ways to calculate them numerically, see (Golub and Van Loan, 1989).

Writing numerically sound routines solving these problems is not a trivial task. Moreover, even merely implementing the routines from a textbook like (Golub and Van Loan, 1989) would be a time consuming enterprise. Fortunately, the Lapack library provides a collection of routines that performs all


Figure 7.8: Hierarchy of objects implementing matrices
these tasks. This library is publicly available on the internet and is documented in (Anderson et al., 1995). All of the problems mentioned above are relatively easily solved by calling routines from Lapack.

The routines in LAPACK expect that matrices are stored in a well-defined way. indexRanges are unknown to LAPACK. Therefore if a LAPACK routine expects a matrix, it can not handle a matrix whose entries are a subset of a vectorData accessed through a vectorCls. It can only handle an array of values containing one column of that matrix after the other, together with two integers specifying how many rows and columns the matrix has. Separate routines are provided for real matrices and complex matrices. It is the task of matrixCls's methods to make sure that the appropriate LAPACK routine is called and that Lapack finds the matrices in the way it expects them. Also any temporary working space required by LAPACK is allocated by the matrixCls methods, so that all these technical details are invisible to the user.

The storage of vectors in vectorData has been arranged such that real numbers and complex numbers are stored in a way that is compatible with Lapack. Also an instance of matrixCls stores its elements one column after the other. Therefore the storage of entries in vectorData is exactly what LAPACK expects, except for a possible indexRange in vectorCls that selects a subset of that vectorData. The matrix entries do not need to be rearranged to call Lapack routines. Strictly speaking, this violates the concept of information hiding, matrixCls uses information on how vectorData stores its information. The alternative would be not to use this information and to make a copy of every matrix such that the copy is stored in the way Lapack expects it, regardless of how the original matrix was stored. This has large repercussions in terms of memory usage and computing time. This was considered an unreasonably high price to pay for information hiding.

Wherever it makes sense, routines are implemented in two ways: one in which the result of an operation overwrites the arguments to the operation and one in which the original arguments are preserved. For example

$$
\text { a.inv }() ;
$$

overwrites the matrix a with its inverse, assuming a is a square, non-singular matrixCls, and

```
inv(a);
```

is an expression returning the inverse of a, leaving the original value of a intact. Using the first approach saves memory. Because the LaPACK routine mostly use this approach as well, it also saves an extra copy operation, so it also saves time. Nevertheless the second approach is easier in some circumstances and is therefore provided as well.

## Special forms of matrices

Apart from general matrices LAPACK also supports triangular matrices, band matrices, symmetric and hermitian matrices, positive matrices and some special orthogonal matrices. Exploiting the structure that a matrix is known to have can result in lower demands on memory. Special "packed" representations of triangular matrices, band matrices and symmetric matrices exist in Lapack that remove the redundancy that otherwise would be present in the "full" representation of the matrix. For example, instead of mapping the square, upper triangular matrix

$$
A=\left(\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{q 1} \\
0 & a_{22} & & a_{q 2} \\
\vdots & \ddots & \ddots & \vdots \\
0 & \ldots & 0 & a_{q q}
\end{array}\right)
$$

to the vector representation

$$
\left[a_{11}, 0, \ldots, 0, a_{12}, a_{22}, 0, \ldots, a_{q q}\right]
$$

one could map it to the more compact representation

$$
\left[a_{11}, a_{12}, a_{22}, \ldots, a_{q q}\right]
$$

Extension to the lower triangular and the non-square case is straightforward. The same storage scheme that can be applied for triangular matrices can be used for symmetric and hermitian matrices. As an alternative, one may use the non-packed storage scheme that matrixCls uses and simply ignore the values that would be omitted from the packed scheme. In this case there is no gain
in terms of memory, but execution can be faster, because the implementation need not be concerned any more with keeping the entries up to date that are ignored any way.

Another advantage of using structure of a matrix is that some problems can be solved more easily if the matrix has a certain structure. For example, the eigenvalue problem is easier to solve numerically for symmetric matrices than for non-symmetric matrices. Also, taking the inverse of an orthogonal matrix is an especially easy operation; taking its transpose suffices.

Due to time constraints, these structured matrices have not been implemented, but below will be indicated how such an implementation could proceed.

The first thought that comes to mind considering how to implement these special matrix forms is to make these matrices a descendant from matrixCls. After all, they are matrices that have, compared to general matrices, some extra properties. From an implementation point of view, this is not such a good idea. For example, in a general matrix, one can assign an arbitrary value to any of its entries. If, say, a symmetric matrix is implemented as a descendant of matrixCls, this operation also exists for the derived (symmetric) class. Assigning a value to a single entry of a symmetric matrix is likely to destroy the symmetry of the matrix and should not be allowed. As another example, a symmetric matrix that is stored in non-packed form uses only either the upper triangular or lower triangular part of the general matrix that is used to store it. The entries below resp. above the diagonal are simply ignored and contain random data. If a general multiplication is performed with such a matrix, these entries will no longer be ignored and will give an erroneous result.

Of course these deficiencies can be fixed by adapting the appropriate methods in the derived classes such that these problems are avoided. Provided dynamic binding is used, this seems a solution to the problem in the true object-oriented spirit. There are two points of criticism one may raise:

- In C++ a class designer has the option whether or not to use dynamic binding. Dynamic binding implies some overhead in the execution of the code that is not present in static binding. A class designer may therefore tend not to use dynamic binding, provided he is not cutting off any valuable possibilities for himself.
- A class should be derived from the superclass with which it has most in common. Some of the methods of matrixcls would need to be changed to reap the performance benefit that can be obtained by exploiting the structure in the matrix or to avoid destroying the structure of the matrix. If it is analysed which methods need to be changed, it turns out that these special matrices might as well be derived from vectorCls.

Because of this it is proposed to derive these matrices from vectorCls instead of matrixCls.

If the calculation of the result of an operation can not benefit from the structure in the matrix, the easiest thing to do is to omit implementation of that operation for the structured matrix but implement instead a method that converts the structured representation of the matrix to an equivalent matrix in general form. If the $\mathrm{C}++$ compiler encounters an operation that it does not know how to perform for, say, a triangular matrix but that it is able to perform for a matrixCls, it will convert the triangular matrix automatically (under certain conditions) to a copy of the matrix in matrixCls representation and use the matrixCls in the operation instead. If the run-time overhead implied by this conversion is not desired, one can decide to opt for the less easy possibility of implementing the operation explicitly for this matrix structure.

Having defined such a conversion method, a class representing a structured matrix still needs to do two things: change those methods of vectorCls that need special attention for structured matrices and add methods for matrix operations that preserve or exploit the structure. For each type of structure, one has to decide what this general observation means in practice.

### 7.4 Building on vectors and matrices

### 7.4.1 Polynomials

Scalar polynomials can be relatively easily derived from vectors. A vector is used to contain the coefficients of the polynomial. Addition or subtraction of two polynomials involves padding the shortest of the two polynomials with zeroes to the same length as the longest polynomial. Then a simple vector addition/subtraction will give the coefficients of the sum/difference of the two polynomials.

Other methods that one wants to have for a polynomial are multiplication by a real or complex scalar or another polynomial. Also evaluation of the polynomial using a real or complex scalar as its argument is supported. Evaluation of the polynomial with a square matrix argument is also straightforward to implement. The problem of finding the zeros of a polynomial can be translated to an eigenvalue problem. Class matrixCls can then be used to solve this problem. All this functionality is available in the class polynomialCls.

These methods make it possible to write for example

```
m1 = (p1*p2+p3)(m2);
```

for $\mathrm{p} 1, \mathrm{p} 2$ and p 3 of type polynomialCls and m 1 and m 2 of type matrixCls. This line of code means: calculate the polynomial that is the sum of p3 and of the product of p 1 and p 2 . Then evaluate this polynomial for m 2 and store the result in m 1 .

### 7.4.2 Signals

Discrete time signals defined on a finite interval are very similar to matrices. Multiplication by a scalar, addition or subtraction of a scalar and addition or subtraction of two signals all require the same calculations as the corresponding operations for matrices. Therefore the class signalCls that implements signals uses a matrixCls internally to represent signals.

Despite this similarity between signals and matrices, signals and matrices can also be very different. For matrices the norm used most often is the matrix norm, the largest singular value of the matrix. This norm is rather meaningless for signals. Also multiplication of two matrices is defined, multiplication of two signals is not. Taking the autocorrelation of a signal has a clear interpretation, for a matrix this interpretation is unclear, unless the matrix is interpreted as a discrete time signal any way. For this reason signalCls is not derived from matrixCls. It just has an attribute of type matrixCls.

Whether a signal needs to know at what rate it was sampled and at what time instant sampling started is a point of discussion. It seems that important information regarding the signal is missing if these attributes are not part of the signalCls specification. On the other hand, experience with the implementation of the algorithm presented in this thesis is that this information is never used. Keeping these attributes up to date amounts in this case to extra overhead from which no real benefit can be derived.

Practice has to show whether there is really a need to record sampling frequency and starting time of a signal. As an intermediate solution, this functionality is present as an option.

### 7.4.3 Linear, time-invariant processes

Software intended to support system identification is useless if it does not support models of some form. In this implementation models for linear, timeinvariant processes are supported that can be interpreted as input/output operators. As in chapter 3, some generality is lost by distinguishing between inputs and outputs in the set of signals for which a model is derived. The capability of the software to avoid making this distinction would require a large programming effort to implement. As the algorithm of chapter 4 distinguishes between inputs and outputs, this capability would not be used. This applies also to many other algorithms. Therefore it was not considered a problem that models make this distinction.

Having decided that the models to be used represent linear time-invariant processes, there are a number of things that every type of model should be capable of:

- A model should be able to filter a signal, mapping an input signal to an output signal.


Figure 7.9: Constructing models from other models. Left: addition, middle: cascading, right: appending

Suppose that the input signal is cut into two signals, representing the first and the second half of the experiment. If these two inputs are filtered by the model and the two outputs are concatenated, the result should be the same as when the input signal was filtered in one operation. This implies that every model should have an attribute representing its state or its initial condition. This attribute will be called state further on. This does not imply that only state-space models are admissible.

- A model should be able to store its state in a variable and restore it from that variable. Resetting the state to zero initial conditions should also be possible.
- The response to a number of special input signals should be generated without creating that input signal explicitly. In particular, the impulse response and step response for zero initial conditions should be accessible. Moreover, the response to an input equal to zero for the given state of the model has to be made accessible. All these operations should update the state of the model, so that the various responses can be extended without starting from scratch.
- The frequency response of the model for a certain frequency or a number of frequencies needs to be available.
- New models can be constructed from existing models. This includes addition of two models, cascading two models and appending two models, see figure 7.9. Also selecting a subsystem consisting of a subset of the original system's inputs and outputs needs to be supported.

All of the functionality mentioned above is encapsulated by the class LTItransferCls. (Maybe the name LTImodelCls would have been more appropriate.) The LTItransferCls is, like indexRange, an abstract class. It does not implement the functionality itself, it just specifies how its descendants should make this functionality accessible to their outside world. The class stateCls represents the state of a model. For every descendant of LTItransferCls there
should be a corresponding descendant of stateCls to hold the state information of that type of model.

Polymorphism and dynamic binding is again used to increase the flexibility of the classes: if a user writes a routine to make a Bode plot for class LTItransferCls, the routine will function with any descendant of LTItransferCls too. No separate routines for different model representations are used any more.


Figure 7.10: LTItransferCls hierarchy

Two descendants of LTItransferCls have been implemented: SStransferCls and MPItransferCls, see figure 7.10. SStransferCls represents a state-space realisation of linear, time-invariant models:

$$
G(z)=C(z I-A)^{-1} B+D
$$

The model class MPItransferCls is derived from the Minimal Polynomial Identification toolbox, a toolbox used in our group written by H. Falkus, based on (Falkus, 1994, chapter 2). The relation between inputs and outputs used by this model is

$$
\begin{equation*}
a(z) y(k)=\frac{B(z)}{f(z)} u(k)+\frac{C(z)}{d(z)} \epsilon(k) \tag{7.1}
\end{equation*}
$$

$\epsilon(k)$ is a noise sequence, $u(k)$ is the process input and $y(k)$ is the process output. $a(z), f(z)$ and $d(z)$ are scalar polynomials, $B(z)$ and $C(z)$ are matrix polynomials of appropriate size. As far as the MPItransferCls is concerned, both $u$ and $\epsilon$ are considered inputs to the model. The inputs are just partitioned into two parts, one filtered by $\frac{B(z)}{f(z)}$, the other by $\frac{C(z)}{d(z)}$.

The popular FIR, ARX, output error and Box-Jenkins models are all a special case of the model structure (7.1). These model structures can therefore be implemented by deriving them from MPItransferCls. For different structures, different polynomials in (7.1) need to be fixed to one or the identity, depending on whether it is a scalar polynomial or a matrix polynomial. The task of
the derived classes is now to restrict the capabilities of the superclass, not to expand it. In this case this is particularly easy to do.

In order to keep the structure flexible, one can not require that every descendant of LTItransferCls "knows" how to convert to any other descendant of LTItransferCls. Nevertheless it is desirable to be able to convert from one representation of a linear, time-invariant model to another. The solution that has been adopted here is that every descendant of LTItransferCls must be able to convert to and from a state-space representation. This excludes the possibility of handling non-proper models, which can not be represented in state-space form but which are in principle possible in other model structures. This is not considered too big a problem, because non-proper models correspond in discrete time to non-causal models. It is assumed that the partitioning of signals into inputs and outputs has been carried out correctly. This implies that there exists a causal relationship between inputs and outputs.

How should one descendant of LTItransferCls be added to another? If a model g2 needs to be added to a model g1, g1 may check whether g2 is the same type of model as itself and perform the addition without further conversion if it is. This implies that every model structure should be able to add two instances of that structure together, which seems reasonable to require. If g2 is not the same structure as g1 is, g2 should be converted to state-space. One can then choose whether to implement a routine that adds the state-space representation of g 2 to g 1 or to use the conversion from state-space representation to the model representation used by g1 and add the converted model to g1.

This approach is still not a full solution to the problem. An ARX model can not be added to an output error model with an output error model as a result. One may use polymorphism in this case and decide to use a minimal polynomial representation for the result. This involves a lot of C++ technicalities, but in principle it can be done. The choice which representation to use for the sum of two systems is then made during the class design. If new types of model structure are added to the class hierarchy, these choices may not be optimal any more. It is probably better not to pick a conversion in the class design but to generate an error if addition of two incompatible model representations is requested. This forces the programmer to be more explicit concerning his intentions with respect to the sum of the model. If the programmer can not say what model structures he will encounter in a routine, he can decide to use a state-space representation. The class design already imposed the restriction that every model should be convertible to a state-space representation, so he can be sure not to run into trouble if he uses a state-space representation.

### 7.4.4 Generalised bases and generalised FIR models

In appendix B it is shown that a generalised basis for $h_{2}$ can be derived from a basis generating system. In order to represent a basis, one could therefore
suffice with a class storing the basis generating system and routines to extract the required basis functions from this system. According to appendix B, extracting this information involves an input balanced system derived from the basis generating system and an all-pass system, also derived from the basis generating system. To avoid repeated balancing and generating of an all-pass system, a basis for $h_{2}$ is represented by a pair of SISO transfers, the input balanced transfer and the all-pass transfer. The two SISO transfers are stored in two instances of type SStransferCls. This is done by the class SISOpair.

This class "knows" how to calculate the impulse response of any of the basis functions in the set of basis functions it represents. It can also perform all other operations that were defined above for a LTItransferCls, either for a single basis function or for a combination thereof. A difference between an infinite set of basis functions and models is that it is unknown how many basis functions out of the infinite set of basis functions are actually used. A consequence of this is that the operations constructing models from a combination of other models are not implemented for sets of basis functions. Another consequence is that it is impossible to store the states of all basis functions in the basis. Therefore the states of the basis functions are stored outside the class SISDpair. Consequently a filtering operation requires not only a signal to filter, but also a variable containing the states of the basis functions at the beginning of the filtering operation and another variable that will hold the states after the operation. If the last variable is omitted from the request to filter a signal, the states at the end of the filtering operation are simply discarded. If the initial states are also omitted, zero initial conditions are assumed.

A MIMO basis is simply obtained as an array of SISO bases. Most C++ implementations provide ways to create with very little effort dynamically sized arrays containing arbitrary types. This is also required by the draft ANSI C++ standard. Maintaining the array of SISO bases is therefore easily realised. This task is carried out by the class ortFirSet. ortFirSet is not an array of SISOpair instances but of SISOpairCopy instances. A SISOpairCopy instance can share a SISOpair instance with other SISOpairCopy instances in the same way as a vectorCls instance can share a vectorData instance with other vectorCls instances. This reduces the memory requirements of ortFirSet if the same SISO basis is used for all input/output pairs in a MIMO basis.
ortFirSet is derived from MIMOtransferSet. The MIMOtransferSet specifies the interface of ortFirSet with the outside world. This makes it possible to derive other classes from MIMOtransferSet representing other ways to generate a MIMO basis. Programs using ortFirSet without using any of the specific properties of system based orthonormal basis functions will then be able to handle other types of bases without further modifications.

Having a class that represents a MIMO basis, implementing a generalised FIR model class requires the following actions:

- Provide an array of integers specifying the number of basis functions to
use in each SISO subtransfer.
- Store the states of the basis functions incorporated in the model somewhere.
- Provide sufficient "glue" between these component and the MIMO basis, so that the resulting model class is accessible through the LTItransferCls interface.

The last step suggests that the resulting model class shall be a descendant of LTItransferCls. This is indeed the case. The ortFirModelCls is a descendant of LTItransferCls and implements generalised FIR models using the components discussed above.

The hierarchy between the different classes discussed in this section is as shown in figure 7.11. The classes representing the states of the ortFirModelCls and the number of basis functions for each input/output pair are not shown for clarity.


Figure 7.11: Classes involved in the implementation of ortFirModelCls

It is now interesting to consider the data that is contained in an instance of ortFirModelCls. For every input/output pair, there is an integer specifying the number of basis functions to use and a pair of state-space models defining a SISO basis. Every state-space model is represented by four matrices. The state dimension of these state-space models need not be the same for different input/output pairs. It was the frightening idea of having to implement this in Matlab that led to the decision to implement the algorithm in $\mathrm{C}++$ and looking back on the whole project, it still seems the most important reason to switch to C++.

### 7.5 Keeping all data of an identification session

While implementing the algorithm the question rose how to make it possible to add different identification algorithms for the identification of the auxiliary models to the program, without requiring changing any of the other code. As before, polymorphism seemed an obvious candidate to tackle this problem. A general "identification procedure class" could be defined and every identification procedure derives from this class. Indeed, a class idProcCls having exactly this purpose was defined.

Unfortunately this can only be a partial solution. Different identification procedures require different prior knowledge. One can not pass all this prior knowledge explicitly to the methods of the identification procedure class: as soon as an identification procedure emerges that requires prior knowledge that was not anticipated on, the idProcCls needs to be changed. For a specific identification method it also means that it is offered a lot of data, a considerable part of which it is probably not using.

This was solved by creating a tree structure of different pieces of information, see figure 7.12. The root of the tree represents the complete identification project. Below this root, information is organised in different subtrees, representing the main categories of information. "Data" represents (initially) the experimental data and prior knowledge of the process, "Identification" specifies which identification method to use together with possible extra information the identification method needs to have, such as model order or signal scaling. Under "Data" information is organised according to the operating point it belongs to. To each operating point belongs one or more data sets and maybe other information. For each data set, there must be input and output measurements. Additionally there may be other information that is specific to this data set, etc.

Every node in this tree is an instance of (a descendant of) class propertyCls. propertyCls handles the connection with other nodes. It also contains the name of the node. If this is all the information that a node requires, an instance of propertyCls suffices to represent this node. For a node representing a data set, the node should also contain the input and output signals. A node representing a transfer should contain some representation of that transfer, etc. The ability to contain this information is obtained by deriving properly extended descendants from propertyCls.

Helper classes are provided to iterate over all or part of the nodes in the tree, either starting from the root or from any other node. Functions searching for certain nodes are also implemented, as well as an extensible set of classes that can express conditions that a node must satisfy before being considered as a candidate for a search.

Different parts of the algorithm put their results at appropriate places in this tree. The identification of auxiliary models should put the model it finds


Figure 7.12: Tree of all information pertaining to an identification session
for an operating point in the subtree corresponding to that operating point. Information required by the identification should also be found somewhere in the tree. It is the user's responsibility to make sure that all information required by the identification method he requests is available either beforehand or is made available in the steps preceding the identification.

This structure has proved flexible enough to meet all the needs of the implementation of the algorithm.

### 7.6 User interface

The user interface for the algorithm has deliberately been kept very rudimentary. There were three reasons for this.

- It is generally considered good practice to separate computational aspects in a software design from user interface issues. Therefore no user interface aspects have been built into the class design, except perhaps for the methods to read from and write to files or the terminal.
- Software is called stable if it does not (or hardly) change any more in the future. Because this prototype of the algorithm is not stable software, it was considered unwise to go through a lot of effort to provide an easy to use interface for it:
- A lot of class libraries exist that facilitate the development of graphical user interfaces on different platforms. Some of them provide a unified framework for different platforms: all details concerning the windowing
system under which a user interface is running are hidden by the class library. The same source code can be used on all supported platforms. Unfortunately these libraries are more expensive if more platforms are supported.

As it is at this point not clear on which platforms the algorithm will be running in the future, it was decided not to make a decision which platforms to support. In this way it is avoided that an expensive library is purchased that supports a lot of platforms that will never be used. It is also avoided that a cheap library turns out not to support a platform that will turn out to be attractive in the future.

Basic terminal and file I/O routines are used to communicate with the user. This was never felt to be a severe restriction in the use of the software.

### 7.7 Performance

It is difficult to assess the performance of the class hierarchy that has been introduced only superficially in this chapter. Making statements about performance amounts to making statements about the quality of software. It is a large subject of dispute how to measure the quality of software. A discussion of the more general aspects of the quality of the class hierarchy discussed in this chapter would largely mean a repetition of section 7.1 .2 . The reader is therefore referred to that section for a consideration of these aspects.

There is at least one aspect of the quality of the software that can be measured objectively. This is the speed with which the code executes.

It depends highly on the kind of code that is executed what a comparison between Matlab and the class library described in this chapter will look like. Three different situations will be distinguished.

- If a routine is mainly concerned with manipulation of matrices, without spending much time on decompositions and the like, Matlab turns out to be more than twice as fast.
This is due to the flexibility of the class library. Every access of a vector or a matrix requires some communication between a vectorData and a vectorCls. Moreover, if subvectors and submatrices are used, there is a lot of overhead in the communication with the indexRange descendants and their iterators. Some optimisation of the code is still possible to reduce the current amount of overhead, but even after optimisation significantly more overhead will still be involved compared to the Matlab code.
- If a routine is mainly concerned with decomposing a matrix, calculating an inverse, etc. the speed comparison comes down to a comparison
between the Matlab routines and the Lapack routines. Both routines have common ancestors: the eispack and linpack libraries. Nevertheless the Matlab routines turn out to be faster than the LAPACK routines, at least on a MS-DOS machine running Windows 3.11.
The Mathworks must have invested a lot of effort in optimising their routines. It is not surprising that their code outperforms the LaPaCK library in this respect.
- Pieces of code that do not rely heavily on matrix manipulations or on functions decomposing a matrix tend to be relatively slow in Matlab. Therefore Matlab is unsuited for heavy numerical optimisations. In these cases the benefits of compiled code make $\mathrm{C}++$ outperform Matlab. Speed gains of a factor three or more are not unusual.

It is clear that the advantages of the classes described in this chapter should not be sought in the speed of the executing code. The advantages lie more in the speed and the relative ease with which one can develop this code.

If speed of execution would have been of the utmost importance in the implementation of this library, a number things should have been arranged differently. For example, the ability of a vectorCls to refer to part of a vectorData instance implies extra overhead in some circumstances. It is not quite clear whether the savings in execution time that can be obtained by the current scheme outweigh the extra computing time spent due to this overhead. In fact, using a vectorData instance as an intermediate between the vectorCls instance and its data may not be such a good idea as far as minimisation of computing time is concerned. There are other ways to prevent the unnecessary copying of data during the evaluation of an expression.

Also one may choose to implement only equivalences of steppedRange and explicitRange and implement all operations on sub-vectors and sub-matrices in the respective vectorCls or matrixCls.

Every decision above that is made for the sake of speed of execution sacrifices part of the modularity of the current design. This means that the structure that will result from these decision is less flexible than the current structure. It will become more difficult to maintain and the initial coding will probably take longer too.

As far as the algorithm in chapters 4 and 5 is concerned, these considerations are irrelevant as far as computing time is concerned. This algorithm spends by far most of its computing time in solving the many linear programming problems that are part of the algorithm. The only way to speed up the algorithm, apart from using a faster computer, is to use a faster LP-solver.

## 8

# Conclusions and recommendations 

### 8.1 Conclusions

### 8.2 Recommendations

### 8.1 Conclusions

The first goal of this thesis was formulated in chapter 1 as:
Provide a fundamental analysis of the factors that lead to model uncertainty in the context of black-box identification for robust control design. Moreover, analyse the steps that are involved in deriving error bounds from experimental data and prior knowledge. Provide a justification for the choices that are generally made in the design of such a technique.

Chapter 2 considered which factors lead to model uncertainty. These factors were divided into three categories. (a) The model is only a simplified representation of the true process. Often only linear, time-invariant approximations to the true process are considered that, moreover, have only a low order. (b) Experimental data from which the model is derived is incomplete and inaccurate. The duration of experiments is only finite, as is the sampling frequency with which measurements are taken. Moreover, data is corrupted by noise and disturbances. (c) Model uncertainty manifests itself in both shape and magnitude. The uncertainty bounds contain in general processes that can be shown, based on experimental and prior knowledge, not to be the true process. This is called conservatism. An important reason for this is that the shape of the uncertainty bound does not correspond to the true shape of the uncertainty.

A technique for model uncertainty bounding should try to reduce conservatism as far as possible. However, compromises have to be made concerning computability and applicability for robust control design. It should further
be possible to interpret the bounds in physical terms and to tell what determined the eventual error bounds: the experimental data, prior knowledge, or a combination thereof.

A framework in which model uncertainty bounding can be discussed without restricting attention to a particular algorithm was presented in chapter 3 . In this framework it is made explicit that model uncertainty can not be bounded by experimental data alone. Prior knowledge/assumptions of some form are always required. In a sense, model error bounds are therefore always determined directly or indirectly by the prior assumptions.

In general, model error bounds alone can not account for all aspects of the process behaviour. Part of the process behaviour should be represented by a "noise" bound. This noise bound does not only represent noise and disturbances, but also relations between process inputs and outputs that can not be described by the model uncertainty bounds. For example, if the model uncertainty bounds contain only linear models, as is normally assumed in model uncertainty bounding algorithms, non-linear relations that exist between inputs and outputs can not be represented by the uncertainty bound and thus have to be considered disturbances.

Given the fact that both model uncertainty bounds and noise bounds describe the process, the uncertainty bounding procedure should search for the largest model uncertainty that is compatible with the noise and the prior knowledge. This worst-case approach is commonly used in uncertainty bounding, but the reason for this is not that the noise now has been "absorbed" in the model uncertainty. The true reason is that this is the only way to stand a chance that the noise that is faced by the eventual controller does not exceed the noise bound that was used during the identification.

The previous results were formulated in a deterministic fashion, but they can be reformulated in a probabilistic way too. For simplicity this has not been further developed.

Given these results, it can be concluded that the first goal of this thesis has been achieved.

The second goal of this thesis reads (see chapter 1):

> Based on the results of the aforementioned analysis, adapt existing techniques for model error bounding for application in robust control design or develop new ones such that they fit better to the requirements of robust control design in general and are less conservative in particular.

The basic idea behind the algorithm presented in chapter 4 and 5 is the hypothesis that in practice only a limited number of physical causes induce shifts in process dynamics. Because this number is limited, there must be a considerable amount of structure in the changes in the process dynamics they generate. This means that the changes that occur in a process transfer for
one frequency should be related to those that occur for another. In the same way, changes that occur in the transfer from one input to one output should be related to those occurring in transfers for other input/output pairs.

This has led to the formulation of an algorithm in which both structured and unstructured errors are distinguished. The structured errors account in a detailed way for the effects that were mentioned above, the unstructured errors account for all other effects in a much less detailed way. The algorithm can estimate how the separation between structured and unstructured parts should take place.

This algorithm has, potentially, two advantages: first, the large contributions to the error are bounded in a detailed way. The remaining parts are bounded much less accurately but they can be much smaller. Second, the assumption that the process is linear, time-invariant is relaxed further than in other techniques: the new algorithm recognises explicitly that the process dynamics change in different operating points.

Clearly, the algorithm is in line with the spirit of the second goal of this thesis. This was enhanced further by the extensions presented in chapter 5. By means of these extensions, more prior knowledge can be used in the uncertainty bounding: by estimating the transients in the data, more accurate bounds on their influence can be given. Prior knowledge concerning the DC gain of the process or concerning the complex gain for arbitrary frequencies can be used. Solutions are provided to tune the separation into structured and unstructured parts. Alternatively it is also shown how the user may overrule the algorithm completely in this respect and specify this separation explicitly.

The algorithm relies heavily on the use of models containing a finite number of system-based orthonormal basis functions. Chapter 5 also shows how the estimation of such models can be improved in two ways. It is shown that it is relatively easy to estimate transients for these models. Further a regularisation procedure is proposed for model sets containing models whose impulse response is considerably longer than the experiment intervals. Contrary to other regularisation schemes, this regularisation has a very clear physical interpretation and justification.

The case studies in chapter 6 show that the algorithm is indeed capable of reducing the unstructured error by splitting off structured error components. Simulation results suggest that this even applies in cases where the structure in the model uncertainty can not be represented exactly by the algorithm. The algorithm then finds an approximation to the true structure in the model uncertainty that it can represent and bounding this separately still reduces the unstructured error significantly.

Application of the algorithm to practical data obtained from an asynchronous machine showed that there are two requirements that need to be fulfilled before application of the algorithm pays off: It seems that the signal to noise ratio needs to be at least 40 dB and the poles of the system on which
the system based orthonormal basis function are based need to be sufficiently close to those of the true process.

If the latter requirement is not fulfilled, the bounds on undermodelling will dominate the total model uncertainty. The relatively strict requirements on the noise, especially in a practical setting, have two causes. Deterministic assumptions are made on the noise. This worst-case approach allows much more noise sequences than seems realistic. Moreover, any relation that exists between different structured error components is lost in the algorithm, because the structured errors are bounded separately. This also applies to the relation between unstructured errors in different input/output channels, for different frequencies in case of frequency domain bounds on the unstructured error and for different sample instances in case of time domain bounds. See also section 8.2.2.

Apart from these problems there is another issue that limits practical application. The computational load of the algorithm increases rapidly with the number of inputs and outputs of the system and with the length of the data sets. Application of the algorithm for systems with more than two inputs and two outputs will easily result in computing times in the order of weeks or worse, making application of the algorithm impractical.

The algorithm was implemented in C++ because MATLAB, the obvious alternative, seemed to provide insufficient support for this. A generally applicable library for computer aided system identification was developed. Use of this library makes it possible to code an algorithm at a level of abstraction that is much higher than that of matrices and vectors. This makes the code faster to develop and easier to maintain. The advantages of $\mathrm{C}++$ over Matlab become more important for large programming projects. For small programming projects, the advantages of MATLAB over C++ are clear.

Overlooking the results that were obtained, it seems that a number of important steps towards realisation of the second goal of this thesis have been made. Given the practical problems that the proposed algorithm still exhibits the second goal can not be considered completely achieved.

### 8.2 Recommendations

### 8.2.1 General remarks concerning the algorithm

Some recommendations, including a potential solution to the problems indicated in the previous section, are given below, but some more general remarks seem in order first. The argumentation for the expectation that splitting off structured error components reduces the remaining unstructured error lies in the physical causes of model uncertainty. In the categorisation of model uncertainty sources that was made above, these causes fall in category (a). The explanation for the large uncertainty bounds that were obtained for the asyn-
chronous motor was found to be a large noise bound, corresponding to an effect of category (b), and the fact that some knowledge about the uncertainty bound was not used. This is by definition called conservatism, so it falls in category (c).

It seems that the wrong effects determine the error bounds. These effects are considered the wrong effects, not because they are irrelevant for robust control design, but because they represent our inability to give tight bounds on the true process behaviour, even if it were linear, time-invariant. The fact that different linear representations for the true process are appropriate under different conditions, which is in my view what robust control is all about, is completely overshadowed by that inability.

The solution to these effects can probably only be found by using more or more accurate knowledge in procedures for bounding model uncertainty. As much prior knowledge as possible should be incorporated in the uncertainty bounds. This requires further improving uncertainty bounding algorithms to make the use of this information possible.

It was mentioned earlier that information was lost in the algorithm concerning the relation between the structured error components and between the structured errors and the unstructured errors. Also the relation between the unstructured errors for different input/output channels, for different frequencies or for different sample instances is not represented by the uncertainty bounds. It is not quite clear what should be considered the true origin of this loss of information. Certainly, designers of model uncertainty bounding techniques should continue to seek the fault with themselves and try to avoid such loss of information in their algorithms as far as possible. Nevertheless one may start wondering to what extent conservatism is induced by the format in which robust control wants to get its uncertainty bounds. The system identification community has spent some years catching up with the robust control designers. Maybe it is time now to continue together and to increase the awareness in the robust control design community of the kind of information that can be assumed to be available in practice.

The algorithm presented in chapter 4 may serve as an example for the latter remark. In this algorithm, complex polytopes in the Nyquist plane were outerbounded by polytopes having only a small number of vertices. Of these simple polytopes, the vertex having the largest distance to a nominal model was then used to bound the absolute value of the model error for that frequency. This effectively outerbounds these polytopes by a circumscribed circle with its center "in the nominal model." Of these circles, the one with the largest (weighted) radius is finally selected to bound the $\mathcal{H}_{\infty}$-norm of the model uncertainty, where care has been taken to take into account the effects that occur between the frequencies for which such circles have been calculated. In each of the steps, a more compact but less accurate representation of the available data is derived, yielding finally a representation that is suitable for robust con-
trol design. If robust control design develops such, that the results that are now intermediate results can be used directly as a characterisation of model uncertainty, conservatism will be reduced.

As another example one may further think of using probabilistic bounds on the uncertainty, derived from probabilistic assumptions on the noise, instead of the hard, deterministic bounds that are now used by robust control design.

### 8.2.2 Stochastic assumptions on noise

The algorithm was formulated in chapter 4 using a deterministic framework for the noise. These deterministic bounds on the noise were then mapped to deterministic bounds on the model parameters and these were finally translated to deterministic bounds on the structured and unstructured error components.

This is not essential to the algorithm. From stochastic assumptions on noise the variance of model parameters can be derived using Cramer-Rao bounds. The derivation of Cramer-Rao bounds relies on the fact that the process can be represented exactly by a model in the model set, which is a serious drawback of Cramer-Rao bounds in this context. Fortunately it is shown in (Falkus, 1994) how stochastic bounds on the model parameters can be estimated even if the process can not be represented exactly by a model in the model set. This still results in an estimate for the covariance matrix of the model parameters.

Another option to come to an estimate of the model parameter covariance matrix can be derived from the results in Hakvoort (1994, section 5.4). In their present form, these results do not estimate the covariance matrix of the model parameters, but only minor modifications are required to achieve this.

Based on an asymptotic normality assumption, the covariance matrix of the model parameters can be translated to ellipsoidal confidence regions for these parameters. These confidence regions can be outerbounded by an orthotope, representing a number of constraints equal to two times the number of parameters. If these and only these constraints are used in the algorithm, the number of constraints that needs to be handled in the subsequent linear programming algorithms reduces enormously. This may provide a solution for the long computing times that are now required by the algorithm.

Another benefit that may be derived from this approach is that the worstcase nature of the bounds is relaxed. This may result in smaller, more realistic model uncertainty bounds. The price that has to be paid for this is that more stringent assumptions on the noise acting on the system need to be made, which may in turn be unrealistic.

As a further refinement to this approach, it can be considered whether the linear programming problems can not be eliminated all together in the algorithm: the uncertainty in any (linear combination of) model parameters maps linearly to uncertainty in Nyquist plane. This is due to the linear parametrisation of the models. Let the matrix of the operator mapping the param-
eter uncertainty to frequency response uncertainty for a certain frequency $\omega^{\prime}$ be denoted $X\left(\omega^{\prime}\right)$. Then the covariance matrix of the model parameters maps to the covariance in Nyquist plane through pre-multiplication by $\left[\operatorname{Re} X\left(\omega^{\prime}\right)^{T}, \operatorname{Im} X\left(\omega^{\prime}\right)^{T}\right]^{T}$ and post multiplication by $\left[\operatorname{Re} X\left(\omega^{\prime}\right)^{T}, \operatorname{Im} X\left(\omega^{\prime}\right)^{T}\right]$. This basic procedure needs to be modified to exclude the structured error components from the unstructured errors. As only linear operations are involved in this exclusion, this seems straightforward.

### 8.2.3 Other uncertainty structures

The algorithm was developed for structured uncertainty that manifests itself as additive uncertainty with an unknown but bounded gain factor. This limits the class of uncertainty structures that can be described exactly by the algorithm. The practically relevant case of a moving pole for example can not be described in this way.

In section 5.8 the algorithm was applied to bound the uncertainty in the Youla parameter of a system transfer instead of the uncertainty in the system transfer itself. Translating the structured uncertainty in the Youla parameter back to uncertainty in the system transfer, a kind of structure is found in the system uncertainty that could not be obtained by direct application of the algorithm. In section 5.8 this was mainly done to bound the uncertainty in unstable systems, but a similar approach could possibly be used to make the algorithm suitable for more kinds of structure in the model uncertainty.

Being able to handle many uncertainty structures is important, because the more error structures can be recognised by the algorithm, the less conservative the eventual error bounds will be.

### 8.2.4 Basis generation

In the presentation of the algorithm it has been argued that the poles of the basis generating system should be chosen as close as possible to the poles of the true system in different operating points. Although some suggestions were given how to achieve this, some further automation of this step seems worthwhile.

## $\mathcal{A}$

## Overview of notation

A. 1 Latin symbols
A. 2 Greek symbols
A. 3 Other symbols
A. 4 Other notations
A. 5 Acronyms and abbreviations

## A. 1 Latin symbols

| $\hat{a}(t)$ | Contribution of basis functions 0 to $\tilde{k}$ to the extended noise . p. 107 bound due to inputs $u(0)$ to $u(t)$. |
| :---: | :---: |
| $a_{i}(t)$ | Impulse response of the $i$ th structured error component $A_{i}, \ldots$ p. 78 $i \in \boldsymbol{n}_{s}$. |
| $A_{i}$ | $i$ th structured error "direction": the structured error com- . p. p. 78 ponents are equal to $\delta_{i} A_{i}(\zeta), i \in \boldsymbol{n}_{\boldsymbol{s}}$. |
| $\hat{b}(t)$ | Contribution of basis functions $\bar{k}$ to $\tilde{k}$ to the extended noise . p. 107 bound due to inputs $u(-\infty)$ to $u(-1)$. |
| $b_{k}(t)$ | Impulse response of the $k$ th basis function. ............... p. 75 |
| $B_{k}(z)$ | Transfer function of the $k$ th basis function. .............. p. 75 |
| C | Complex plane. |
| $C_{0}$ | The designed controller. ...................................... . p. 61 |
| $\mathrm{C}_{0}$ | Set of candidate controllers. ................................. p. 61 |
| $C_{\text {sta }}$ | A stabilising controller for the system $S$. ................. p. 41 |
| $d(\ell)$ | Contribution of basis functions $\vec{k}$ to $\vec{k}$ to the extended . p. 111 cross-covariance bound due to inputs $u(0)$ to $u\left(T^{d}\right)$. ( $\ell$ is the index of the instrumental variable involved.) |
| $d_{u}$ | Bound on the inputs $u$, used in chapter 3. ................. p. 47 |
| $d_{\Delta}$ | Bound on the model uncertainty $\Delta$. .................... p. 78 |
| $d_{\Delta_{C}}$ | Bound on the difference between the designed controller and .. p. 61 the implemented controller. |
| $d_{\xi}$ | Bound on $\xi$, used in chapter 3. ............................ p. p. 47 |
|  | Continued on next pas |


| D | The set of all data set indices. ............................. p. p. 80 |
| :---: | :---: |
| $\mathcal{D}(w)$ | The set of all data sets of operating point $w$. ............. p. 80 |
| $\operatorname{diag}_{i \in I} X_{i}$ | The block diagonal matrix having the $X_{i}, i \in I$ on its diag- . p. p. 11 onal. |
| $e_{i}^{n}$ | $i$ th unit vector in $\mathbb{R}^{n}$ or $\mathbb{C}^{n}$. |
| $E^{\sigma}$ | A matrix in $\mathbb{R}^{q \times p}$ having a one in row $j$, column $i$ and zeros . p. 77 elsewhere ( $\sigma=(j, i)$ ). |
| $f(\ell)$ | Contribution of basis functions 0 to $\tilde{k}$ to the extended. p. 111 cross-covariance bound due to inputs $u(-\bar{t})$ to $u(-1)$. ( $\ell$ is the index of the instrumental variable involved.) |
| $\hat{g}(t)$ | Impulse response of the nominal model $\hat{G}$. ............... p. 78 |
| $g_{t r}^{w}(t)$ | Impulse response of the linear manifestation of the true pro- . p. 81 cess in the operating point $w$. |
| $g_{\theta}($ | Impulse response of the model $G_{\theta}$. . ....................... p. 78 |
| $\hat{G}$ | Nominal model. ............................................... p. p. 78 |
| $G_{t r}^{w}$ | Linear manifestation of the true process in the operating .. p. 81 point $w$. |
| $G_{\theta}$ | Model corresponding to parameter vector $\theta$. ............. p. 78 |
| $G_{\hat{\theta}}$ | Approximation of the nominal model by truncating its ex- . p. 113 pansion in a certain basis to a finite number of basis functions. |
| $G_{\boldsymbol{\theta}^{w}}$ | Auxiliary model, intended as an approximation for $G t^{w}$. .... p. 85 |
| $\mathcal{H}_{2}$ | The field of complex valued functions that are analytic out- . . p. 12 side the unit disc and square integrable over the unit circle. |
| $\mathcal{H}_{\infty}$ | The field of complex valued functions that are analytic out- . p. p. 13 side the unit and are bounded on the unit circle. |
| $I_{n}$ | Identity matrix in $\mathbb{R}^{n \times n}$. |
| $\operatorname{Im} z$ | Imaginary part of complex number $z$. |
| img $P$ | Image space of a linear operator $P$ or column span of a . . p. 11 matrix $P$. |
| $J_{\text {id }}$ | The identification criterion used in chapter 3. .............. p. 54 |
| $k^{*}$ | First parameter for which exponentially decaying bound on .. p. 82 tail applies: $\forall k \geq k^{*},\left\|\theta_{k}^{\sigma}\right\|<\bar{\theta}_{k}=M\left(\rho^{\sigma}\right)^{k}$. |
| $\mathcal{L}_{2}$ | The field of complex valued functions that are square inte- . p. 11 grable over the unit circle. |
| $\mathcal{L}_{\boldsymbol{\theta}}^{w}$ | Set of linear constraints on the model parameters in the $w$ th . p. 106 operating point representing data and prior knowledge. |
| $\mathcal{L}_{\theta}^{\prime}$ | Projection of $\mathcal{L}_{\theta}^{j}$ onto the orthogonal complement of the . p. 128 span of the parameter vectors of the structured error components for an implicit operating point $j \in \mathcal{W}$. |


| $M^{\sigma}$ | Part of the bound on the tail parameters: $\forall k \geq k^{*},\left\|\theta_{k}^{\sigma}\right\|<\bar{\theta}_{k}=M\left(\rho^{\sigma}\right)^{k}$ |
| :---: | :---: |
| $M_{i d}$ | Set of nominal models. .................................... p. 54 |
| $\mathcal{M}_{\text {rel }}$ | Set of all processes consistent with the nominal model and .. p. 58 the uncertainty bound. |
| $\mathcal{M}_{\text {set }}$ | Set of all possible models. ................................. p. p. 78 |
| $n_{s}$ | Number of structured error components. ..................... p. 78 |
| $n_{w}$ | The number of operating points for which experimental data .. p. 80 has been collected. |
| $\mathbb{N}$ | Set of non-negative integers: $0,1,2$, |
| $p$ | Number of inputs. |
| $\mathcal{P}$ | Process uncertainty set. The set of all processes consistent .. p. 51 with prior knowledge and experimental data. |
| $P_{A}$ | Projection onto $\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i \in n_{s}}$. In case of orthonormal $\boldsymbol{\theta}_{A, i}$ this . p. 128 is given by $\sum_{i=1}^{n_{s}} \boldsymbol{\theta}_{A, i} \boldsymbol{\theta}_{A, i}^{T}$. |
| $P_{A}^{\perp}$ | Projection onto the complement of $\left\langle\boldsymbol{\theta}_{A, i}\right\rangle_{i \in n_{s}}$. In case of . p. 128 orthonormal $\boldsymbol{\theta}_{A, i}$ this is given by $I-\sum_{i=1}^{n_{s}} \boldsymbol{\theta}_{A, i} \boldsymbol{\theta}_{A, i}^{T}$. |
| $\mathcal{P}_{i}$ | All processes in $Q$ consistent with a certain type of prior .. p. 51 knowledge. |
| $\mathcal{P}_{m}\left(\omega_{i}\right)$ | Outer-bounding polytope for $\mathcal{P}_{\omega_{i}}$, obtained by intersecting . p. 118 $m$ half-planes. |
| $\overline{\mathcal{P}}_{m}\left(\omega_{i}\right)$ | Polytope in complex plane containing $\Delta\left(e^{j \omega_{i}}\right)$. ......... p. 118 |
| $\mathcal{P}_{\omega_{i}}$ | Polytope in complex plane containing all values representing . p. 117 the set $\left\{G_{\theta}\left(e^{j \omega_{i}}\right)-\hat{G}\left(e^{j \omega_{i}}\right) \mid \boldsymbol{\theta} \in \mathcal{L}_{\theta}\right\}$. |
| $q$ | Number of outputs. |
| 2 | Class of processes to which the true process $S$ is assumed . . p. 42 to belong approximately. |
| $\mathbb{R}$ | Set of real numbers. |
| $\operatorname{rank} M$ | The rank of a matrix $M$. |
| $\operatorname{Rez}$ | Real part of complex number $z$. |
| $\mathcal{R H}_{2}$ | The subset of $\mathcal{H}_{2}$ consisting of finite dimensional systems. |
| $\mathcal{R} \mathcal{H}_{\infty}$ | The subset of $\mathcal{H}_{\infty}$ consisting of finite dimensional systems. |
| $S$ | True system, interpreted as a behaviour: $S \subset W^{T}$. .......p. 37 |
| S | (chapter 3): Set of all systems of the same kind as the true .. p. 38 system $S$. |
| $\mathcal{S}$ | (except chapter 3): Set of subtransfers: $\mathcal{S}=\boldsymbol{q} \times \boldsymbol{p} . \ldots . . . .$. p. 77 |
| $\operatorname{sbs}_{i \in I} X_{i}$ | The matrix obtained by putting the $X_{i}, i \in I$ next to each .. p. 11 other (Side By Side). |
| $\operatorname{stack}_{i \in I} X_{i}$ | The matrix obtained by putting the $X_{i}, i \in I$ on top of. p. 10 each other. |
|  | . Continued on next page ........................ |


| TT | (chapter 3): The time set of a behavioural representation. .. p. 35 |
| :---: | :---: |
|  | A behaviour is a subset of $W^{\mathbb{T}}$. |
| $T$ | (except in chapter 3): The time interval on which a data set . . p. 80 is collected. |
| $T^{\text {d }}$ | The time interval on which data set $d$ is collected. .......... p. 80 |
| W | The experimental data set. ............................... p. p. 40 |
| W | The signal alphabet of a behavioural representation. A be- . . p. 35 haviour is a subset of $W^{T}$. |
| W | The set of operating points. ................................. p. 80 |
| $u$ | General input. |
| $\bar{u}$ | Bound on inputs before $t=0: \forall t<0,\|u(t)\|<\bar{u} . \quad \ldots \ldots \ldots \ldots$ p. 82 |
| $u^{d}$ | Input measurements for data set $d$. .......................... p. p. 80 |
| $v^{d}$ | Instrumental variable for data set $d . . . . . . . . . . . . . . . . . . . . . . . . ~ p . ~ p . ~ 81 ~$ |
| $y$ | General output. |
| $y^{d}$ | Output measurements for data set $d$. ..................... . p. 80 |
| $\mathbb{Z}$ | Set of integers: ..., $-2,-1,0,1,2$, |
| A. 2 | Greek symbols |
| $\delta$ | Contribution of basis functions $\tilde{k}+1$ to $\infty$ to the extended . p. 108 noise bound due to inputs $u(-\infty)$ to $u(t)$. |
| $\delta_{1}(\ell)$ | Contribution of basis functions 0 to $\tilde{k}$ to the extended. p. 111 cross-covariance bound due to inputs $u(-\infty)$ to $u(-\bar{t}-1)$. ( $\ell$ is the index of the instrumental variable involved.) |
| $\delta_{2}(\ell)$ | Contribution of basis functions $\tilde{k}$ to $\infty$ to the extended. p. 111 cross-covariance bound due to the inputs $u(-\infty)$ to $u\left(T^{d}\right)$. ( $\ell$ is the index of the instrumental variable involved.) |
| $\Delta$ | General model uncertainty. In chapters 4-6 more specifically .. p. 78 the unstructured model uncertainty. |
| $\Delta_{C}$ | The uncertainty in the implemented controller. ............... p. 61 |
| $\zeta$ | Forward shift operator: $[\zeta f](t)=f(t+1) . \ldots \ldots \ldots \ldots \ldots \ldots$ p. 12 |
| $\theta$ | Model parameter vector. . ...................................... 78 |
| $\bar{\theta}_{k}$ |  |
| $\hat{\mu}_{\ell}\left(\omega^{\prime}\right)$ | Uncertainty of $\Delta\left(j \omega^{\prime}\right)$ in the direction $e^{j \phi_{\ell}}$ due to the model p. 117 parameter uncertainty. |
| $\bar{\mu}\left(\omega^{\prime}\right)$ | Uncertainty of $\Delta\left(j \omega^{\prime}\right)$ in any direction in the complex plane . p. 118 due to the tail of the basis. |
| $\xi$ | Signal accounting for all effects that can not be accounted for . . p. 43 by a relation between $u$ and $y$ present in $Q$ (non-linearities, disturbances, noise, etc.) |
|  | Continued on next page. . |

```
.............Greek symbols, continued from previous page
\rho
        \forallk\geq\mp@subsup{k}{}{*},|\mp@subsup{0}{k}{\sigma}|<\mp@subsup{\overline{0}}{k}{}=M(\mp@subsup{\rho}{}{\sigma}\mp@subsup{)}{}{k}.
```



```
\sigmai}\quadith singular value of a matrix. Singular values are sorted in
        order of decreasing magnitude.
```


## A. 3 Other symbols

$a \wedge b \quad a$ and $b$ (logical and).
$a \vee b \quad a$ or $b$ (logical or).
$a \Longrightarrow b \quad a$ implies $b$.
$a \Longleftrightarrow b \quad a$ is equivalent to $b$, i.e. $a$ if and only if $b$
$a:=b \quad a$ shall be defined as $b$.
$a=: b \quad a$ shall be the definition of $b$.

## A. 4 Other notations

In this section, it is not the symbols that determine the interpretation, but the way in which they are denoted:
$z^{*} \quad$ The complex conjugate of $z \in \mathbb{C}$.
$A^{T} \quad$ The transpose of a matrix $A \in \mathbb{R}^{m \times n}$, the complex conjugate transpose of a matrix $A \in \mathbb{C}^{m \times n}$.
$2^{\mathcal{S}} \quad$ The set of all subsets of $\mathcal{S}$.
$\lfloor a\rfloor \quad$ The largest integer not exceeding $a$.
$\lceil a\rceil$ The smallest integer not exceeded by $a$.
$\lceil a\rceil_{b} \quad$ The smallest multiple of $b$ not exceeded by $a,\lceil a\rceil_{b}:=\left\lceil\frac{a}{b}\right\rceil b$.
$\langle a, b, c\rangle$ The span of vectors $a, b, c$.
$(a \mid b) \quad$ The inner product of vectors $a$ and $b$.
$M_{i *} \quad i$ th row of matrix $M$.
$M_{* i} \quad i$ th column of matrix $M$.
$n \quad$ The range $1,2, \ldots, n \quad(n \in \mathbb{N})$.
$\underline{x} \quad$ A stochastic variable/process $x$.
$x_{i} \quad$ A sample from the stochastic process $\underline{x}$.
$\operatorname{var} \underline{x} \quad$ The variance of the stochastic process $\underline{x}$.

## A. 5 Acronyms and abbreviations

ARX Auto Regressive with eXternal input; a model structure.
CASID Computer Aided System IDentification.
FIR Finite Impulse Response; a model structure.
Matlab A commercial software package for matrix manipulation and graphical presentation.
MIMO Multiple input, multiple output.
MISO Multiple input, single output.
PRBNS Pseudo random binary noise sequence.
SIMO Single input, multiple output.
SISO Single input, single output.

## B

## System based orthonormal basis functions

B. 1 Generating a basis from a
general system
B. 2 Properties of system
based orthonormal basis functions

## B. 3 Transients of a generalised FIR model

In (Heuberger, 1991) system based orthonormal basis functions were introduced. This section will briefly present a number of results on these orthonormal basis functions taken from (Heuberger, 1991; Heuberger et al., 1995). Also a conjecture taken from (Hakvoort, 1994, Section 4.B) and used in chapters 4 and 5 of this thesis is stated.

After these results by other authors, some original results are presented. These results are concerned with the space of transients that an unknown linear combination of a (known) number of basis functions may exhibit. In general this space is spanned by a number of signals that is equal to the sum of the McMillan degrees of the basis functions involved in the linear combination. For this particular basis, a procedure is derived yielding a much smaller number of signals spanning this space. These signals can then be applied in the estimation of transients as discussed in section 5.2.

Before presenting these results, first some preliminary definitions: Let $G_{b}(z)$ be the transfer function of a single-input, multiple-output, stable, finite dimensional, discrete time, causal, linear, time-invariant system, so let $G_{b} \in \mathcal{R} \mathcal{H}_{2}^{q \times 1}$, where $q$ denotes the number of outputs of $G_{b}$. Let $n_{b}$ denote the McMillan degree of $G_{b}$. The notation

$$
G_{b}(z)=(A, B, C, D)
$$

means that

$$
\begin{equation*}
G_{b}(z)=D+C(z I-A)^{-1} B \tag{B.1}
\end{equation*}
$$

where

$$
\begin{equation*}
A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times 1}, C \in \mathbb{R}^{q \times n}, D \in \mathbb{R}^{q \times 1}, n \geq n_{b} . \tag{B.2}
\end{equation*}
$$

The quadruple $(A, B, C, D)$ is called a state-space representation of $G_{b}(z)$. Such a representation is called minimal if $n=n_{b}$. A realisation $(A, B, C, D)$ is called stable if it is minimal and all eigenvalues of $A$ have an absolute value that is less than one. Equivalently, $G_{b}(z)$ is stable if $G_{b}(z)$ is analytic outside the unit circle. For such a realisation, the controllability gramian $P$ is defined as the solution of

$$
A P A^{*}+B B^{*}=P
$$

and the observability gramian $Q$ as the solution of

$$
A^{*} Q A+C^{*} C=Q
$$

A stable realisation is called balanced if $P=Q=\Sigma$ and input balanced if $P=I$ and $Q=\Sigma^{2}$, where $\Sigma$ is a diagonal matrix containing the Hankel singular values in decreasing order on its diagonal. Finally, a system $G \in \mathcal{R} \mathcal{H}_{2}$ is called inner if it is stable and satisfies

$$
G^{T}\left(z^{-1}\right) G(z)=I
$$

## B. 1 Generating a basis from a general system

Given an arbitrary $G_{b} \in \mathcal{R} \mathcal{H}_{2}^{q \times 1}$, an inner function $G_{i}(z)$ can be derived from $G_{b}$ using the following theorem. In the statement of this theorem, the convention to denote the impulse response parameters of a system $G_{i}(\zeta)$ by $g_{i}(k), k=0, \ldots$ has been used.

Theorem B. 1 (see (Heuberger et al., 1995)) Let $G_{b} \in \mathcal{R H}_{2}^{q \times 1}$ have a (minimal) input balanced realisation $\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$. Denote the McMillan degree of $G_{b}$ by $n_{b}$ and let $n_{b}>0$. It now holds:
(a) There exist $C_{i} \in \mathbb{R}^{1 \times n_{b}}$ and $D_{i} \in \mathbb{R}$ such that $\left(A_{b}, B_{b}, C_{i}, D_{i}\right)$ is a minimal balanced realisation of an inner function $G_{i}$ with $\left|g_{i}(0)\right|<1$.
(b) $\left(A_{b}, B_{b}, C_{i}, D_{i}\right)$ has the property (a) if and only if

$$
\begin{gathered}
C_{i}=B_{b}^{*}\left(I+A_{b}^{*}\right)^{-1}\left(I+A_{b}\right) \\
D_{i}=B_{b}^{*}\left(I+A_{b}^{*}\right)^{-1} B_{b}-1
\end{gathered}
$$

In the derivation of the inner function $G_{i}(z)$ from the transfer function $G_{b}(z)$ only the matrices $A_{b}$ and $B_{b}$ of an input balanced realisation $\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$ of $G_{b}(z)$ were used. Instead of saying that $G_{i}(z)$ was derived from $G_{b}(z)$, we might as well say that $G_{i}(z)$ was derived from the input balanced pair $\left(A_{b}, B_{b}\right)$. Because $G_{b}$ and $G_{i}$ can be represented by state-space realisations using the same system matrix $A_{b}, G_{i}$ has the same poles as $G_{b}$.

With $G_{b}=\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$ and $G_{i}=\left(A_{b}, B_{b}, C_{i}, D_{i}\right)$ as in theorem B.1, define $V_{k} \in \mathcal{R} \mathcal{H}_{2}^{n_{b} \times 1}, k \in \mathbb{N}$ recursively as

$$
\begin{align*}
& V_{0}(z):=\left(z I-A_{b}\right)^{-1} B_{b}=\left(A_{b}, B_{b}, I_{n_{b}}, 0\right) \\
& V_{k}(z):=V_{k-1}(z) G_{i}(z), \quad k \in \mathbb{N} \backslash\{0\} \tag{B.3}
\end{align*}
$$

The $V_{k}(z)$ have the following property (see (Heuberger et al., 1995))

$$
\forall k, \ell \in \mathbb{N} \quad \sum_{t=0}^{\infty} v_{k}(t)^{T} v_{\ell}(t)= \begin{cases}I_{n_{b}} & \text { if } k=\ell  \tag{B.4}\\ 0 & \text { if } k \neq \ell\end{cases}
$$

where $v_{k}(t)$ is the impulse response associated with the transfer function $V_{k}(z)$. Any system $G_{0} \in \mathcal{R} \mathcal{H}_{2}$ can now be written as

$$
G_{0}(z)=D+\sum_{k=0}^{\infty} L_{k} V_{k}(z), \quad D \in \mathbb{R}, L_{k} \in \mathbb{R}^{1 \times n_{b}}
$$

This is already very similar to the expansion

$$
G_{0}(z)=\sum_{k=0}^{\infty} \theta_{k} B_{k}(z), \quad \theta_{k} \in \mathbb{R}, B_{k} \in \mathcal{R} \mathcal{H}_{2}
$$

that was used in chapter 4 and further. Indeed, the $B_{k}, k \in \mathbb{N}$ are obtained as follows

$$
\begin{aligned}
B_{0}(z) & =1 \\
B_{k n_{b}+\ell}(z) & =\left[V_{k}(z)\right]_{\ell}, \quad k \in \mathbb{N}, \ell \in n_{b}
\end{aligned}
$$

The $B_{k}, k \in \mathbb{N}$ obtained in this way represent a basis for $\mathcal{R} \mathcal{H}_{2}$ generated by $G_{b}$. Because of (B.4), the basis is an orthonormal basis. The $L_{k}$ are called the grouped expansion coefficients of $G_{0}$ expressed in the basis generated by $G_{b}$. The $V_{k}$ are called the grouped basis functions generated by $G_{b}$ or by $\left(A_{b}, B_{b}\right)$. It holds

$$
L_{k}=\left[\theta_{k n_{b}+1}, \ldots, \theta_{(k+1) n_{b}}\right]
$$

A system of the form

$$
G(z)=\sum_{k=0}^{\bar{k}} \theta_{k} B_{k}(z)
$$

is called a generalised FIR model. The expansion coefficients $\theta_{k}, k=0, \ldots, \bar{k}$ are called generalised FIR parameters.

## B. 2 Some known properties of system based orthonormal basis functions

Property 4.1 on page 77 is now trivial. In the notation of this appendix, property 4.1 reads

$$
\forall k \in \mathbb{N}, \ell \in n_{\mathrm{b}}, \omega^{t} \in \mathbb{R} \quad\left|B_{\ell}\left(e^{j \omega^{\prime}}\right)\right|=\left|B_{k n_{b}+\ell}\left(e^{j \omega^{t}}\right)\right|
$$

This equality follows immediately from the fact that

$$
\begin{aligned}
&\left|B_{k n_{b}+\ell}\left(e^{j \omega^{t}}\right)\right|=\left|B_{\ell}\left(e^{j \omega^{\prime}}\right) G_{i}^{k}\left(e^{j \omega^{\prime}}\right)\right| \\
&=\left|B_{\ell}\left(e^{j \omega^{\prime}}\right)\right| \cdot\left|G_{i}^{k}\left(e^{j \omega^{\prime}}\right)\right|=\left|B_{\ell}\left(e^{j \omega^{\prime}}\right)\right| \cdot 1
\end{aligned}
$$

The fact that the expansion coefficients $\theta_{k}$ of a system $G$ expressed in a basis generated by $G_{b}$ decay to zero faster if the poles of $G_{b}$ are closer to the poles of $G$ follows from the following theorem. For convenience, this theorem is stated in terms of the grouped expansion coefficients $L_{k}$ instead of $\theta_{k}$.

Theorem B. 2 (see (Heuberger et al., 1995)) Let $G \in \mathcal{R} \mathcal{H}_{2}$ have a statespace representation $(A, B, C, D)$. Denote the McMillan degree of $G$ by $n$ and let $\left(A_{b}, B_{b}\right)$ be an input balanced pair that generates a set of grouped basis functions $\left\{V_{k}\right\}_{k=0}^{\infty}, A_{b} \in \mathbb{R}^{n_{b} \times n_{b}}, B_{b} \in \mathbb{R}^{n_{b}}$. Denote the eigenvalues of $A$ by $\mu_{i}, i \in n$. Denote the eigenvalues of $A_{b}$ by $\rho_{j}, j \in n_{b}$. Express $G$ as follows

$$
G(z)=D+\sum_{k=0}^{\infty} L_{k} V_{k}(z)
$$

$0, L_{0}, L_{1}, \ldots$ is the sequence of impulse response coefficients of a dynamical system

$$
G^{\prime}(z)=\sum_{k=0}^{\infty} L_{k} z^{-k-1}
$$

with state-space representation $(X, Y, Z, 0)$ satisfying
(a) $X$ has dimension $n$, i.e the McMillan degree of $(X, Y, Z, 0)$ does not exceed that of $G$.
(b) $X$ has eigenvalues $\lambda_{i}, i \in \boldsymbol{n}$ for which it holds

$$
\forall i \in n, \quad\left|\lambda_{i}\right|=\prod_{j=1}^{n_{b}}\left|\frac{\mu_{i}-\rho_{j}}{1-\mu_{i} \rho_{j}}\right|
$$

In the remainder of this appendix, the following theorem will be used
Theorem B. 3 (see (Heuberger et al., 1995)) Let $G_{b} \in \mathcal{R} \mathcal{H}_{2}$ be a system with minimal input balanced state-space representation $\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$ and with McMillan degree $n_{b}, n_{b}>0$. Let $\left(A_{b}, B_{b}, C_{i}, D_{i}\right)$ be the representation of an inner transfer function derived from $G_{b}$. If a transfer function $G(z)$ can be expressed by a sum of $m$ grouped basis functions $V_{k} k=0, \ldots, m-1$, generated by $G_{b}, m \in \mathbb{N}_{+}$, i.e. there exist $L_{k}, k=0, \ldots, m-1$, such that

$$
G(z)=D+\sum_{k=0}^{m-1} L_{k} V_{k}(z)
$$

then $G(z)$ has a state-space realisation $\left(A_{m}, B_{m}, C_{m}, D\right)$ with state dimension $m \cdot n_{b}$,

$$
C_{m}=\left[L_{0}, \ldots, L_{m-1}\right]
$$

and For $m \geq>1$

$$
A_{m}=\left[\begin{array}{ccccc}
A_{b} & 0 & \ldots & \ldots & 0 \\
B_{b} C_{i} & A_{b} & 0 & \ldots & 0 \\
B_{b} D_{i} C_{i} & B_{b} C_{i} & A_{b} & \ddots & 0 \\
\vdots & \vdots & \ddots & \ddots & 0 \\
B_{b} D_{i}^{m-2} C_{i} & B_{b} D_{i}^{m-3} C_{i} & \ldots & B_{b} C_{i} & A_{b}
\end{array}\right], \quad B_{m}=\left[\begin{array}{c}
B_{b} \\
B_{b} D_{i} \\
B_{b} D_{i}^{2} \\
\vdots \\
B_{b} D_{i}^{m-1}
\end{array}\right]
$$

For $m=1$ it holds in particular $A_{m}=A_{b}, B_{m}=B_{b}$.
In (Hakvoort, 1994) the following conjecture is stated:
Conjecture B. 4 Consider the pulse response representation of any basis function, generated with the procedure described in (Heuberger et al., 1995),

$$
\forall i \in n_{b}, \quad\left[V_{k}(z)\right]_{i}=\sum_{t=0}^{\infty}\left[v_{k}(t)\right]_{i} z^{-t}
$$

then it is conjectured that for any $p \geq 0$,

$$
\sum_{t=0}^{\infty} t^{p}\left|\left[v_{k}(t)\right]_{i}\right| \leq c_{1}^{i}+c_{2}^{i} k^{p+1 / 2}
$$

for some finite constants $c_{1}^{i}, c_{2}^{i}$ which are independent of $k$, but possibly depend on $p$. This implies that

$$
\left\|\left[V_{k}(z)\right]_{i}\right\|_{\ell_{1}} \leq c_{1}^{i}+c_{2}^{i} \sqrt{k}
$$

The notation in the statement of this conjecture has been slightly changed to put it more in line with the notation used in the rest of this appendix. The conjecture is motivated in (Hakvoort, 1994) by "extensive simulations."

The reader may verify that conjecture 4.2 on page 77 is implied by the above conjecture.

## B. 3 Transients of a generalised FIR model

In this section the following problem is considered. For a system of the form

$$
\begin{equation*}
G(z)=\sum_{k=0}^{\bar{k}} \theta_{k} B_{k}(z) \tag{B.5}
\end{equation*}
$$

but with unknown coefficients $\theta_{k}, k=0, \ldots, \bar{k}$, find a set $\left\{y_{t r, 1}(t), \ldots, y_{t r, \ell}(t)\right\}$ of signals, such that all transients that a system of the form (B.5) may exhibit can be written as a linear combination of the signals in $\left\{y_{t r, 1}(t), \ldots, y_{t r, \ell}(t)\right\}$. Moreover, to reduce computational complexity, an attempt should be made to make $\ell$ small.

For a general stable $q \times p$ system having a state-space representation with state dimension $n$ given by $(A, B, C, D)$, the transient of that system for an initial state $x_{0}$ is given by

$$
\left[\begin{array}{c}
y^{T}(0)  \tag{B.6}\\
y^{T}(1) \\
\vdots \\
y^{T}(k) \\
\vdots
\end{array}\right]=\left[\begin{array}{c}
x_{0}^{T} C^{T} \\
x_{0}^{T} A^{T} C^{T} \\
\vdots \\
x_{0}^{T}\left(A^{k}\right)^{T} C^{T} \\
\vdots
\end{array}\right]=: Y_{t r}\left(x_{0}\right)
$$

For a single-output system, (B.6) can be rewritten

$$
\left[\begin{array}{c}
y(0)  \tag{B.7}\\
y(1) \\
\vdots \\
y(k) \\
\vdots
\end{array}\right]=\left[\begin{array}{c}
C \\
C A \\
\vdots \\
C A^{k} \\
\vdots
\end{array}\right] x_{0} .
$$

$Y_{t r}\left(x_{0}\right)$ is linear in $x_{0}$. Therefore the space of all transients that the system $(A, B, C, D)$ may exhibit is spanned by

$$
\left\{Y_{t r}\left(e_{1}^{n}\right), \ldots, Y_{t r}\left(e_{n}^{n}\right)\right\}
$$

Now consider the first $n$ samples of the state signals in response to an impulse on $t=0$. It is easily verified that it holds

$$
[x(0), x(1), \ldots, x(n-1)]=\left[B, A B, \ldots, A^{n-1} B\right] .
$$

For a controllable representation $(A, B, C, D)$ we have

$$
\operatorname{rank}\left[B, A B, \ldots A^{n-1} B\right]=n
$$

so that

$$
\left\langle e_{1}^{n}, \ldots, e_{n}^{n}\right\rangle=\left\langle B, A B, \ldots, A^{n-1} B\right\rangle
$$

If $h \in h_{2}^{q}$ denotes the impulse response of $(A, B, C, D)$, then it follows that

$$
\left\langle Y_{t r}\left(e_{1}^{n}\right), \ldots, Y_{t r}\left(e_{n}^{n}\right)\right\rangle=\left\langle\left[\zeta h^{T}\right](t), \ldots,\left[\zeta^{n} h^{T}\right](t)\right\rangle=\left\langle h^{T}(t+1), \ldots, h^{T}(t+n)\right\rangle
$$

where the shifted signals are defined on $\mathbb{N}$. This is summarised by the following lemma.

Lemma B. 5 The space of transients of the system $(A, B, C, D)$ with statedimension $n$ is for controllable $(A, B)$ spanned by the $n$ signals that are obtained by shifting the impulse response of $(A, B, C, D) 1$ to $n$ samples forward in time.

Now let $(A, B)$ be an input balanced pair with $B \in \mathbb{R}^{n}$. According to lemma B.5, the space of transients of the system represented by $\left(A, B, I_{n}, 0\right)$ is spanned by

$$
\left\{\left[\begin{array}{c}
B^{T} \\
B^{T} A^{T} \\
B^{T}\left(A^{2}\right)^{T} \\
\vdots
\end{array}\right],\left[\begin{array}{c}
B^{T} A^{T} \\
B^{T}\left(A^{2}\right)^{T} \\
B^{T}\left(A^{3}\right)^{T} \\
\vdots
\end{array}\right], \ldots,\left[\begin{array}{c}
B^{T}\left(A^{n-1}\right)^{T} \\
B^{T}\left(A^{n}\right)^{T} \\
B^{T}\left(A^{n+1}\right)^{T} \\
\vdots
\end{array}\right]\right\}
$$

Instead of the $1 \times n$ system $\left(A, B, I_{n}, 0\right)$ consider now the following set of SISO systems

$$
\begin{equation*}
\mathcal{G}:=\left\{G(z) \in \mathcal{R} \mathcal{H}_{2} \mid \exists C \in \mathbb{R}^{1 \times n}, \quad G(z)=(A, B, C, 0)\right\} \tag{B.8}
\end{equation*}
$$

and let

$$
y:=\left\{y \in \mathcal{R} \mathcal{H}_{2} \mid \exists G \in \mathcal{G}, \quad y \text { is a transient of } G\right\}
$$

be the set of transients that can be exhibited by all systems in $\mathcal{G}$.

Clearly,

$$
y=\left\langle\left[\begin{array}{c}
B^{T}  \tag{B.9}\\
B^{T} A^{T} \\
\vdots
\end{array}\right]_{* 1}, \ldots,\left[\begin{array}{c}
B^{T} \\
B^{T} A^{T} \\
\vdots
\end{array}\right]_{* n},\left[\begin{array}{c}
B^{T} A^{T} \\
B^{T}\left(A^{2}\right)^{T} \\
\vdots
\end{array}\right]_{* 1}, \ldots,\left[\begin{array}{c}
B^{T}\left(A^{n-1}\right)^{T} \\
B^{T}\left(A^{n}\right)^{T} \\
\vdots
\end{array}\right]_{* n}\right\rangle
$$

Because

$$
\left[\begin{array}{c}
B^{T}\left(A^{k}\right)^{T} \\
B^{T}\left(A^{k+1}\right)^{T} \\
\vdots
\end{array}\right]=\left[\begin{array}{c}
B^{T} \\
B^{T} A^{T} \\
\vdots
\end{array}\right]\left(A^{k}\right)^{T}
$$

the $n+1$ st to $n^{2}$ th signal on the right hand side of (B.9) depend linearly on the first to $n$th signal. As a result

$$
y=\left\langle\left[\begin{array}{c}
B^{T}  \tag{B.10}\\
B^{T} A^{T} \\
\vdots
\end{array}\right]_{* 1}, \ldots,\left[\begin{array}{c}
B^{T} \\
B^{T} A^{T} \\
\vdots
\end{array}\right]_{* n}\right\rangle
$$

The connection of this result with the space of transients that a generalised FIR model can exhibit is given by the following lemma.

Lemma B. 6 The matrices $A_{m}$ and $B_{m}, m \in \mathbb{Z}_{+}$as defined in theorem B.3 on page 295 form an input balanced pair.

The reader may convince himself of this result by noting that the Z-transform of the impulse response of the system $\left(A_{m}, B_{m}, I, 0\right)$ is given by

$$
\left[V_{0}(z)^{T}, \ldots, V_{m-1}(z)^{T}\right]^{T}
$$

and that the controllability gramian $P$ of this system satisfies

$$
P=\sum_{k=0}^{\infty} B_{m} A_{m}^{k}\left(A_{m}^{T}\right)^{k} B_{m}^{T}=\sum_{t=0}^{\infty}\left[\begin{array}{c}
v_{0}(t) \\
\vdots \\
v_{m}(t)
\end{array}\right]\left[v_{0}^{T}(t), \ldots, v_{m}^{T}(t)\right]=I_{m n_{k}}
$$

Remark B. 1 In (Heuberger, 1991) the line of reasoning was actually the other way round. It was established there by other means that $\left(A_{m}, B_{m}\right)$ was an input balanced pair and that lead to the conclusion that the $V_{k}$ must satisfy equation (B.4).

These results are now combined in the following way. The set of systems of the form

$$
G(z)=\sum_{k=0}^{m n_{b}} \theta_{k} B_{k}(z), \quad m \in \mathbb{N}
$$

for unknown $\theta_{k}$ is a set like $\mathcal{G}$ in (B.8), where the input balanced pair $(A, B)$ is to be taken equal to $\left(A_{m}, B_{m}\right)$ as given in theorem B.3. According to (B.10), the space of transients of all systems in this set can be spanned by the impulse responses of basis functions 1 to $m n_{b}$ shifted one sample forward in time:

Lemma B. 7 Let $y$ be the set of transients that can be exhibited by models $G(z)$ of the form

$$
\begin{equation*}
G(z)=\sum_{k=0}^{m n_{b}} \theta_{k} B_{k}(z), \quad m \in \mathbb{N} \tag{B.11}
\end{equation*}
$$

where $\left\{B_{k}(z)\right\}$ is a set of basis functions generated by a system $G_{b}(z)$ with state space representation $\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$ and with McMillan degree $n_{b}$. Let $b_{k}(t)$ be the impulse response of the $k t h$ basis function, then

$$
y=\left\langle\zeta b_{1}, \ldots, \zeta b_{m n_{b}}\right\rangle
$$

This gives a particularly easy procedure to generate a set of signals to span the space of transients that can be exhibited by models of the form (B.11): for every basis function $B_{k}(z), k=1, \ldots, m n_{b}$, calculate the impulse response of that basis function and discard the first sample. The set of signals obtained in this way spans the space of transients. $B_{0}(z)$ is excluded from this procedure, because $b_{0}(t)=\delta(t)$ and $\zeta b(t)=0$.

That only $m n_{b}$ signals are needed to span the space of transients is not surprising considering the realisation that was given for such systems in theorem B.3. However, lemma B. 7 goes beyond that result by specifying an explicit set of $m n_{b}$ signals spanning that space.

For models of the form (B.5) with $\bar{k} / n_{b} \notin \mathbb{N}$ lemma. B. 7 does not provide a procedure to generate a set of signals spanning the space of transients for all such models. To state the procedure for such models, some extra notation needs to be introduced.

Let $\left\{B_{k}(z)\right\}$ be a set of basis functions generated by the system $G_{b}$ with realisation $\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$ and with McMillan degree $n_{b}$. Let $\left(A_{b}, B_{b}, C_{i}, D_{i}\right)$ be a realisation of an inner function derived from $G_{b}$. Let $m$ be the smallest integer such that $m n_{b}>\bar{k}$. The set

$$
\mathcal{G}_{i}^{j}:=\left\{G(z) \in \mathcal{R} \mathcal{H}_{2} \mid \exists \theta_{k}, k=i, \ldots, j, \quad G(z)=\sum_{k=i}^{j} \theta_{k} B_{k}(z)\right\}
$$

contains all models that are linear combinations of basis functions $i$ through $j$. Assume that all models of the form (B.5) are represented by a realisation as given in theorem B.3. Note that a number of elements of the last grouped FIR parameter is zero if $\bar{k}<m n_{b}$ and $n_{b}>1$ :

$$
\left[L_{m-1}\right]_{\bar{k}+1-(m-1) n_{b}}=\cdots=\left[L_{m-1}\right]_{n_{b}}=0
$$

For $X \subset \mathbb{R}^{m n_{b}}$ the set

$$
y(X)_{i}^{j}:=\left\{y \in \mathcal{R} \mathcal{H}_{2} \mid \exists G \in \mathcal{G}_{i}^{j}, x \in X, \quad y \text { is the transient of } G \text { with } x_{0}=x\right\}
$$

contains all transients of models in $\mathcal{G}_{i}^{j}$ for initial states restricted to the set $X$. In this notation, lemma B. 7 states that

$$
y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{m n_{b}}=\left\langle\zeta b_{1}, \ldots, \zeta b_{m n_{b}}\right\rangle
$$

The problem that needs to be solved is to find a set of signals that spans

$$
y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{\bar{k}}
$$

There always exist models $G \in \mathcal{G}_{0}^{m n_{b}}$ such that the realisation of $G$ according to theorem B. 3 is fully observable. Because this may not be the case for $G \in \mathcal{G}_{0}^{\bar{k}}$, $k<m n_{b}$, this problem can not be solved by applying lemma B.7. As an example of this, consider an input balanced pair ( $A_{b}, B_{b}$ ) from which an input balanced pair $\left(A_{2}, B_{2}\right)$ with state dimension equal to twice the state dimension of $\left(A_{b}, B_{b}\right)$ is constructed according to theorem B.3: $\left(A_{2}, B_{2}\right)$ is taken equal to $\left(A_{m}, B_{m}\right)$ for $m=2$. Suppose that the dimension of $A_{b}$ is $n_{b}$ and that $\bar{k}=n_{b}$, so $n_{b}+1$ of the basis functions generated from $\left(A_{2}, B_{2}\right)$ are incorporated in the model set $\mathcal{G}_{0}^{\bar{k}}$. From the structure of $A_{2}$, see theorem B.3, can be concluded that for any $G \in \mathcal{G}_{0}^{\bar{k}}$ the $n_{b}+1$ st to $2 n_{b}$ th entry of the state vector of its realisation according to theorem B. 3 are not observable. Consequently the response to initial states $x$ for which

$$
\forall k \in \boldsymbol{n}_{\boldsymbol{b}}, \quad[x]_{k}=0
$$

is zero for all models in $9_{0}^{\bar{k}}$ and so

$$
y\left(\mathbb{R}^{2 n_{b}}\right)_{0}^{\bar{k}} \subsetneq y\left(\mathbb{R}^{2 n_{b}}\right)_{0}^{2 n_{b}}
$$

Let

$$
\begin{equation*}
X^{\prime}:=\left\{x \in \mathbb{R}^{m n_{b}} \mid \forall k \in n_{b}, \quad[x]_{k}=0\right\} \tag{B.12}
\end{equation*}
$$

then the structure in $A_{m}$ implies that

$$
\begin{equation*}
y\left(X^{\prime}\right)_{(m-1) n_{b}+1}^{m n_{b}}=y\left(\mathbb{R}^{m n_{b}}\right)_{(m-2) n_{b}+1}^{(m-1) n_{b}}=y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{(m-1) n_{b}} \tag{B.13}
\end{equation*}
$$

In words this means that any transient that can be observed for $B_{(m-1) n_{b}+1}$ to $B_{m n_{b}}$ in response to an initial condition for which the first $n_{b}$ entries are equal to zero can also be observed for at least one of the basis functions $B_{0}$ to $B_{(m-1) n_{b}}$ in response to some other (arbitrary) initial condition. It generally holds

$$
\begin{equation*}
y\left(X^{\prime}\right)_{(m-1) n_{b}+1}^{\frac{\pi}{k}} \subset y\left(X^{\prime}\right)_{(m-1) n_{b}+1}^{m n_{b}}, \quad \tilde{k}<m n_{b} \tag{B.14}
\end{equation*}
$$

and

$$
y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{\bar{k}}=y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{(m-1) n_{b}}+y\left(\mathbb{R}^{m n_{b}}\right)_{(m-1) n_{b}+1}^{\bar{k}}, \quad(m-1) n_{b}<\bar{k}
$$

Due to the linearity of a transient in the entries of its initial state vector it follows that

$$
\begin{align*}
y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{\bar{k}} & =y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{(m-1) n_{b}}+y\left(X^{\prime}\right)_{(m-1) n_{b}+1}^{\bar{k}}+y\left(X^{\prime \prime}\right)_{(m-1) n_{b}+1}^{\bar{k}} \\
& =y\left(\mathbb{R}^{m n_{b}}\right)_{0}^{(m-1) n_{b}}+y\left(X^{\prime \prime}\right)_{(m-1) n_{b}+1}^{\bar{k}} \tag{B.15}
\end{align*}
$$

where $X^{\prime}$ is as in (B.12) and $X^{\prime \prime}$ is any subspace of $\mathbb{R}^{m n_{b}}$ such that

$$
X^{t}+X^{\prime \prime}=\mathbb{R}^{m n_{b}}
$$

The second equality in (B.15) follows from (B.13) and (B.14). Now consider the first $n_{b}$ samples of the state vector of $\left(A_{m}, B_{m}, I, 0\right)$ if the system is excited by $\delta(t)$. These $n_{b}$ samples span a space that satisfies the requirements on $X^{\prime \prime}$ ! This follows from the structure in $A_{m}$ and $B_{m}$ and the fact that

$$
\operatorname{rank}\left[B_{b}, A_{b} B_{b}, \ldots, A_{b}^{n_{b}-1} B_{b}\right]=n_{b}
$$

The interpretation of this result is, that $y\left(X^{\prime \prime}\right)_{(m-1) n_{b}+1}^{\bar{k}}$ can be spanned by the impulse responses $b_{(m-1) n_{b}+1}(t)$ to $b_{\bar{k}}(t)$ shifted forward in time for one to $n_{b}$ samples.

The above result is summarised in the following extension of lemma B.7.

Theorem B. 8 Let $y$ be the set of transients that can be exhibited by any model $G(z)$ of the form

$$
\begin{equation*}
G(z)=\sum_{k=0}^{\bar{k}} \theta_{k} B_{k}(z), \quad \breve{k} \in \mathbb{N} \tag{B.16}
\end{equation*}
$$

where $\left\{B_{k}(z)\right\}$ is a set of basis functions generated by a system $G_{b}(z)$ with state space representation $\left(A_{b}, B_{b}, C_{b}, D_{b}\right)$ and with McMillan degree $n_{b}$. Let
$m$ be the smallest integer such that $\bar{k}<m n_{b}$ and let $b_{k}(t)$ be the impulse response of the $k$ th basis function, then

$$
\begin{aligned}
y= & \left\langle\zeta b_{1}, \ldots, \zeta b_{(m-1) n_{b}}\right. \\
& \left.\zeta b_{(m-1) n_{b}+1}, \zeta^{2} b_{(m-1) n_{b}+1}, \ldots, \zeta^{n_{b}} b_{(m-1) n_{b}+1}, \zeta b_{(m-1) n_{b}+2}, \ldots, \zeta^{n_{b}} b_{\bar{k}}\right\rangle
\end{aligned}
$$

The set of signals spanning $y$ in theorem B. 8 need not be a minimal set of signals spanning $y$. However, the computational effort that is involved in calculating the set of signals spanning $y$ according to theorem $B .8$ seems acceptable, so no additional effort is spent to further reduce the number of signals spanning $y$.

Theorem B. 8 reduces to lemma B. 7 if $\bar{k} / n_{b} \in \mathbb{N}$.

## C

## Application of the formal framework: the algorithm of chapter 4

As an example of application of the formalism in chapter 3 , it will be shown how the algorithm of chapters 4 and 5 fits in the formal framework. Only the aspects of the formalism concerned with modelling and uncertainty bounding will be discussed. The algorithm does not require a more specific interpretation for the true system than was given in chapter 3, aspects of control design and implementation are not part of the algorithm.
$W$ and $T D$ The signal space $W$ is assumed to be partitioned as

$$
W=\mathbb{R}^{p} \times \mathbb{R}^{q},
$$

where $\mathbb{R}^{p}$ represents the signal space for the input component $u$ of a $w=$ $(u, y) \in W^{\mathbb{T}}$ and $\mathbb{R}^{q}$ the signal space for the output component $y$.
All signals are interpreted as discrete time signals, so the time set $\mathbb{T}=\mathscr{Z}$.
$W \triangleright$ The data $W$ contains for every data set $d \in \mathcal{D}$ a pair of input measurements and output measurements.

Notation was simplified in the algorithm by assuming that every data set started at $t=0$. This was possible because the data sets will be used to estimate time-invariant models anyway. In the formalism this is not possible for the sake of generality. Therefore the experiment intervals $\mathbb{T}^{d}, d \in \mathcal{D}$ should be reinterpreted as corresponding to the "real" times at which the measurements were taken. All these separate experiment intervals can then be combined into a single "experiment time set" $T^{\prime}$ according to

$$
\mathbb{T}^{\prime}=\bigcup_{d \in \mathcal{D}} \mathbb{T}^{d}
$$

Then $T^{\prime}$ corresponds to the notation used in chapter 3 , the right hand side corresponds to notation used in chapter 4, with the modification mentioned above.
$W$ consists now of a pair of signals $u$ and $y$,

$$
u: \mathbb{T}^{\prime} \rightarrow \mathbb{R}^{p}, \quad y: \mathbb{T}^{\prime} \rightarrow \mathbb{R}^{q}
$$

so that

$$
W=\{(u, y)\}
$$

As extra information concerning $W$ it is known which parts of $W$ belong to different manifestations of $S$ in Q, as was briefly mentioned in section 3.3.5.
$C_{\text {stab }} \triangleright$ The algorithm applies only to stable systems. For stable systems it is known that $C_{S t a b}$ is equal to the null controller.
$Q>$ The set $Q$ consists for the SISO case of all systems whose transfer function can be represented as

$$
Q(z)=\sum_{k=0}^{\infty} \theta_{k} B_{k}(z)
$$

subject to

$$
\forall k \in \mathbb{N}, \quad\left|\theta_{k}\right| \leq \bar{\theta}_{k} .
$$

The generalisation to the MIMO case is straightforward.
It would be more accurate to say that $Q$ contains the input-output behaviours of these systems.
$\|\xi\| \leq d_{\xi} \triangleright$ In the algorithm, all noise effects and disturbances are lumped at the output. The signal $\xi$ contains both inputs and outputs. Lumping all uncertainty at the output can be interpreted as a limiting case of the more general bounds on $\xi$. Let $f_{u}, f_{y} \in \mathbb{R}_{+}$be given and let the norm $\|\cdot\|$ on $\xi$ be defined as

$$
\|\xi\|=\left\|\left(\xi_{u}, \xi_{y}\right)\right\|:=\max \left\{f_{u} \cdot\left\|\xi_{u, i}(t)\right\|_{\ell_{\infty}}, f_{y} \cdot\left\|\xi_{y, j}(t)\right\|_{\varepsilon_{\infty}}\right\}
$$

With this definition we have

$$
\|\xi\| \leq d_{\xi} \Longleftrightarrow\left\{\begin{array}{l}
\left\|\xi_{u}\right\|_{e_{\infty}} \leq d_{\xi} / f_{u} \quad \text { and } \\
\left\|\xi_{y}\right\|_{e_{\infty}} \leq d_{\xi} / f_{y}
\end{array}\right.
$$

so that for $f_{u} \rightarrow \infty$

$$
\|\xi\| \leq d_{\xi} \Longleftrightarrow\left\{\begin{array}{l}
\xi_{u} \rightarrow 0 \\
\left\|\xi_{y}\right\|_{\ell_{\infty}} \leq d_{\xi} / f_{y} .
\end{array} \quad\right. \text { and }
$$

(Obviously, convergence of $\xi_{u}$ is to be interpreted in the topology induced by the $\ell_{\infty}$ norm.) Taking $f_{y}=1$, the norm on $\xi$ defined above reduces therefore in the limit to the $\ell_{\infty}$-norm on the output component $\xi_{y}$ of $\xi$, with the additional requirement that $\xi_{u}=0$. This in turn corresponds to the way in which the algorithm bounds the uncertainty.
If the same value $\bar{e}$ can be used for all $\bar{e}^{d}(t), t \in \mathbb{T}, d \in \mathcal{D}$, the bound on $\xi$ that is used by the algorithm is simply given by

$$
\|\xi\| \leq \bar{e}=: d_{\xi}
$$

If a truly time-dependent bound $\bar{e}^{d}(t)$ is to be used, or a different bound for different data sets, the definition of the norm on $\xi$ can be adapted accordingly.
$\|u\| \leq d_{u} \triangleright$ The algorithm does not use explicitly the bound on the inputs represented by $\|u\| \leq d_{u}$ in the formalism. Therefore there is no implementation for this requirement in the algorithm.
For the application of the algorithm on a physical process, the bound is used implicitly: obviously some constraints on the input signals need to be satisfied to ensure that the linearisation of the process remains a sufficiently accurate approximation of the process.
The bound $\bar{u}$ on past inputs that is used by the algorithm does not correspond to $\|u\| \leq d_{u}$. The purpose of the bound $\|u\| \leq d_{u}$ is stated in the previous paragraph. The purpose of the bound $\bar{u}$ is to bound the effect of unknown initial conditions.
$\mathcal{P}_{2}, \ldots, \mathcal{P}_{N_{\beta}} \triangleright$ The different types of prior knowledge that are used by the algorithm are each represented by a set $\mathcal{P}_{i}$. This includes the aforementioned bound on past inputs, the noise bounds $\bar{e}^{d}, d \in \mathcal{D}$, the cross-covariance bounds and bounds on the static gain or on the complex gain for any frequency. The only type of prior knowledge that is not represented in this way is

$$
\forall_{k} \in \mathbb{N}, \quad\left|\theta_{k}\right| \leq \bar{\theta}_{k} .
$$

This is represented by $\mathcal{Q}$. Because every $\mathcal{P}_{i}$ is a subset of $\mathbb{Q}$, every $\mathcal{P}_{i}$ represents (part of) this prior knowledge as well.
$J_{i d}, \mathcal{M}_{i d}, \mathcal{M}_{\text {set }} \triangleright$ The algorithm does not specify how its nominal model $\hat{G}$ is to be identified, it only requires that the nominal model is linear, timeinvariant. Therefore nothing can be said about $J_{i d}$ and all that can be said about $\mathcal{M}_{\text {set }}$ is that it should be a subset of the set of linear, time-invariant models. The algorithm assumes that there is one nominal model, so $\mathcal{M}_{i d}$ should be a singleton.

In the estimation of auxiliary models, an identification criterion and a model set are used. In the presentation of the algorithm, least squares estimation was mentioned as a possible estimation method for the auxiliary models. For this method, the identification criterion $J_{i d}$ is identical to the sum of squares of the residuals for a certain model $M \in \mathcal{M}_{\text {set }}$. The model set $\mathcal{M}_{\text {set }}$ contains all models $M$ whose transfer function can be represented as

$$
M(z)=\sum_{k=0}^{\bar{k}} \theta_{k} B_{k}(z)
$$

Here too, the set of identified models $\mathcal{M}_{i d}$ is a singleton taken from $\mathcal{M}_{\text {set }}$. $\mathcal{M}_{i d} "+" \Delta \triangleright$ The $\Delta$ used by the formalism corresponds to $\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right)$ used by the algorithm. The interconnection of this uncertainty description and a nominal model $\mathcal{M}_{\text {id }}$, or $\hat{G}$ in the notation of the algorithm, is obtained by

$$
\hat{G} "+"\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right):=\hat{G}+\Delta+\sum_{i=1}^{n_{s}} \delta_{i} A_{i} .
$$

where the addition on the right hand side is to be interpreted as addition of transfer functions or impulse responses. Part of the interpretation of " + " is now determined by the uncertainty bounding: the $A_{i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$ are determined by the algorithm during the uncertainty bounding
$d(\Delta) \leq d_{\Delta} \triangleright$ To avoid confusion of the two different symbols $\Delta$, the $\Delta$ as used by the formalism will be denoted $\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right)$, where these symbols have the meaning as used by the algorithm. The uncertainty bounding function $d(\Delta)$ is consequently denoted

$$
d\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right)
$$

As in chapter $3, d$ is assumed to take its values in a partially ordered set ID.

The uncertainty bounds derived by the algorithm consist of two components. One is a bound on $\Delta$, the other is a set of intervals, one for every $\delta_{i}, i \in \boldsymbol{n}_{\boldsymbol{s}}$, to which that $\delta_{i}$ is constrained. Ignoring the unstructured error $\Delta$ for a moment, we can define

$$
d\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right):=\left(\delta_{1}, \ldots, \delta_{n_{s}},-\delta_{1}, \ldots,-\delta_{n_{s}}\right)^{T}
$$

so that in this case $\mathbb{D}=\mathbb{R}^{2 n_{s}}$. Using moreover the partial ordering

$$
\forall a, b \in \mathbb{R}^{n}, \quad a<b: \Longleftrightarrow \forall i \in n \quad a_{i}<b_{i}
$$

and

$$
\forall a, b \in \mathbb{R}^{n}, \quad a=b: \Longleftrightarrow \forall i \in n \quad a_{i}=b_{i}
$$

on $\mathbb{R}^{n}$ for any $n \in \mathbb{Z}_{+}$, the fact that

$$
\forall i \in n_{s}, \quad \delta_{i} \in\left[\underline{d}_{i}, \bar{d}_{i}\right]
$$

can simply be expressed as

$$
d\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right) \leq\left(\bar{d}_{1}, \ldots, \bar{d}_{n_{s}},-\underline{d_{1}}, \ldots,-\underline{d}_{n_{s}}\right)^{T}
$$

Adding a bound on $\Delta$ can be done in several ways. For the simple case of an $\mathcal{H}_{\infty}$-norm bound on $\Delta$, it suffices to (re)define

$$
d\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right):=\left(\|\Delta\|_{\mathcal{H}_{\infty}}, \delta_{1}, \ldots, \delta_{n_{s}},-\delta_{1}, \ldots, \delta_{n_{s}}\right)^{T} .
$$

If the bound on the $\mathcal{H}_{\infty}$-norm is denoted $d_{\Delta}$, then the bound on both the structured and unstructured errors can be expressed as

$$
d\left(\Delta, \delta_{1}, \ldots, \delta_{n_{s}}\right) \leq\left(d_{\Delta}, \bar{d}_{1}, \ldots, \bar{d}_{n_{s}},-\underline{d}_{1}, \ldots,-\underline{d}_{n_{s}}\right)^{T} .
$$

More detailed bounds on $\Delta$ can be added in a similar way as the structured errors were bounded. Examples of such more detailed bounds are (weighted) impulse response bounds and bounds defining the simple polytopes in the complex plane that are obtained as an intermediate result of the algorithm.
It is interesting to consider what it means that " $\leq$ " is only a partial ordering on $I D$. Because the ordering is not a total ordering, we can not say which of the two situations below is "better:"
case 1
case 2
$\|\Delta\|_{\mathcal{H}_{\infty}} \leq 1.0 \quad\|\Delta\|_{\mathcal{H}_{\infty}} \leq 0.9$
$-0.1 \leq \delta_{1} \leq 0.1 \quad-0.11 \leq \delta_{1} \leq 0.09$
Being able to decide which of the two is best might be interesting for the formulation of an optimisation criterion for the determination of the structured error components $A_{i}$. One could then search explicitly for those structured error "directions" that lead to the tightest uncertainty descriptions in terms of $d_{\Delta}$ and $\underline{d}_{i}, \bar{d}_{i}, i \in n_{s}$.
In principle it can be determined which of the two cases is to be preferred by designing a robust controller for both uncertainty descriptions and by looking which controller performs best. Apart from the fact that a similar problem may now occur in deciding which controller performs best, this would specify an ordering on, in this case, triples $\left(d_{\Delta}, \underline{d}_{1}, \bar{d}_{1}\right)$ that is far too complex to be used in an optimisation criterion.

Apart from the fact that this possibility to specify a total ordering on $\mathbb{D}$ is impractical, it goes beyond the problem of uncertainty bounding. This is recognised by the formalism by only requiring a partial ordering on $\mathbb{D}$. In that case the uncertainty bounding technique is allowed to be unable to decide which of the two cases mentioned above is "best."

## Bibliography

Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Ostrouchov, S. and Sorensen, D. (1995). Lapack Users' Guide, 2nd edn, Society for Industrial and Applied Mathematics, Philadelphia.

Ariaans, L. J. J. M., Damen, A. A. H. and Weiland, S. (1996). A flexible algorithm for structuring and reducing model error bounds, Proceedings of the 35th Conference on Decision and Control, Kobe, Japan.

Berkelaar, M. (n.d.). lpsolve package. in volume 02 of comp.sources.reviewed.
De Vries, D. (1994). Identification of Model Uncertainty for Control Design, PhD thesis, Delft University of Technology, Delft.

Doyle, J. C., Glover, K., Khargonekar, P. P. and Francis, B. A. (1988). Statespace solutions to standard $H_{2}$ and $H_{\infty}$ control problems, Proceedings of the 1988 American Control Conference, Atlanta, pp. 1691-1696.

Doyle, J. C., Glover, K., Khargonekar, P. P. and Francis, B. A. (1989). Statespace solutions to standard $H_{2}$ and $H_{\infty}$ control problems, IEEE Transactions on Automatic Control 34(8): 831-847.

Draper, N. S. and Van Nostrand, R. C. (1979). Ridge regression and JamesStein estimation: Review and comments, Technometrics 21(4): 451-466.

Falkus, H. (1994). Parametric Uncertainty in System Identification, PhD thesis, Eindhoven University of Technology, Eindhoven.

Fialho, I. J. and Georgiou, T. T. (1995). On the $\mathcal{L}_{1}$ norm of uncertain linear systems, IEEE Transactions on Automatic Control 40(6): 1142-1147.

Gevers, M. (1991). Connecting identification and robust control: A new challenge, Proceedings of the 9th IFAC/IFORS Symposium on Identification and System Parameter Estimation, Budapest, Hungary, pp. 1-10.

Gevers, M. (1993). Towards a joint design of identification and control?, in H. Trentelman and J. Willems (eds), Essays on Control: Perspectives in the Theory and its Applications, Birkhäuser, Boston, pp. 111-151.

Golub, G. H. and Van Loan, C. F. (1989). Matrix computations, Johns Hopkins Series in the Mathematical Sciences, 2nd edn, The Johns Hopkins University Press, Baltimore and London.

Goodwin, G. C., Gevers, M. and Ninness, B. (1992). Quantifying the error in estimated transfer functions with application to model order selection, IEEE Transactions on Automatic Control 37(7): 913-928.

Goodwin, G. C. and Salgado, M. E. (1989). A stochastic embedding approach for quantifying uncertainty in the estimation of restricted complexity models, International Journal of adaptive control and signal processing 3: 333356.

Gorter, R. J. A. (1997). Identification of the physical model parameters of induction machines. On-line and off-line approaches, PhD thesis, Eindhoven University of Technology.
$\mathrm{Gu}, \mathrm{G}$. (1994). Suboptimal algorithms for worst case identification in $H_{\infty}$ and model validation, IEEE Transactions on Automatic Control 39(8): 16571661.

Gu, G. and Khargonekar, P. P. (1991). A class of algorithms for identification in $H_{\infty}$, Proceedings of the 30th Conference on Decision and Control, Brighton, England, pp. 634-639.

Gutman, P. O. (1988). Robust certainty equivalence - a new principle for adaptive control, Proceedings of the 1988 IFAC Workshop on Robust Adaptive Control, Newcastle, Australia, pp. 123-128.

Hakvoort, R. G. (1994). System Identification for Robust Process Control, Nominal Models and Error Bounds, PhD thesis, Delft University of Technology, Delft.

Hansen, F. R. (1989). A Fractional Representation Approach to Closed-Loop System Identification and Experiment Design, PhD thesis, Stanford University, Stanford, CA, USA.

Hansen, F. R. and Franklin, G. F. (1988). On a fractional representation approach to closed-loop experiment design, Proceedings of the American Control Conference, Atlanta, GA, USA, pp. 1319-1320.

Hansen, F. R., Franklin, G. F. and Kosut, R. (1989). Closed-loop identification via the fractional representation: Experiment design, Proceedings of the American Control Conference, Pittsburgh, PA, USA, pp. 1422-1427.

Helmicki, A. J., Jacobson, C. A. and Nett, C. N. (1989). $H_{\infty}$ identification of stable LSI systems: A scheme with direct application to controller design, Proceedings of the American Control Conference, Vol. 2, Pittsburg, PA, pp. 1428-1434.

Helmicki, A. J., Jacobson, C. A. and Nett, C. N. (1990a). Identification in $H_{\infty}$ : A robustly convergent, nonlinear algorithm, Proceedings of the 1990 American Control Conference, San Diego, California, pp. 386-391.

Helmicki, A. J., Jacobson, C. A. and Nett, C. N. (1990b). Identification in $H_{\infty}$ : Linear algorithms, Proceedings of the 1990 American Control Conference, San Diego, California, pp. 2418-2423.

Helmicki, A. J., Jacobson, C. A. and Nett, C. N: (1991). Control oriented system identification: A worst-case / deterministic approach in $H_{\infty}$, IEEE Transactions on Automatic Control 36(10): 1163-1176.

Heuberger, P. S. C. (1991). On Approximate System Identification with System Based Orthonormal Functions, PhD thesis, Delft University of Technology, Delft.

Heuberger, P. S. C., Van Den Hof, P. M. J. and Bosgra, O. H. (1995). A generalized orthonormal basis for linear dynamical systems, IEEE Transactions on Automatic Control 40(3): 451-465.

Jolliffe, I. T. (1986). Principal Component Analysis, Vol. XIII of Springer Series in Statistics, Springer, Berlin.

Kosut, R. L. (1988). Adaptive control via parameter set estimation, International Journal of Adaptive Control and Signal Processing 2: 371-399.

Kosut, R. L., Lau, M. K. and Boyd, S. P. (1992). Set-membership identificatioin of systems with parametric and nonparametric uncertainty, IEEE Transactions on Automatic Control 37(7): 929-941.

Livstone, M. M. and Dahleh, M. A. (1995). A framework for robust parametric set membership identification, IEEE Transactions on Automatic Control 40(11): 1934-1939.

Ljung, L. (1987). System Identification: Theory for the User, Prentice-Hall, Englewood Cliffs, New Jersey.

Mäkilä, P. M. and Partington, J. R. (1992). Robust identification of strongly stabilizable systems, IEEE Transactions on Automatic Control 37(11): 1709-1716.

Mäkilä, P. M., Partington, J. R. and Gustafsson, T. K. (1994). Robust identification, in M. Blanke and T. Söderström (eds), Preprints of the 10th IFAC Symposium on System Identification, Vol. 1, Copenhagen, pp. 1.45-63.

Mäkilä, P. M., Partington, J. R. and Gustafsson, T. K. (1995). Worst-case control-relevant identification, Automatica 31(12): 1799-1819.

Milanese, M. and Elia, N. (1993). Worst-case $l_{1}$ system identification using perturbed ARMA models, Proceedings of the International Symposium on Circuits and Systems, IEEE, Chicago.

Murtagh, B. A. (1981). Advanced Linear Programming: Computation and Practice, McGraw-Hill Inc.

Packard, A. and Doyle, J. (1993). The complex structured singular value, Automatica 29(1): 71-109.

Packard, A. K. (1988). What's new with Mu: structured uncertainty in multivariable control, PhD thesis, University of California, Berkeley.

Partington, J. R. (1991). Robust identification and interpolation in $H_{\infty}$, International Journal of Control 54(5): 1281-1290.

Partington, J. R. (1992). Robust identification in $H_{\infty}$, Journal of Mathematical Analysis and Applications 166: 428-441.

Pree, W. (1994). Design Patterns for Object-Oriented Software Development, ACM Press Books.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W. (1991). Object-Oriented Modeling and Design, Prentice-Hall, Englewood Cliffs, NJ.

Schrama, R. J. P. (1991). An open-loop solution to the approximate closedloop identification problem, Preprints of the 9th IFAC/IFORS Symposium on Identification and System Parameter Estimation, Budapest, Hungary, pp. 1602-1607.

Schrama, R. J. P. (1992). Approximate Identification and Control Design, with application to a mechanical system, PhD thesis, Delft University of Technology, The Netherlands.

Schrama, R. J. P. and Van den Hof, P. M. J. (1992). An iterative scheme for identification and control design based on coprime factorizations, Proceedings of the American Control Conference, Chicago, IL, USA, pp. 28422846.

Van den Boom, T. (1993). MIMO system identification for $H_{\infty}$ robust control, PhD thesis, Faculty of electrical engineering, Eindhoven University of Technology.

Van den Hof, P. M. J., Heuberger, P. S. C. and Bokor, J. (1995). System identification with generalized orthonormal basis functions, Automatica 31(12): 1821-1834.

Van den Hof, P. M. J. and Schrama, R. J. P. (1995). Identification and control - closed-loop issues, Automatica 31(12): 1751-1770.

Vicino, A. and Zappa, G. (1996). Sequential approximation of feasible parameter sets for identification with set membership uncertainty, IEEE Transactions on Automatic Control 41(6): 774-785.

Vidyasagar, M. (1985). Control System Synthesis: a Factorization Approach, MIT Press Series in Signal Processing and Control, MIT Press, London.

Wahlberg, B. and Ljung, L. (1992). Hard frequency-domain model error bounds from least-squares like identification techniques, IEEE Transactions on Automatic Control 37(7): 900-912.

Walter, E. and Piet-Lahanier, H. (1990). Estimation of parameter bounds from bounded-error data: a survey, Mathematics and Computers in Simulation 32(5 \& 6): 449-468.

Willems, J, C. (1986). From time series to linear system - part i. finite dimensional linear time invariant systems, Automatica 22(5): 561-580.

Zhou, T. and Kimura, H. (1993). Time domain identification for robust control, Systems and Control Letters 20: 167-178.

Zhou, T. and Kimura, H. (1994). Simultaneous identification of nominal model, parametric uncertainty and unstructured uncertainty for robust control, Automatica 30(3): 391-402.

## Samenvatting

In veel processen worden regelingen toegepast om het dynamisch gedrag van het proces zodanig te veranderen dat het proces in bepaalde zin beter presteert. Moderne technieken voor regelaarontwerp zijn in het algemeen gebaseerd op een model voor het proces. Als de modelonzekerheid of modelfout, d.w.z. het verschil tussen het model en het werkelijke proces, begrensd kan worden en als het regelaarontwerp rekening kan houden met deze begrensde onzekerheid, wordt de aldus verkregen regelaar robuust genoemd. Dit proefschrift behandelt het probleem van het vinden van grenzen voor modelonzekerheid voor toepassing in robuust regelaarontwerp, uitgaande van experimentele data en voorkennis.

Praktische toepassing van modelonzekerheidsgrenzen wordt momenteel bemoeilijkt door het feit dat deze grenzen vaak onrealistisch en onbruikbaar groot blijken uit te vallen. Geanalyseerd is, welke effecten (moeten) bijdragen aan de modelonzekerheidsgrens. Deze effecten kunnen in drie categorieën worden onderverdeeld: (a) modellen die voor robuuste regelingen worden gebruikt zijn lineair, tijd-invariant en hebben een lage orde, het onderliggende proces heeft die eigenschappen niet. (b) experimentele data van een proces geeft een incompleet en onzeker beeld van het procesgedrag t.g.v. eindige experimentduur, eindige bemonsteringsfrekwentie en onbekende externe factoren zoals ruis. (c) bepaalde kennis die over het proces beschikbaar is, kan niet of slechts bij benadering door het nominale model en/of de onzekerheidsgrenzen worden gerepresenteerd. Dit leidt tot conservatisme. De praktische reden om robuuste regelingen toe te passen is voornamelijk om onzekerheid uit categorie (a) het hoofd te kunnen bieden. Huidige modelonzekerheidsgrenzen begrenzen slechts de onzekerheid ten gevolge van categorie (b) en (c).

Een algemeen raamwerk is ontwikkeld om de interactie te onderzoeken tussen verschillende factoren die een rol spelen in een procedure voor onzekerheidsbegrenzing. In tegenstelling tot andere kaders waarin identificatieprocedures worden "ingebed" vermijdt dit kader zorgvuldig dat onrealistische aannames voor het proces of de ruis gedaan worden. Enkele eigenschappen die alle procedures voor onzekerheidsbegrenzing gemeen (zouden moeten) hebben worden onderzocht. Bovendien wordt de relatie verduidelijkt die er moet bestaan tussen ruis, verstoringen en vereenvoudigingen zoals lineariteit en tijd-invariantie aan de ene kant en de onzekerheidsgrenzen aan de andere kant.

Naar aanleiding van de opmerkingen in het bovenstaande wordt een algoritme voorgesteld dat modelonzekerheid voor MIMO systemen opsplitst in zogenaamde gestructureerde en ongestructureerde componenten. De gestructureerde component wordt gedetailleerd begrensd en dient om de veranderingen in procesdynamica te beschrijven die optreden als het proces in verschillende
werkpunten wordt bedreven. In de ongestructureerde component worden alle andere bronnen van modelonzekerheid samengevoegd en op een veel minder gedetailleerde manier begrensd. Als de gestructureerde componenten de dominante factoren in de procesonzekerheid beschrijven, zal de resterende ongestructureerde component veel kleiner uitvallen dan zonder een afzonderlijke begrenzing van de dominante factoren het geval zou zijn. Bovendien omvatten de zo verkregen onzekerheidsgrenzen meer aspecten van modelonzekerheid die voor robuuste regelingen relevant zijn dan tot dusver vaak het geval was.

De dominante bijdragen aan de modelonzekerheid kunnen geschat worden uit data sets voor verschillende werkpunten. Er wordt afgeleid hoe deze bijdragen uitgesloten kunnen worden van de grenzen op de ongestructureerde fout. Deterministische aannames worden gedaan voor de ruis en verstoringen die op het proces ingrijpen. Verschillende uitbreidingen op het basisalgoritme zijn ontwikkeld om het gebruik van meer voorkennis mogelijk te maken bij het begrenzen van de modelonzekerheid en het bepalen van de dominante factoren.

Het algoritme is geïmplementeerd in een C++ programma. Na een zorgvuldige overweging van een implementatie in Matlab is geconcludeerd dat MatLAB onvoldoende ondersteuning biedt. Een algemeen toepasbare bibliotheek is ontwikkeld voor de implementatie. Uit eenvoudige objecten als vectoren en verzamelingen van gehele getallen worden complexere objecten samengesteld, zoals signalen, modelsets, identificatiesessies, etc. Hoewel deze hulpmiddelen werden ontwikkeld voor de implementatie van het algoritme, hebben ze hun eigen waarde op het gebied van computerondersteunde systeemidentificatie.

Het algoritme is getest op simulatiedata en op praktijkdata van een asynchrone machine. Uit de resultaten kan worden geconcludeerd, dat het algoritme inderdaad een significante reductie van de ongestructureerde onzekerheid kan opleveren door een beperkt aantal gestructureerde componenten af te splitsen die door het algoritme zijn bepaald. Bovendien kan het algoritme een benadering vinden voor foutstructuren die het niet exact kan representeren. Dit leidt opnieuw tot een duidelijke afname van de ongestructureerde fout.

Zowel de grenzen op de gestructureerde als op de ongestructureerde componenten zijn erg gevoelig voor de manier waarop de invloed van ondermodellering wordt begrensd. Bovendien is data van hoge kwaliteit vereist om zinvol onderscheid te kunnen maken tussen gestructureerde en ongestructureerde bijdragen aan de totale onzekerheid. De signaal/ruisverhouding lijkt tenminste 40 dB te moeten zijn voor een succesvolle toepassing van het algoritme. Aangezien de ruis dit niveau overschrijdt in het voorbeeld van de asynchrone motor, vermindert afsplitsen van gestructureerde componenten hier nauwelijks de ongestructureerde fout. Dit is voornamelijk te wijten aan de deterministische (worst-case) aannames op de ruis. Het verdient aanbeveling in toekomstig onderzoek het gebruik van stochastische aannames op de ruis te onderzoeken. Dit kan resulteren in kleinere, maar zwakkere, foutengrenzen, die echter beter praktisch toepasbaar lijken voor realistische signaal/ruis verhoudingen.

## Curriculum vitae

November 4, 1968 Born in Thorn, the Netherlands.
September 1981 - Secondary school: Gymnasium $\beta$, Scholengemeenschap
June 1987 St. Ursula, Horn, the Netherlands.
September 1987 - Master student at the Eindhoven University of Technol-
January 1993 ogy, Department of Electrical Engineering, Eindhoven, the Netherlands; with
April 1991 - Practical training at the Université des Sciences et TechJuly 1991 niques du Languedoc CEM, Montpellier, France: Experiments and theories on low-frequency noise in GaAs resistors under supervision of F. Pascal, M. de Murcia and prof. G. Lecoy.
January 1992 - Master project at the Unilever Research Laboratories,
January 1993 Vlaardingen, the Netherlands: Specific growth rate control for fed-batch fermentation of bakers' yeast under supervision of dr.ir. A.R.M. Soeterboek, dr. ir. M. Keulers and prof. dr.ir. P. Eykhoff.
January 1993 - $\quad$ Ph.D. research at the Measurement and Control group,
January 1997 Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, the Netherlands: Structure in practical model error bounds under supervision of dr.ir. A.A.H. Damen, dr. S. Weiland and prof. dr. ir. A.C.P.M. Backx.
January 1997 - Process engineer; AspenTech Europe B.V., Best, the Netherlands.

# SUMMARY <br> of the algorithm presented in <br> chapter 4 of the Ph.D. thesis 

## Structure in Practical Model Error Bounds

by<br>Leon Ariaans

Eindhoven, February 26, 1997

## Algorithm

- Do experiments
$-\mathcal{W}, \mathcal{D}(w)$ : sets of operating points and (prospective) data sets
$\rightarrow u^{d}(t), y^{d}(t)$ : input/output measurements, data set $d$
$\longrightarrow v^{d}(t), r_{\ell}^{d}(t)$ : instruments for identification/cross-covariance bound
Gather prior information p. 80
$\longrightarrow \vec{u}:$ input amplitude bound $\quad$ p. 82
$\longrightarrow \bar{e}(t):$ noise amplitude bound $\quad$ p. 82
$\rightarrow \bar{c}_{k}:$ cross-covariance bound p. 83
$\longrightarrow \bar{\theta}_{k}, M, p$ : model parameter bounds p. 82
Choose basis functions and model orders ..................p. 88
$\longrightarrow b_{k}(t)$ : set of basis functions p. 75
$\longrightarrow \bar{k}:$ model order p. 77
Estimate auxiliary models ......................................... 91
$\vdash^{<}<u(t), y(t), v(t)$, model parametrisation
$\longrightarrow \theta^{w}$ : parameters of auxiliary model for operating point $w$ p. 92
Estimate error structure .......................................... 101
- $<\Delta \theta^{w}=\theta^{w}-$ mean $_{w w^{\prime}} \theta^{w^{\prime}} \quad$ p. 105
$\rightarrow X:$ matrix with in its columns the $\Delta \theta^{w} \quad$ p. 102
$\rightarrow U \Sigma V^{T}:$ singular value decomposition of $X$
$\rightarrow n_{s}:$ number of structured error components p. 105
$\longrightarrow \theta_{A, i}=U_{* i}, i \in n_{s}:$ parameters of structured errors $A_{i} \quad$ p. 105
Construct set of linear constraints ............................... 106

$\vdash^{<} \bar{e}(t)$ : (unextended) noise bound p. 82
$\rightarrow \hat{a}(t):$ influence of basis $\bar{k}+1$ to $\tilde{k}$, no transient p. 107
$\rightarrow \hat{b}(t):$ transient of basis 0 to $\tilde{k} \quad$ p. 107
$\longrightarrow \delta$ : influence of basis $\tilde{k}+1$ and higher, incl. transient p. 108 $\longrightarrow\left|y(t)-G_{\theta}(\zeta) u(t)\right| \leq \bar{e}_{e}(t):=\tilde{e}(t)+\hat{a}(t)+\hat{b}(t)+\delta \quad$ p. 108
Extend cross-covariance bound .............................. p. p. 110
- $<\bar{c}_{\ell,}, r_{\ell}(t)$ : (unextended) cross-covariance bound p. 83
$\rightarrow d(\ell)$ : basis function $\bar{k}+1$ to $\hat{k}$, known inputs p. 111
$\rightarrow f(\ell)$ : basis function $\bar{k}+1$ to $\tilde{k}$, inputs $u(-\hat{t})$ to $u(-1)$ p. 111 $\rightarrow \delta_{1}(\ell)$ : basis function $\bar{k}+1$ to $\bar{k}$,
inputs $u(-\tilde{t}-1)$ to $u(-\infty)$
p. 111
$\rightarrow \delta_{2}(\ell)$ : basis function $\tilde{\tilde{k}}+1$ and higher, all inputs p. 111
$\longrightarrow\left|\sum_{t} r_{\ell}(t)\left(y(t)-G_{\theta}(\zeta) u(t)\right)\right|$

$$
\begin{equation*}
\leq d(\ell)+f(\ell)+\delta_{1}(\ell)+\delta_{2}(\ell)+\bar{c}_{\ell} \sqrt{T^{d}} \tag{p. 112}
\end{equation*}
$$

$\longrightarrow \mathcal{L}_{\theta}^{w}$ : set of linear constraints on $\boldsymbol{\theta}$ for every operating point

- Bound errors locally
- Split parameter uncertainty in structured and unstructured parts
$\vdash^{<} \hat{G}, \hat{\theta}$ : nominal model, vector of its first $\bar{k}$ expansion coefficients in basis $b_{k}$
$\rightarrow\left(\theta_{A, i} \mid \theta^{\prime}-\hat{\theta}\right) \theta_{A, i}$ : ith structured component for arbitrary $\theta^{\prime}$
$\rightarrow P$ projection onto span of $\boldsymbol{\theta}_{A, i} \quad$ p. 128
$\rightarrow P^{\perp}:$ projection onto orthoplement of span of $\boldsymbol{\theta}_{A, i} \quad$ p. 128
$\rightarrow P^{\perp}\left(\hat{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right)$ : unstructured error component in $\boldsymbol{\theta}^{\prime}$
$\hat{G}^{\perp}=\hat{G}-G_{P \hat{g}}:$ nominal model with structured error directions removed
p. 126
$\square$ Bound structured errors............................................ 113
$\longrightarrow \bar{d}_{i}^{w}:=\max _{\theta^{\prime} \in \mathcal{L}_{\theta}^{w}}\left(\theta_{A, i} \mid \theta^{\prime}-\dot{\theta}\right) \theta_{A, i}:$
upper bound on ith structured component p. 114
$\longrightarrow \underline{d}_{i}^{w}:=\min _{\boldsymbol{\theta}^{\prime} \leq \mathcal{L}_{\boldsymbol{\theta}}^{\alpha}}\left(\boldsymbol{\theta}_{A, i} \mid \boldsymbol{\theta}^{\prime}-\hat{\boldsymbol{\theta}}\right) \boldsymbol{\theta}_{A, i}:$
lower bound on th structured component p. 115
Bound unstructured error in frequency domain ..... p. 116
$\vdash<e^{j \phi_{t}}, \ell=1, \ldots, m$ : set of directions in complex plane p. 117
$-\Omega=\left\{\omega_{i}\right\}$ : discrete set of frequencies p. 117
$\rightarrow \hat{\mu}_{\ell}^{\prime}\left(\omega_{i}\right)=\max _{\theta^{\prime} \in \mathcal{L}_{\theta}^{\prime \prime}} \operatorname{Re}\left(\left(G_{P+\theta^{\prime}}\left(e^{j \omega_{i}}\right)-\hat{G}^{\perp}\left(e^{j \omega_{i}}\right)\right) e^{-j \phi_{i}}\right)$
uncertainty in direction $e^{j \phi_{i}}$ for frequency $\omega_{i}$ due to param. uncertainty not covered by struct. error p. 128
$\rightarrow \tilde{\mu}\left(\omega_{i}\right):$ error due to tail of basis p. 118
$\rightarrow \overline{\mathcal{P}}_{m}^{\prime}\left(\omega_{i}\right)$ : uncertainty region in complex plane determined by $\hat{\mu}_{\ell}\left(\omega_{i}\right)$ and $\bar{\mu}\left(\omega_{i}\right)$
p. 128
— Bound the interpolation error for $\omega \notin \Omega \quad$ p. 130
Bound unstructured error in time domain ........... p. p. 134
$-<W(\zeta)$ : stable weighting filter or $z /(z-1) \quad$ p. 134

| - | $t_{\text {max }}:$ time after which constant bounds are used |
| :--- | :--- |
| $\longrightarrow s(t):$ upper bound for $W(\zeta) \Delta(t)$ | p. 134 |

$\rightarrow \underline{s}(t)$ : lower bound for $W(\zeta) \Delta(t) \quad$ p. 139
Combine local results............................................. 140
$\rightarrow \bar{d}_{i}:=\max _{w \in w} \tilde{d}_{i}:$ upper bound for $i$ th structured component
$\longrightarrow \underline{d}_{i}:=\min _{w \in w} \underline{d}_{i}:$ lower bound for $i$ th structured component
Take union of all unstructured bounds

## STELLINGEN

behorende bij het proefschrift

# Structure in Practical Model Error Bounds 

Leon Ariaans

Eindhoven, 26 februari 1997

1. Het is grotendeels schijn dat de aansluiting tussen de vakgebieden robuust regelaarontwerp en modelonzekerheidsbegrenzing de laatste jaren verbeterd is.
Hoofdstuk 2 van dit proefschrift.
2. Het schatten van structuur in modelonzekerheid gebaseerd op fysische voorkennis van het te modelleren proces kan een bijdrage leveren aan het verminderen van conservatisme in modelonzekerheidsgrenzen.
Hoofdstuk 4 t/m 6 van dit proefschrift.
3. De Matlab programmeertaal en de implementatie daarvan zijn niet geschikt voor software projecten die een omvangrijke programmeerinspanning vergen.
Sectie 7.1 van dit proefschrift.
4. Voor succesvolle toepassing van black-box identificatietechnieken kan het te identificeren proces niet als een black box worden opgevat.
5. Een procedure voor robuust regelaarontwerp die een uniforme kansverdeling over de hele set van procesonzekerheden kan hanteren zou minder conservatieve resultaten geven dan de huidige worst-case technieken, daar deze huidige technieken de kansverdeling van de onzekerheden concentreren in die onzekerheden die de slechtst mogelijke prestaties geven.
6. Dynamica in de stapresponsie van een proces is als een kater. Je moet even wachten en dan gaat het meestal vanzelf over. Wie daar geen genoegen mee wil nemen haalt er een doctor bij.
7. De argumenten van economische noodzaak die worden aangevoerd voor het aanleggen van de Betuwelijn maken duidelijk dat het spoor geen oplossing biedt voor het file-probleem.
8. Het voorschrijven van een standaard voor de typografie van enig onderdeel van een op zichzelf staande publicatie doet de schrijver van het betreffende werk tekort.
9. De opvatting, gebezigd in Orwell's "Nineteen eighty-four," dat begrippen waarvoor iemand geen woord heeft uit de gedachtenwereld van die persoon verdwijnen, is onjuist.
G. Orwell, Nineteen eighty-four : a novel (1949)

Dit proefschrift, de set $Q$ in hoofdstuk 3
10. De term niet-parametrische modellen is misleidend. Modellen van deze soort bevatten doorgaans juist meer parameters dan andere veel gebruikte modellen.
11. Promovendi zouden in het laatste half jaar voor hun promotie vrijstelling moeten krijgen van het betalen van kijk- en luistergeld.

