KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELECTROTECHNIEK
Afdeling ESAT
Kardinaal Mercierlaan 94
3030 Leuven (Heverlee)

A NEW CLASS OF ALGORITHMS IN LINEAR SYSTEM THEORY

With application to real-time aircraft model identification

Jury :
Voorzitter Prof. J. Delrue, Vice-dekaan
Prof. J. Vandewalle, promotor
Prof. A. Oosterlinck
Prof. J. Peperstraete
Dr. P. Van Dooren (PRLB)
Prof. A. Haegemans
Dr. A. Barbe

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen
door

MICHEL H. VERHAEGEN

U.D.C. 629–7–016

November 1985

Preface

The field of numerical analysis and system theory are both very young. Numerical analysis dates from the first attempts to solve mathematical problems on the first computers of the early 50's. Modern system theory is even younger, going back to 1960. In the 20 to 30 years since that time both fields have grown into mature disciplines. For numerical analysis in particular, problems in many areas of applied mathematics have been studied from a computational, numerical point of view, resulting in a large body of theories. However, the interdisciplinary approach between numerical analysis and system theory is still in its infancy. This thesis attempts to contribute to the development of this young but very interesting research area.

This thesis would not exist today if it wasn't for the help of many people. The first and foremost of them is without question **Paul Van Dooren**. His broad field of interest and enthusiastic support have been invaluable for the successful completion of this research. Working with him has thaught me how to use the instruments of numerical analysis in order to tackle the different problems considered in this thesis. I shall always be grateful for the inspiration and personal education he gave me.

I am also indebted to Prof. J. Vandewalle for his suggestions and stimulating remarks, which have contributed to the success of this thesis. The opportunity to work in the group of system theory at the departement of Electrical Engineering has been very stimulating for me and for my technical work. The nice atmosphere created by the people in that group, Bart De Moor, Jan Vanderschoot, Sabine Van Huffel, Tan Shaohua and Dirk Callaerts has prevented me from shutting myself up with my aeronautical problems. I particularly would like to thank Bart De Moor for our interesting discussions. Thanks are also due to M. Cumps and F. Bril for their help with some of the programming.

Equally important are the practical criticism and interest of a number of industrial and academic people in connection with the results obtained. In this respect I may mention A.J. Mulder and Prof. O.H. Gerlach from the Delft University of Technology, H.P. Blom of the Dutch Aerospace Laboratory NLR, R. De Keyser from the University of Ghent (RUG) and H. Maes from the Belfotop-Eurosense company.

I want to thank the IWONL of Belgium for the grant I received the past three years for doing research in aircraft model identification. Without this grant it would have been much harder, if not impossible, to complete this thesis. I also want to thank the NFWO of Belgium for the travel funds I was granted. They allowed me to discuss my work at several conferences, and to receive some valuable feedback.

I am very grateful to my father for dealing with the graphical layout in this thesis and also for the endless support and encouragements during the years of doctoral research. His warm friendship and that of his family combined with the indulgence of my little wife Hilde has turned the difficult task of writing this thesis in a feasible charge.

CONTENTS

Introduction v

Outline of the thesis ix

1. Real-time aircraft model identification 1
- 1.1. Introduction . . . 1
- 1.2. The operation of present FMS . . . 4
  - 1.2.1. The general architecture of a FMS . . . 4
  - 1.2.2. Optimal path definition methods . . . 7
    - 1.2.2.1. A very simple approach . . . 7
    - 1.2.2.2. A parameter optimization method . . . 9
    - 1.2.2.3. A global optimization scheme . . . 10
- 1.3. Aircraft identification: The classical approach . . . 14
  - 1.3.1. The problem statement . . . 14
  - 1.3.2. The mathematical aircraft model . . . 15
  - 1.3.3. The one-step identification method . . . 18
- 1.4. The two-step identification method . . . 19
  - 1.4.1. The mathematical formulation . . . 19
  - 1.4.2. Remarkable system properties . . . 22
- 1.5. The perspectives of the two-step method . . . 24
- 1.6. The description of the research goals . . . 27

2. Condensed forms in linear system theory 29
- 2.1. Introduction . . . 29
- 2.2. Review of basic orthogonal transformations . . . 32
  - 2.2.1. Elementary permutations . . . 32
  - 2.2.2. The Givens transformation . . . 33
  - 2.2.3. The Householder reflection . . . 35
- 2.3. Condensed state-space models . . . 37
- 2.4. Unification of using condensed forms . . . 43
  - 2.4.1. Some applications in numerical analysis . . . 43
    - 2.4.1.1. Updating Schur forms . . . 44
    - 2.4.1.2. Performing an implicit QR step . . . 44
    - 2.4.1.3. Reduced QR decomposition . . . 46
  - 2.4.2. A number of applications in system theory . . . 46
    - 2.4.2.1. Linear state variable feedback . . . 46
    - 2.4.2.2. Lyapunov and Riccati equations . . . 49
    - 2.4.2.3. Transfer function, poles and zeros . . . 50

    - 2.4.2.4. Frequency and system response . . . 55
  - 2.4.3. State observation with a Luenberger observer . . . 56
    - 2.4.3.1. The full order observer . . . 56
    - 2.4.3.2. The reduced order observer . . . 59
- 2.5. Generalized state space descriptions . . . 67
  - 2.5.1. Condensed generalized state space models . . . 67
  - 2.5.2. The generalized reduced order observer . . . 70
- 2.6. Concluding remarks . . . 73

3. A new approach towards Kalman filtering 75
- 3.1. Introduction . . . 75
- 3.2. Notation and preliminaries . . . 77
- 3.3. The discrete Kalman filter problem . . . 78
- 3.4. A general covariance representation . . . 80
- 3.5. A general, recursive Kalman filter algorithm . . . 83
  - 3.5.1. The new problem formulation . . . 83
  - 3.5.2. A recursive solution . . . 84
- 3.6. Two extremes of the general algorithm . . . 91
  - 3.6.1. The Square Root Covariance Filter . . . 91
  - 3.6.2. The Square Root Information Filter . . . 94
- 3.7. Concluding remarks . . . 96

4. The Reliability and efficiency of Kalman filtering 99
- 4.1. Introduction . . . 99
- 4.2. The four different Kalman filter implentations . . . 101
  - 4.2.1. The Square Root Covariance Filter . . . 102
  - 4.2.2. The Square Root Information Filter . . . 103
  - 4.2.3. The Conventional Kalman Filter . . . 105
  - 4.2.4. The Chandrasekhar Square Root Filter . . . 106
- 4.3. Theoretical error analysis . . . 108
  - 4.3.1. Round-off errors . . . 108
    - 4.3.1.1. Propagation of a single error . . . 109
    - 4.3.1.2. The local error . . . 114
    - 4.3.1.3. The total error . . . 117
  - 4.3.2. Modeling errors . . . 118
  - 4.3.3. Discretization errors . . . 120
- 4.4. New efficient Kalman filter implementations . . . 124
- 4.5. Comparison of the different filters . . . 128
- 4.6. Concluding remarks . . . 129

5. The flight-path reconstruction module 131
5.1. Introduction . . . 131
5.2. Experimental evaluation of the different Kalman filters . . . 133
5.2.1. Experimental set-up . . . 134
5.2.2. Comparing the SRCF/SRIF with the CKF . . . 135
5.2.3. Comparing the SRCF/SRIF with the CSRF . . . 138
5.2.4. Comparing the SRCF with the SRIF . . . 141
5.2.5. Modeling and discretization errors . . . 144
5.2.6. Computational complexity . . . 148
5.3. The adaptive flight-path reconstruction module . . . 149
5.3.1. Real-time convergence control of the SRCF . . . 149
5.3.2. Dealing with time-varying zero bias errors . . . 152
5.3.2.1. The extended state space description . . . 154
5.3.2.2. The algorithm . . . 156
5.3.2.3. Adaptive tracking capabilities . . . 161
5.3.2.4. Efficient implementation . . . 165
5.4. Algorithmic synthesis . . . 166
5.5. Concluding remarks . . . 168

6. The aerodynamic model identification module 169
6.1. Introduction . . . 169
6.2. The aerodynamic model identification problem . . . 170
6.3. Least squares estimation of time-varying parameters . . . 173
6.3.1. The least squares method . . . 173
6.3.2. Difficulties with real-time RLS estimation . . . 179
6.4. Computational aspects of different RLS methods . . . 183
6.4.1. The different RLS implementations . . . 183
6.4.1.1. The conventional RLS method . . . 183
6.4.1.2. The square root cov. RLS method . . . 183
6.4.1.3. The square root inf. RLS method . . . 184
6.4.2. Theoretical analysis of the reliability . . . 185
6.4.3. Comparison of the efficiency . . . 189
6.4.4. Experimental evaluation . . . 190
6.5. Real-time aircraft model identification . . . 194
6.5.1. Checking the persistent excitation condition . . . 194
6.5.1.1. The theoretical framework . . . 194
6.5.1.2. The algorithmic design . . . 196
6.5.1.3. The experimental evaluation . . . 198
6.5.1.4. Model reduction . . . 200

6.5.2. Accurate dealing with time varying parameters . . . 202
6.6. Algorithmic synthesis . . . 204
6.7. Concluding remarks . . . 206

7. The conclusions 207
7.1. Review of the obtained results . . . 207
7.2. Main conclusion . . . 212
7.3. Suggestions for further research . . . 213

Appendices

A. Linearization of the aircraft's equations of motion 217
A.1. Outline of the linearization procedure . . . 217
A.2. Linear state equations . . . 219
A.3. Linear observation model . . . 224

B. The skew Householder transformation 225

C. Error analysis of some linear algebra problems 227

D. Summary (in Dutch) 231
D.1. Introductie . . . 231
D.2. Formulering van het vliegtuig identificatie probleem . . . 232
D.3. Concrete doelstellingen . . . 232
D.4. Reconstructie van de vliegbaan . . . 234
D.4.1. Het gebruik van gecondenseerde vormen . . . 235
D.4.2. Het gebruik van het Kalman filter . . . 236
D.4.3. Het globale algoritme . . . 237
D.1. Identificatie van het aerodynamisch model . . . 240
D.1. Hoofdconclusie . . . 242
D.1. Suggesties voor verder onderzoek . . . 244

References 247

Introduction

The research reported in this thesis lies in the area of *linear system theory*, a field of science that has recently developed itself into a general support for a broad class of scientific disciplines, such as automatization, signal processing, medicine, biology, economics, etc. and this especially to acquire or to enlarge insight in these specific application areas.

The foundation of this evolution lies in the "system" approach, often called the "black-box" approach. This allows to tackle the specific problems of these different disciplines in *a unified manner*. Apart from gaining insight, also (if not more) important is the ability to come forward with practical applicable solutions to certain problems in the different fields mentioned above. Historical evolution has demonstrated that many theoretical concepts in system theory were developed when a need arose from practical applications. And this often happened to be problems in the Aeronautical field. Examples are the least mean squares method developed to determine planet orbits, the optimal control problem to minimize fuel consumption during rocket launching, etc. The developments in linear (multivariable) system theory have proceeded essentially along two lines which correspond to different ways of considering a linear system: from the point of view of internal system description, *the state space approach*, or from the point of view of external system description, *the transfer function approach*. The last 2 decades considerable progress has been made in both approaches, and this as well as conceptually as computationally. Hence one often prefers a combined approach for the analysis, synthesis and simulation of a linear system. This supposes that the system representations taken as a starting point in both approaches, have to be transformable to one another.

In this thesis, the main emphasis is put on the the *state space approach*, with brief attention to the transformation of the state space model to the transfer function description. The state space model, which forms the basis in this approach, can be obtained in 2 different ways:

1. Based on the physical laws describing the behaviour of the system, e.g. the laws of Newton, thermodynamics, etc. Often the resulting set of differential equations is *non-linear*, so that linearization is necessary in order to obtain a linear model.

2. Based on input and output measurements and then using the (minimal) realization theory as developed by Ho, Kalman [40], Silverman [89], etc.

For situations where a lot of a priori information about the system is available, such as the possible choice of state variables, the order of the state space model, etc. often the first method is chosen. And this is certainly the case for the application that will be analyzed throughout this thesis, namely the determination of the aircraft model. Following the first method, the resulting state space model of the aircraft contains a number of parameters that have an important physical interpretation. So that their knowledge is extremely interesting in the design of aircraft control systems, the judgement of aircraft performances, etc. However, the disadvantage is that these parameters are unknown and have firstly to be determined by *system identification techniques*. This is of course not a trivial task (see e.g. in the book of P. Eyckhoff [23]). It is here that a first important aspect of solving system theoretical problems arises, namely *the formulation of the problem*. It will be shown in the first chapter of this thesis that a reformulation of the aircraft model identification problem, according to a method developed by O.H. Gerlach [30], allows to eliminate the lack of information about the state space model.

Based on a given state space model a large number of tools have been developed which make a profound analysis, synthesis and simulation of the linear system under consideration possible. The developed algorithms are often based on the concepts of *numerical linear algebra*. However, many proposed algebraic solutions are numerically not useable, or only useable for a restricted class of system models. This is due to the fact that the numerical results no longer satisfy the algebraic rules. A nice example is the state error covariance matrix in the Kalman filter problem that may become negative definite. In many cases, one has tried to "explain" this discrepancy by the "wrong" way the problem was formulated. Hence one choses to adapt the problem formulation in order to provide a solution. This reformulation was often based on *heuristical grounds*, using the so-called "engineering-insight". Examples are legio: to name only a few, there is the increase of input noise to overcome the above mentioned Kalman filter

problem, discussed e.g. in [86], and the modified Newton-Raphson method to overcome convergence problems in the Maximum Likelihood identification method [35]. This way of approaching system theoretical problems gave rise to an extensive list of variation of algorithms. This may have a discouraging effect on both experienced and unexperienced users of linear system theory, since the heuristic grounds do not allow to formulate sharp guidelines on how to use the algorithm for a specific experimental condition. Furthermore, the resulting algorithms have the following drawback: they are not suited to operate self-adaptive, they often increase the computational complexity and they deteriorate the accuracy of the calculated results.

Numerical analysts who became involved in system theoretical problems have found out that the mismatching of the numerical results and the algebraic rules is not always due to the wrong problem formulation. They pointed out that also the aspect of *sensitivity* of the numerical implementation to round-off errors has to be accounted for. This way of approaching system theoretical problems nowadays gradually becomes accepted as an *independent element* that plays an important role in solving problems in system theory. This will certainly lead to an increased confidence in the *use of linear system theory*. Recent experience has demonstrated (see e.g. the work of P. Van Dooren [110]), that taking care of this aspect can really lead to *generally applicable algorithms*, allowing to use these algorithms for all possible conditions in the problem formulation. This algorithmic property is often called *robustness*, or *reliability*. In numerical analysis, a more refined indication is used, namely (forward and/or backward) *numerical stability* [46]. An example that allows to use such reliable algorithms, implemented for a great number of system theoretical problems, in a flexible, usersfriendly and interactive way is the MATLAB-SC package [111]. Extensive use of this package during this research has demonstrated that such a tool is essential to gain fast insight in system theoretical problems and to come to a solution. Apart from that, this package is very instructive to introduce the unexperienced user in system theory.

From the discussion given above it may become clear that the reliability of algorithms is an absolute necessity when working during system operation, i.e. for *real-time* (or on-line) *applications*. It has become a popular trend, recently, to consider such applications, especially because it allows to further improve or "optimize" the system operation. The development of the algorithms in this thesis fits into this practical demand. However, for this application reliable implementations often have the disadvantage to be

(much) more inefficient than their not-reliable counterparts. This drawback clearly plays an important role when we restrict to the present generation of *digital sequential computers*.

> **A first contribution** of this thesis is the development of new algorithmic implementations which combine the *contradictory* requirements of efficiency and reliability and this for the present generation of computers. Here, we will make use of the so-called "*condensed system representation*", which originates from numerical analysis.

A second important item for real-time applications is the validity of the algorithmic results. Indeed, for a number of problems in system theory there are still no general applicable algorithms and the *validity* (the domain of applicability) of the different existing algorithmic implementations for the same problem are often *unknown*. The latter is the case for the *linear state reconstruction* problem, which is a subproblem in the aircraft model identification problem. Until now, one tried to answer this validity problem by performing simulation tests for all possible conditions that (may) occur during real-time operation and this for all considered numerical implementations. It is evident that this is a very time consuming affair. Moreover, on the one hand it gives *no absolute certainty* and on the other hand it does not imply that the obtained conclusions can be carried over to other "similar" systems.

> In a **second contribution** of this thesis the question of validity is answered for four different Kalman filter implementations and three different recursive least squares methods by performing a *theoretical error analysis*. Here, it will be shown that extensive use is made of the experience acquired in numerical analysis.

A third important aspect that needs to be analyzed for every specific real-time application are the practical problems that (may) occur. Apart from being reliable and efficient, the solutions proposed here must also be *automatically adaptable*. And this while assuring *maximal accuracy* of the calculated results.

> The derivation of such solutions for different practical problems in the aircraft identification problem and the experimental evaluation of their performance using simulated flight test data form the **third main contribution**

of this thesis.

Outline of the thesis

In the first **introductory chapter** of this thesis the aircraft model identification problem is outlined. Attention is given to the increasing interest in solving this problem *real-time*. After a discussion of some aeronautical aspects of (real-time) aircraft model identification, we show how the general non-linear system identification problem can be reformulated into two simple linear estimation problems. The driving force behind this research has been the aim to solve these two subproblems in an efficient, reliable and accurate way. The reader only interested in the system theoretical results of this thesis can restrict his reading of this first chapter to sections 1.3 and 1.4, which describes both subproblems, and section 1.6 in which the specific goals pursued in this thesis are described. With this information the reader can understand the system description that will be used throughout this thesis as an application of the obtained results.

Chapter 2 treats the use of the "condensed state space representation" and shows how to use them to unify efficiency and reliability for a broad class of problems in linear system theory. A brief overview of its application is given in:

- The linear state feedback problem.
- The solution of Lyapunov equations.
- The determination of the transfer function, poles and zeros.
- Frequency- and system response.

A more extensive treatment is given of the use in the *state reconstruction problem* by means of a Luenberger observer. Here we highlight some new results in the solution of the "*generalized* Luenberger observer problem".

Chapter 3 discusses the solution of the so-called "optimal state reconstruction problem". This is indicated as the Kalman filter. In contrast with the classical "statistical" approach to derive that solution, we preferred a more instructive derivation based on the "generalized least squares problem". This approach is used to point out a broad class of *existing* and *new* filter implementations.

In **chapter 4** we expose the *theoretical error analysis* of four different Kalman filter algorithms, namely:

- The conventional Kalman filter.
- The square root covariance filter.
- The square root information filter.
- The Chandrasekhar square root filter.

Models are developed from this analysis that describe the propagation of errors due to round-off, modelling errors and discretization errors for the above mentioned implementations.

Chapters 5 and 6 concentrate on a number of practical problems which (may) occur when both subproblems defined in chapter 1 are solved in real-time. For the first subproblem, called the *flight-path reconstruction* problem, the following items have been analyzed:

- the *automatic determination* of the convergence of the chosen recursive Kalman filter implementation,
- the *automatic treatment* of (time-varying) zero bias errors on the input signals of the Kalman filter model,
- the tracking of the *time-varying* aircraft dynamics,
- the reliability of the Kalman filter recursions.

To investigate this last item we use the the theoretical results of chapter 4. This allows to verify these results experimentally.

In relation to the second subproblem, called the *aerodynamic model identification* problem, the following practical items came up for discussion:

- the *independent determination* whether the observation signals in the aerodynamic model are *persistently exciting*,
- the *automatic treatment* of dependencies in the unknown parameters in the aerodynamic model,
- the *detection and tracking* of *time variations* in the unknown parameters,
- the *reliability* of the least squares algorithm to determine these parameters.

The proposed solutions are combined in a general algorithm that has a *modular structure* and allows a complete adaptive operation.

Finally, in **chapter 7** the summarizing conclusions of this research are presented and suggestions are made for further work.

Chapter 1.

An increasing interest in real-time aircraft model identification

1.1. Introduction

The tendency of "digitalization" manifests itself also in the cockpit design of civil aircrafts. This new era in commercial airplanes began in 1968 with the installation of an Inertial Navigation System (INS) on the *B*747 and *DC*10 aircrafts. Over the past 15 years, other digital equipment was added piecemeal into the commercial airplane: digital air data computers, hybrid (digital/analog) VOR (Very high frequency Omni directional Range), DME (Distance Measurement Equipment) and ILS (Instrument Landing System) navigation sensors. It should be remarked that these instruments did not perform new or extra functions with respect to their analog counterparts. Gradually, digital avionics (standing for aircraft electronic systems) have proven to be *more reliable*, easier *to maintain* and *more efficient* than the analog equipment they replaced. This "digitalization" continued further into the integration and unification of different digital aircraft systems and into the development of new display systems [63]. Modern *Flight Management Systems* (FMS) proved to be a very nice realization combining these different features and marked a remarkable point in this evolution. First, because it drastically changed the cockpit layout and secondly because it restarted the endless discussions about how many crew members are needed. The main goal of the present FMS is to *economize the flight cost.* A goal clearly stirred by the increased operation costs, especially caused by the fuel prices combined with lower revenue levels arising from recessionary trends in the economy. To realize this goal intensive use is made of the advantages offered by digital technology [17]. Among these, the following are especially important for aircraft systems:

(a) *savings in weight and space:*

This is of course a very important issue in the economization of flight, since it directly effects the payload to increase. This reduction is based on the new packaging techniques for resistors, capacitors, transistors and integrated circuits by Leadless Chip Carriers (LCC). Compared to the traditional Dual in Line Packaging (DIP) technique more than 50% savings in size and weight could be achieved [47]. Besides the reduction in weight and space, the nature of the LCC production *increases the reliability* of the end product. It is a trend in digital technology that will certainly gain interest in the (near) future.

(b) *reduction of maintenance costs:*

Digital avionics are made *autotestable* through the use of *built in test equipment* (BITE). This means that test signals are inserted into the digital system that have to give a known response. These test cycles can be repeated during flight operation, allowing *to monitor* the different aircraft systems *in flight.* From this information it is now possible to generate internal system diagnostics which eliminates the need for sophisticated watchmakers skills to maintain the aircraft systems.

(c) *increased flexibility:*

Digital systems are characterized by a certain flexibility that consists in *adapting, extending* or *changing* their functional capabilities. An activity that is simplified by the use of High Order Languages (HOL), such as AED, PL/1, Pascal and possibly ADA [1]. Their use has repeatedly proven [95] to result in *much lower* validation costs, required for every modification in the system design, than when using assembly languages. This is mainly because HOL facilitate the implementation step, and hence also provide an increased reliability of the digital system.

(d) *information management:*

[1] ADA is a trade mark of the US department of Defence

The use of Cathode Ray Tubes (CRT) allows to centralize flight deck control functions and offers at the same time an ergonomically justified *man-machine* interaction. Therefore, the use of CRT simplifies dramatically the cockpit layout. Furthermore it reduces the need for human interaction, thereby guaranteeing an increased safety of flight.

From this discussion it may become clear that digital technology forms *one essential element* towards the optimization of aircraft operation, a goal that is achieved through the automation of a number of aircraft control and guidance activities. This will be outlined in a description of present FMS given in section 1.2. This increased reliance on automated systems makes *reliability* of aircraft systems to become of *utmost importance.* With reliability we mean here as well hardware as software reliability. Again from the above items it has been shown that digital technology improves the hardware reliability. This can be increased through *multiple redundancy.* However, for the second requirement one is much more prudent. At this moment, the algorithmic complexity is minimized *by storing* pre-calculated results, often using programmable read-only memories.
This increased automation has a great impact on the complete aircraft design (aerodynamically and structurally). This is why the development of FMS comes into play in very early stages of the general design. Which causes the development costs *to rise.*

A *second important element* in the realization of FMS is the knowledge of accurate aircraft properties. This knowledge is the basic information source in the optimization of flight execution. The judgment of flight execution is done in two ways. First, more globally, one can decompose a complete flight into a number of subsequent phases (climb, cruise, descent, etc.). Or secondly, more locally, one can consider the different manoeuvres performed during flight. The aircraft properties related to the first case are characterized as *the aircraft performances*, while in the second case one speaks of *the aircraft handling qualities.* More precisely, in the optimization of flight execution the aircraft performances are used to compute the "optimal" flight-path, whereas the handling qualities serve as the information source to stir this path in the most economical and safe way. On the other hand it is necessary to check whether these paths are actually followed by the aircraft. Therefore, it is necessary to *reconstruct the aircraft's flight-path.* The use of the following three sets of data:

(a) the performances of the aircraft

(b) the aircraft handling qualities

(c) the reconstructed flight-path

in different optimization schemes applied in the present FMS is discussed in section 1.2. This certainly clarifies the importance these aircraft properties play in FMS design. This chapter further concentrates on how these different aircraft properties can be determined, a problem that is generally characterized as the *aircraft identification problem.* We start with the formulation of this problem, a formulation that mainly depends on how the mathematical model of the aircraft has been defined. In section 1.3 we treat the classical formulation. The resulting mathematical model is used in *off-line* identification techniques to supply existing FMS with the first two information sources (a) and (b) given above. This method is often indicated as the *one-step identification procedure.* A major drawback of this technique is that the obtained identification results are *fixed* and that it does not take into account actual changes of aircraft or atmosphere.

A second method that solves the aircraft identification problem is described in section 1.4. Here, the identification problem is divided into two subproblems. The method is called the two-step identification procedure and was originally formulated by O.H. Gerlach [30]. The formulation of these two subproblems in this section, will show that this method can be considered as a very interesting candidate for solving the aircraft identification problem *on-line* or *in real-time.* The consequences of this are described to some extent in section 1.5. However, what is necessary to realize this is outlined in section 1.6. Here, we point out clearly the specific problem(s) that are analyzed throughout this thesis.

1.2. The operation of present FMS

1.2.1. The general architecture of a FMS

The FMS has four major functions in the aircraft system. (1) Automatic Flight Control, (2) Performance Management, Guidance and Navigation, (3) Advisory, Caution, Warning, Display and (4) Crew Operation. The subdivision of the aircraft system functions in these four parts is illustrated in figure 1.1. These functions include most traditional functional requirements, but now implemented in digital technology. This partitioning allows a *modular* architecture of the FMS. Each of the four subsystems of the FMS has standalone capabilities. The FMS establishes *communication*

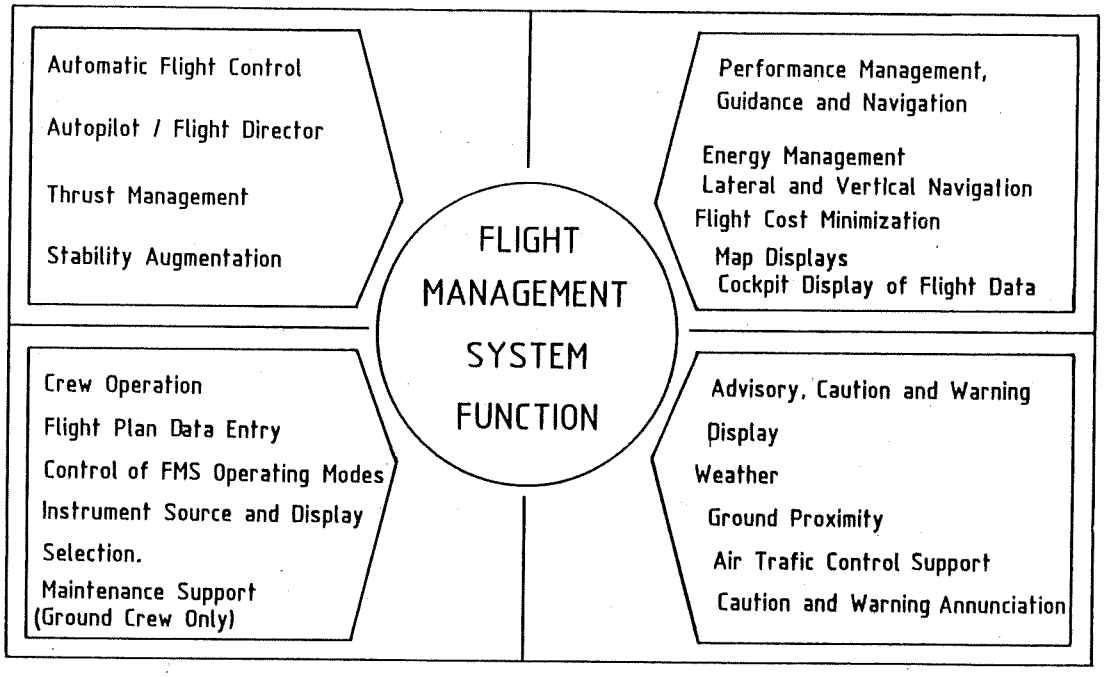


Figure 1.1: A schematic overview of the FMS system functions.

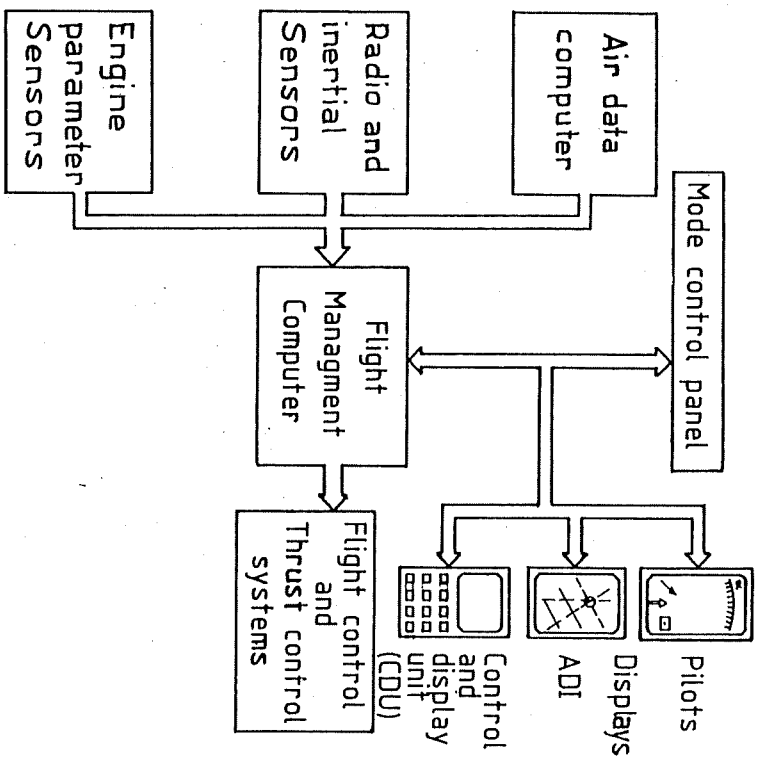


Figure 1.2: Block scheme representation of present FMS.

and *cooperation* between those subsystems in order to achieve the following principal new capabilities [17]:

1. It provides continuous *automatic navigation* (using a combination of the best available sensor data).
2. It allows for the optimization of the operating costs (trade off between fuel, distance and time), by *in-flight performance optimization.*
3. It provides the pilot with a full-time *moving map display,* reducing the crew's workload with increased safety.

A general outline of FMS architecture based on the present systems used on the $B757/B767$ [90] is illustrated in Figure 1.2. This schematic representation indicates the *information stream* through a FMS. The need for communication between the different subsystems of Figure 1.2 and the need for providing the pilot with the most current and accurate information clearly appears from the optimization methods used to achieve the FMS goals.

1.2.2. Optimal path definition methods

The present generation of commercial aircrafts already has the capability to store horizontal flight paths. The *definition of vertical paths* and *the in-flight calculation* of these paths is really new and offers many possible *improvements* in the near future. Methods with varying generality and computational complexity have been used by manufacturers of FMS to define fuel/cost or time efficient trajectories. We now discuss 3 different methods of increasing complexity to calculate *optimal* flight-paths.

1.2.2.1. A very simple approach

The simplest approach is *to store* performance tables or diagrams from the aircraft operation manual in the computer memory. An example of the take-off performances for a $B737$ aircraft is given in fig. 1.3. This information results from the analysis of flight tests performed by the manufacturer. Therefore, the performance information is only correct for a generic sample out of the type of aircraft and does not apply to *possible actual circumstances* which are important in the optimization of flight, such as turbulence, varying engine characteristics, etc. For the example given in fig. 1.3 the optimal take-off engine power setting is constructed in the *TAKEOFF EPR* diagram. Here *EPR* stands for *exaust power ratio* and is

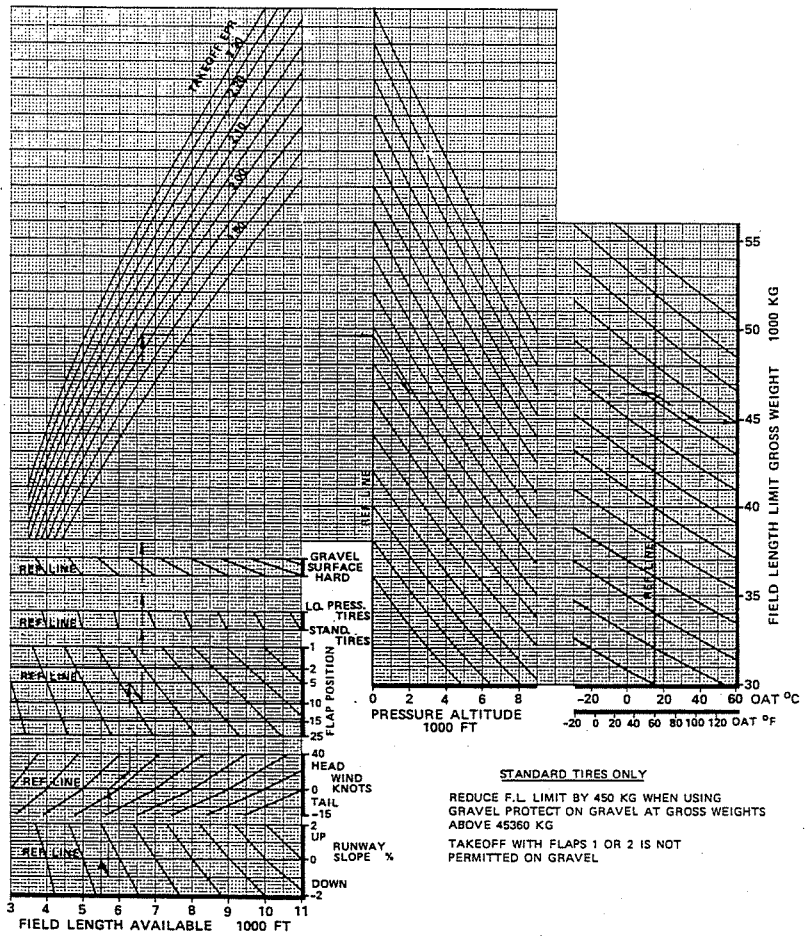


Figure 1.3: Take-off performance diagram, taken from the $B-737$ aircraft operation manual.

used to indicate the trust of the jet engines. The best EPR value for takeoff is determined as the intersection of the line given for a set of current atmospheric conditions (available field length, runway slope, wind, temperature, etc.) and the line given for the aircraft conditions (gross weight, flap settings, etc.). This method is suited for manual operation and is often the only information available to guide the pilot in the flight execution. This method can also be computerized, requiring very little processing time, but a relatively large amount of storage. Another aspect of this procedure is that it optimizes the performances during the current phase of the flight and does not assess the impact of these decisions on subsequent phases of flight. Therefore the method may only be found in some simple FMS, where the method is particularly used for the climb segment.

1.2.2.2. A parameter optimization method

A second but computationally more complex method is a *parameter iterative optimization* process. This method is based on the fact that airlines summarize the running expenses of their different commercial airplanes into one parameter, often called the Direct Operation Costs (DOC). This is done by statistical analysis. This quantity of interest is expressed as a (given) function of a number of key parameters. Some of these parameters are fixed or measured and some of them are considered as *independent variables*, usually one or two in number. The expressions of the DOC are given for all different phases of flight. For each particular phase the expression for the DOC is evaluated for the given and measured quantities and then optimized for the independent parameters by simple optimization methods. For example, for cruise flight the DOC can be expressed into fuel costs, time costs, etc. (parameters fixed in advance) and fuel flow, weight, etc. (which is obtained by measurement) and Mach number. A simple way to find the "optimal" Mach number, used in the present FMS [90], is to calculate the value of the DOC for several values of the Mach number until a minimal value is found. In this method the performance characteristics of the airplane are *roughly summarized* into pre-specified key parameters (e.g. time cost of the airplane). Just like the performance diagram approach, discussed in the previous subsection, this procedure only optimizes current performances.

1.2.2.3. A global optimization scheme

The third method and most complex scheme for computing optimal path profiles is by solving *two point boundary value optimization problems* [12]. For this problem a number of numerical techniques have been developed, such as e.g. the method of calculus of variation [71]. In these methods a mathematical model of the aircraft is used which is a set of state equations. This set of equations generally results from flight test analysis techniques and will be discussed from the next section on. The method is still far too complex to be solved in real-time, and several manufacturers have employed *simplified* versions of the method. Here *fixed state space models* are used for dominant flight reference conditions in the entire flight envelope instead of the *optimal adaptive model* that takes into account changing environmental circumstances. Also the results of the calculus of variation obtained with larger *ground based computer* are modelled into simplified versions for on-line use.

To gain insight in the operation of FMS and the role of aircraft identification methods herein, it is instructive to discuss the results that come out of such a ground based optimization method. This consists first in the determination of the *horizontal trajectory*. The elements that play an important role in the treatment of this problem are well illustrated by the following simple example.

Example 1.1: In this example we consider the trajectory of an airplane moving through a windfield caused by a depression with its origin coinciding with the origin of the coordinate system $(x_1(t), x_2(t))$ taken in fig. 1.4. For these conditions, *a very simple* mathematical model is used to describe the aircraft motion through this wind field:

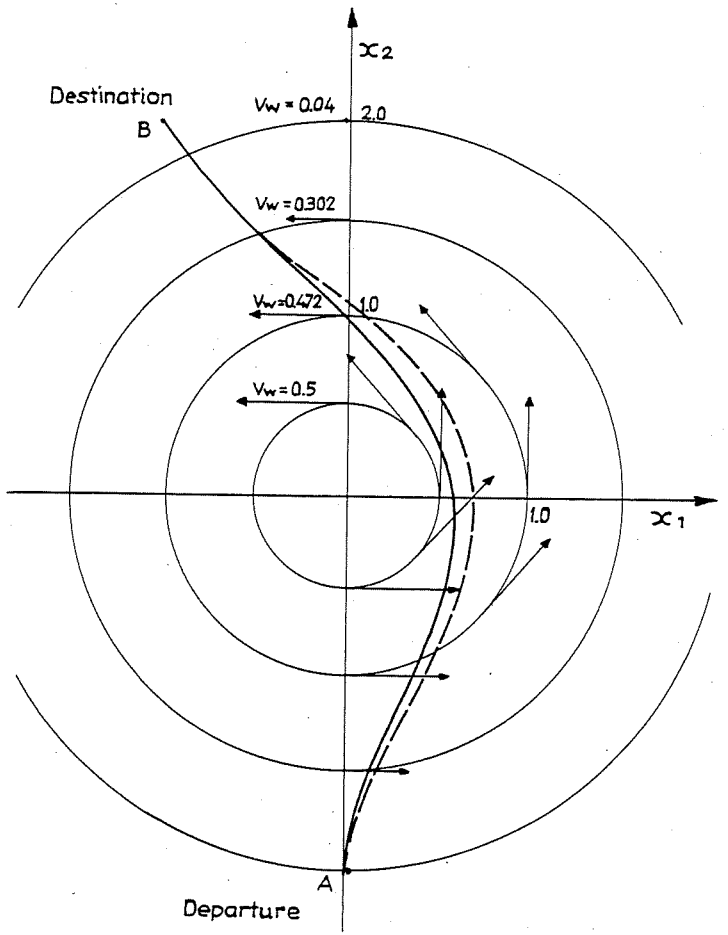
$$\dot{x}_1 = V cos\alpha + x_2 e^{-(x_1^2 + x_2^2 - 0.25)} \tag{1.1}$$

$$\dot{x}_2 = V sin\alpha \underbrace{-x_1 e^{-(x_1^2 + x_2^2 - 0.25)}}_{\text{influence of the windfield}} \tag{1.2}$$

with (x_1, x_2) — the coordinates of the airplane with respect to the chosen coordinate system

V — the magnitude of aircraft speed relative to the air

α — the steering angle relative to the North direction, i.e. the x_2 direction



—— concentric circles of constant wind velocity V_w, which is tangent and counterclockwise
—— *minimum time* trajectory
- - - *minimum fuel* trajectory for fixed final time t_f

Figure 1.4: Optimal horizontal trajectory through a windfield caused by a depression.

In this model the quantities (V, α) are considered as the control inputs of the system. The set of differential equations (1.1) and (1.2) is now used in the method of calculus of variation to determine the optimal trajectory to fly from a given starting point to a given destination, marked by A and B respectively in fig. 1.4. The optimality here is expressed in the choice of a particular criterion. If we e.g. consider the *minimal time path*, the related optimization problem becomes:

$$J_1 = \min \int_{t_0}^{t_f} dt \quad \text{subject to (1.1) and (1.2)} \tag{1.3}$$

where t_0 is the starting time and t_f is the (unknown) final time. The solution of problem (1.3) consists of the determination of the flight-path $(x_1(t), x_2(t))$ and the input quantities $(V(t), \alpha(t))$. The flight path, computed by a numerical implementation of the method of the calculus of variation, i.e. the quasi-linearization method [12], is plotted by the full line in fig. 1.4.

On the other hand, if we want to determine a fuel efficient trajectory, we may consider the following criterion:

$$J_2 = \min \int_{t_0}^{t_f} V^2(t) dt \quad \text{subject to (1.1) and (1.2)} \tag{1.4}$$

for fixed final time t_f.
With the same numerical method as used in the previous problem, the optimal flight path is given by the dotted line in fig. 1.4.
From this example we learn that it is *very useful to have* a mathematical state space model of the aircraft in order to determine optimal flight-paths. The state space model used in the example is far too simple for practical use, since it does not consider the effect of drag, lift, coriolis, etc. This will be analyzed more deeply in the next sections.

In a second part of this global optimization scheme, an *optimal vertical* trajectory is prescribed. Figure 1.5 illustrates the altitude, indicated airspeed (IAS) and the Mach number for the entire flight profile of a generic flight [22]. About this figure it ought to be remarked that the continuous climb during cruise is not allowed by the ATC (Air Traffic Control) procedures. Therefore this "ideal" path has to be approximated by a "step-climb" procedure (denoted in fig. 1.5 by the dotted line). The decision to perform these step-climbs is also made by the FMS. Secondly, it is clear from fig. 1.5 that the IAS varies continuously in climb and descent phase.

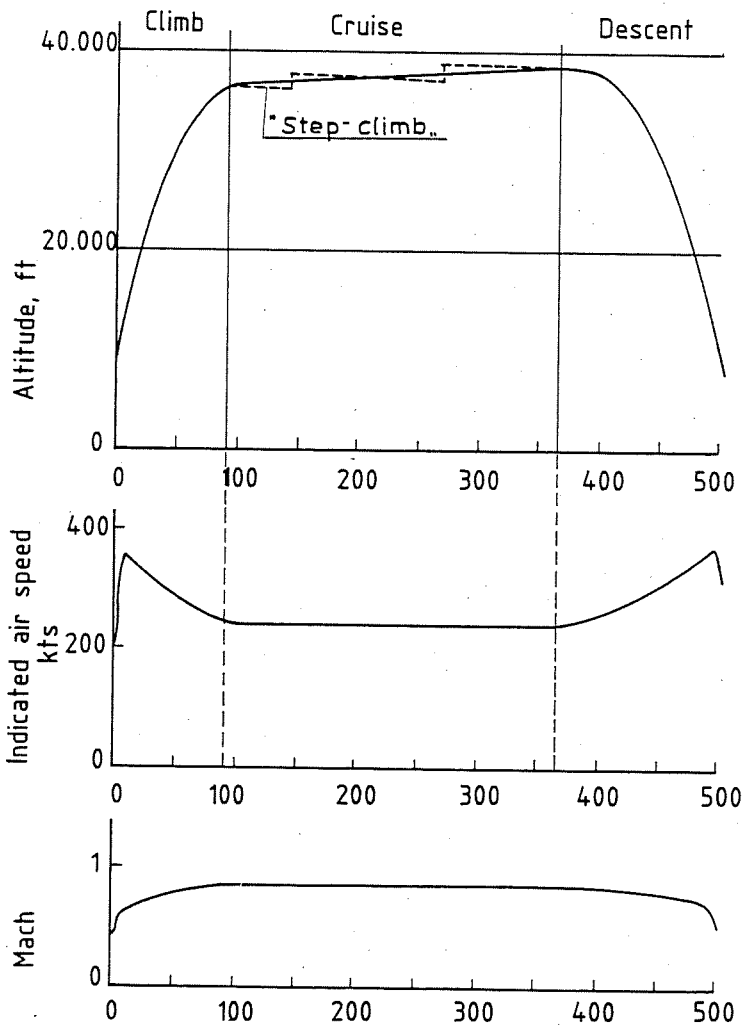


Figure 1.5: Optimal vertical trajectory of a generic flight.

Firstly, this variation is in contrast to the procedures described in the aircraft operation manual, where normally climb is indicated at constant, maximum climb trust (therefore with constant IAS/Mach) and secondly it requires a *coupling* to the autothrottle system, because the pilot is unable to follow these variations.

1.3. Aircraft Identification: The classical approach

1.3.1. The problem statement

The knowledge of performances and handling quantities of an aircraft plays a crucial role in the development of FMS and automatic control systems. Through the increasing flight envelope (flight region) of the present and next generation aircrafts these quantities may vary over a very wide range. Accurate knowledge is however required over the whole flight region in order to achieve the goals of a FMS as discussed in the previous section. The measurement of the quantities of interest in flight is a *very time consuming* affair, letting alone the question whether this can be achieved within the required accuracy.
Therefore, non-stationary flight test methods have been developed. In this case dynamic flight test measurements are used in a *dynamic system identification problem.* More precisely, given the measurements of input signals (control surface angle deflections) and output signals (accelerometer and rate gyro outputs, airspeed, altitude deviation) the *task* is to determine:

(a) the performances of the aircraft

(b) the aircraft handling qualities

(c) the reconstructed flight-path

which are the three sets of data required in the design of FMS. The first two problems are related to the estimation of the so-called *stability and control derivatives*, which are relevant for the characterization of the aerodynamic forces and moments acting on the aircraft. From these identification results, the performances and handling qualities can be easily derived, see [45].

The key to the solution of the identification problem is given by the formulation of a *mathematical model* that represents the aircraft dynamics. In the next subsection, this is first derived for the classical approach called the *one-step identification procedure.*

1.3.2. The mathematical aircraft model

A system model describing the aircraft dynamics is derived from Newton's second law and the kinematical relations. Under the assumption that the aircraft is symmetric, the lateral-longitudinal coupling effects can be neglected. Further, if we assume that the airplane is moving with respect to a planar, non-rotating earth and that the aircraft is considered as a rigid body, it is possible to describe the *symmetric aircraft motion* by the following set of *non-linear* differential equations [24] given in a fixed body reference frame, see fig. 1.6 and table 1.1 for notational explanation:

1. The force equations and moment equation:

$$\begin{aligned}
&\text{along the X-axis :} && m\dot{u} &&= X(u,w,\theta,q,\delta_e) && -mgsin\theta && -mqw && \text{(a)}\\
&\text{along the Z-axis :} && m\dot{w} &&= Z(u,w,\theta,q,\delta_e) && +mgcos\theta && +mqu && \text{(b)}\\
&\text{around the Y-axis :} && I_y\dot{q} &&= M(u,w,\theta,q,\delta_e) && && && \text{(c)}\\
& && && && \underbrace{\qquad}_{(1)} && \underbrace{\qquad}_{(2)} && \underbrace{\qquad}_{(3)}
\end{aligned} \tag{1.5}$$

with m – the aircraft mass
g – the gravitational constant
I_y – the moment of inertia around the Y-axis
(1) – the aerodynamic forces and moment
(2) – the gravitational forces
(3) – the coriolis forces

2. The kinematical relation:

$$\dot{\theta} = q \tag{1.6}$$

3. A differential equation for the deviation of the altitude Δh

$$\Delta\dot{h} = u\,sin\theta - w\,cos\theta \tag{1.7}$$

symbol	quantity
u	component of the aircraft velocity along the X_B-axis
w	component of the aircraft velocity along the Z_B-axis
q	rate of pitch around the Y_B-axis
θ	angle of pitch
Δh	altitude deviation
X	aerodynamic force along the X_B-axis
Z	aerodynamic force along the Z_B-axis
M	aerodynamic moment around the Y_B-axis
δ_e	elevator deflection angle
$c.g.$	center of gravity

Table 1.1: Definition of the quantities that describe the symmetric aircraft motion.

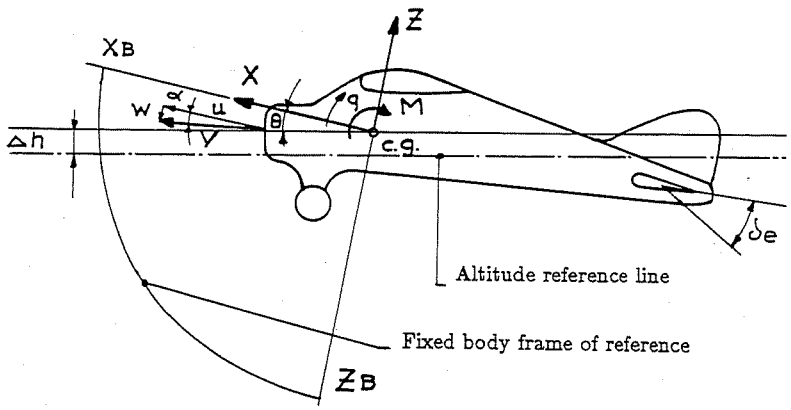


Figure 1.6: Definition of system quantities for the symmetric aircraft motion.

The aerodynamic forces and moment in (1.5) are related to the quantities between brackets. However, these relationships are *not known* a priori. In vector notation the set of equations (1.5-1.7) can be denoted as:

$$\dot{x} = f(x,u,t) \tag{1.8}$$

where f is an unknown vector function and the vector quantity x and u are respectively the state and input variables of the chosen state space description. They are defined in table 1.2.

From a practical point of view, a good approximation to the real aircraft motion is delivered by the linearized state equations [24]. This linearization is explicitly performed in Appendix A. It yields a set of linear differential equations which describe the perturbation of motion from a given working point (taken as a stationary flight condition). The resulting set of linear differential equations can be put in vector notation as:

$$\dot{x} = Ax + Bu \tag{1.9}$$

In this equation the quantities x and u are again the state and input quantities defined in table 1.2, but they now represent the deviations from their reference values. The system matrices $\{A, B\}$ in (1.9) contain the *unknown* stability- and control derivatives. This results from the modelling of the aerodynamic forces and moment. Through the linearization, these are now in a *linear relation* to the state and input quantities of (1.9). These relationships are summarized in the aerodynamic model:

$$\begin{pmatrix} \frac{2\mu_c c}{V_T^2}\frac{X(t)}{m} \\ \frac{2\mu_c c}{V_T^2}\frac{Z(t)}{m} \\ \frac{2\mu_c c k_y^2}{V_T^2}\dot{q}(t) \end{pmatrix} = \underbrace{\begin{pmatrix} C_{X_u} & C_{X_\alpha} & C_{X_q} & C_{X_\delta} \\ C_{Z_u} & C_{Z_\alpha} & C_{Z_q} & C_{Z_\delta} \\ C_{m_u} & C_{m_\alpha} & C_{m_q} & C_{m_\delta} \end{pmatrix}}_{\text{stability- \&}} \underbrace{\begin{pmatrix} \frac{u(t)}{V_T} \\ \alpha(t) \\ \frac{q(t)c}{V_T} \\ \delta_e(t) \end{pmatrix}}_{\text{control}} \quad \text{derivatives} \tag{1.10}$$

with
- μ_c – a dimensionless mass parameter
- c – the mean aerodynamic chord
- k_y^2 – the dimensionless radius of the moment of inertia around the Y-axis
- V – the speed of the aircraft relative to the air
- $(.)_T$ – the value in the ref. condition at time T

It should be remarked that this aerodynamic model is given in dimensionless

form, which is often done in aircraft modelling [24].
To complete the system description, it is necessary to define an observation model. For the measured output quantities, denoted by the vector y in table 1.2, the observation model can simply be given by a set of algebraic equations (explicitly derived in Appendix A), as:

$$y = C_x x + C_u u \tag{1.11}$$

where again some of the entries in the $\{C_x, C_u\}$-matrices are unknown.

	quantity	symbol	unit
input u	elevator angle	δ_e	rad
state x	speed along the X-axis	u	$\frac{m}{sec}$
	angle of attack	α	rad
	angle of pitch	θ	rad
	rate of pitch	$\frac{qc}{V_T}$	/
	altitude deviation	Δh	m
output y	speed along X-axis	u	$\frac{m}{sec}$
	altitude deviation	Δh	m
	specific aerodynamic force along the X-axis, measured with an accelerometer	$\frac{X}{m}$	$\frac{m}{sec^2}$
	specific aerodynamic force along the Z-axis, measured with an accelerometer	$\frac{Z}{m}$	$\frac{m}{sec^2}$
	rate of pitch, measured with a rate gyro	q	$\frac{rad}{sec}$

Table 1.2: Definition of state space quantities of the model for the one-step identification method

1.3.3. The one-step identification method

The general *non-linear* aircraft system can be approximated by a *linear* model similar to (1.9) and (1.11), in which the stability- and control derivatives *vary in time*. These parameters are considered to be constant

for a single stationary flight condition. This way of approximation is often used with success to model the non-linear aircraft dynamics, e.g. in modern flight simulator design [24].
Hence, the basic problem is the determination of the linear model (1.9) and (1.11) for one stationary flight condition. Which reveals more precisely in the estimation of the unknown entries in the system matrices $\{A, B, C_x, C_u\}$. A problem of this kind is characterized as a *non-linear* dynamic system identification problem [23]. And solutions are proposed in the literature, e.g. by Maximum Likelihood techniques [69], [96]. The method is quite elaborate since it essentially works *iteratively* for real aircraft identification problems and on the same time suffers from a lot of numerical difficulties, such as e.g. convergence problems [35]. Therefore, the method relies on a vast basis of experience.
However, it is still used nowadays to supply FMS with the necessary information about the aircraft performances and handling qualities [90]. Hereby, the resulting information can only be "considered" as exact for the performed flight tests and hence it is *fixed* in this sense. For this case, the actual flight path is checked by the use of INS, *necessarily* combined with radio navigation systems to correct for INS inaccuracies.

1.4. The two-step identification method

1.4.1. The mathematical formulation

The mathematical modelling of the aircraft dynamics remains also here based on the set of non-linear differential equations (1.5-1.7). However, we now *first* consider the *subproblem* **to reconstruct the aircraft's flight path**, i.e. to determine the time histories $u(t), \alpha(t), \theta(t), q(t)$ and $\Delta h(t)$. In this case, the lack of information about the aerodynamical relationships is circumvented by using inertial measurements. In this way, the rate of pitch is accurately measured, e.g. by a laser gyro. Therefore (1.5c) is no longer necessary and the symmetric aircraft motion can be described by equation (1.5a-b), (1.6) and (1.7). It may also be cast in the general state space formulation, given in vector notation by,

$$\dot{x}_{fp} = f_{fp}(x_{fp}, u_{fp}, t) \tag{1.12}$$

where now the system state x_{fp} and the input u_{fp} are defined in table 1.3. Here fp indicates that the model is only of mathematical interest for re-

constructing the flight-path. The linearization, according to the procedure described in Appendix A, of the set of equations 1.12 becomes very simple now. For a given reference condition, we obtain:

$$\begin{pmatrix} \dot{u} \\ \dot{\alpha} \\ \dot{\theta} \\ \Delta\dot{h} \end{pmatrix} = \begin{pmatrix} 0 & 0 & -g\cos\gamma_T & 0 \\ 0 & 0 & -g\frac{\sin\gamma_T}{V_T} & 0 \\ 0 & 0 & 0 & 0 \\ \sin\gamma_T & -V_T\cos\gamma_T & V_T\cos\gamma_T & 0 \end{pmatrix} \begin{pmatrix} u \\ \alpha \\ \theta \\ \Delta h \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{V_T} & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{X}{m} \\ \frac{Z}{m} \\ q \end{pmatrix} \tag{1.13}$$

with γ the flight-path angle, defined as $\gamma = \theta - \alpha$.
Or equivalently we can write (1.13) in vector notation as,

$$\dot{x}_{fp} = A_{fp}x_{fp} + B_{fp}u_{fp} \tag{1.14}$$

where again x_{fp} and u_{fp} are as defined in table 1.3, but now represent the deviations from their reference values.
The observation model for the output variables, denoted as y_{fp} in table 1.3, now simply becomes:

$$\begin{pmatrix} u \\ \Delta h \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ \alpha \\ \theta \\ \Delta h \end{pmatrix} \tag{1.15}$$

also denoted as,

$$y_{fp} = C_{fp}x_{fp} \tag{1.16}$$

	quantity	symbol	unit
input u_{fp}	specific aerodynamic force along the X-axis, measured with an accelerometer	$\frac{X}{m}$	$\frac{m}{sec^2}$
	specific aerodynamic force along the Z-axis, measured with an accelerometer	$\frac{Z}{m}$	$\frac{m}{sec^2}$
	rate of pitch, measured with a rate gyro	q	$\frac{rad}{sec}$
state x_{fp}	speed along the X-axis	u	$\frac{m}{sec}$
	angle of attack	α	rad
	angle of pitch	θ	rad
	altitude deviation	Δh	m
output y_{fp}	speed along X-axis	u	$\frac{m}{sec}$
	altitude deviation	Δh	m

Table 1.3: Definition of the state quantities in the basic flight-path reconstruction model

The main advantage of the derived state space description (1.14) and (1.16) is that the entries of the system matrices $\{A_{fp}, B_{fp}, C_{fp}\}$ are *completely known*. Therefore, this model ideally fits in the framework of *linear filtering* for reconstructing its state variables. The latter incorporate the non-measured flight-path quantities $\alpha(t)$ and $\theta(t)$. So that by solving this problem referred to as the **basic flight-path reconstruction problem** the flight-path may now be considered to be known.
To complete the aircraft identification problem, defined in section 1.3.1, we are still left with the derivation of the stability- and control derivatives. This will be handled as a second subproblem in the two-step identification method, called the **aerodynamic model identification problem**. If we assume that the flight-path can be reconstructed *accurately*, the aerodynamic model (1.10) becomes a *regression model* in the unknown stability- and control derivatives.

The investigation of these two subproblems is the practical goal incorporated in this thesis. Initially, we will start this research be exploring some very interesting properties of the flight-path reconstruction model (1.14) and (1.16), based on an initial analysis performed in [101].

1.4.2. Remarkable properties of the basic flight-path reconstruction model

Property 1.1: The *eigenvalues* of the system matrix A_{fp} in (1.14) are all located in the origin of the complex plane.

Corollary 1.1: From this property it follows that the right eigenvector space of A_{fp} is determined by the kernel of A_{fp}, defined as:

$$ker\{A_{fp}\} = \{x | A_{fp}x = 0\} \tag{1.17}$$

Since all the eigenvalues coincide (at zero) and the dimension of the right eigenvector space, defined in (1.17), is less than the state dimension, the eigenvalue structure of A_{fp} is given by the so-called *Jordan form*. From a numerical point of view, see [123], it is not recommended to work with this particular structure.

Property 1.2: The *nilpotency index* of the system state matrix A_{fp} is **2**, i.e. (A_{fp}^2) is the null matrix.

Corollary 1.2: The matrices,

$$A_k = e^{A_{fp}\Delta t} \;;\; D_k = \int_0^{\Delta t} e^{A_{fp}{}^T d\tau}.B_{fp} \;;\; C_k = C_{fp}$$

of the discretized version of (1.14) and (1.16) (discussed more extensively in chapter 4, section 4.3.3) namely:

$$x_{k+1} = A_k x_k + D_k u_k \tag{1.18}$$

$$y_k = C_k x_k \tag{1.19}$$

can be computed *analytically* for all possible reference flight conditions. The system matrices A_k and D_k have the following form:

$$A_k = \begin{pmatrix} 1 & 0 & -gcos\gamma_T \Delta t & 0 \\ 0 & 1 & -g\frac{sin\gamma_T}{V_T}\Delta t & 0 \\ 0 & 0 & 1 & 0 \\ sin\gamma_T \Delta t & -V_T cos\gamma_T \Delta t & V_T cos\gamma_T \Delta t & 1 \end{pmatrix}$$

$$D_k = \begin{pmatrix} \Delta t & 0 & -gcos\gamma_T \frac{\Delta t^2}{2} & 0 \\ 0 & \Delta t & -g\frac{sin\gamma_T}{V_T}\frac{\Delta t^2}{2} & 0 \\ 0 & 0 & \Delta t & 0 \\ sin\gamma_T \frac{\Delta t^2}{2} & -V_T cos\gamma_T \frac{\Delta t^2}{2} & V_T cos\gamma_T \frac{\Delta t^2}{2} & \Delta t \end{pmatrix} .B_{fp}$$

For the symmetric aircraft motion these reference flight conditions are determined by γ_T and V_T. The calculation is based on a choice of a very small discretization period Δt. This is taken in the neighborhood of the sampling period $0.01sec$ which is used in practical flight tests [41]. This allows to evaluate the matrix exponential above into a power series around zero, which based on property 1.2 only contains two terms. The availability of the discrete system model (1.18) and (1.19) is of course very important if we want to solve related problems on a digital computer.

Corollary 1.3: For a single reference flight condition the state space model (1.14) and (1.16) is time-invariant. For such a case, it is possible to analyze the *observability* by checking the singularity of the observability matrix $\mathcal{O}$, defined as [48]:

$$\mathcal{O} = (C'_{fp} \mid A'_{fp}C'_{fp}) \tag{1.20}$$

This special form is again due to property 1.2 and allows an analytical computation:

$$\mathcal{O} = \left(\begin{array}{cc|cc} 1 & 0 & 0 & sin\gamma_T \\ 0 & 0 & 0 & -V_T cos\gamma_T \\ 0 & 0 & -gcos\gamma_T & V_T cos\gamma_T \\ 0 & 1 & 0 & 0 \end{array}\right)$$

which for all flight conditions has *full rank.*

Similarly, the *controllability* could be checked using the controllability matrix $\mathcal{C}$ [48]:

$$\mathcal{C} = (B_{fp} \mid A_{fp}B_{fp}) \tag{1.21}$$

or explicitly:

$$\mathcal{C} = \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & -gcos\gamma_T \\ 0 & \frac{1}{V_T} & 1 & 0 & 0 & -g\frac{sin\gamma_T}{V_T} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & sin\gamma_T & cos\gamma_T & 0 \end{array}\right)$$

which again always has full (row) rank.

Property 1.3: A reordering of the state vector defined in table 1.3 of the basic flight-path reconstruction problem, according to:

$$x = (\theta, \alpha, u, \Delta h)'$$

results in a (lower) *real Schur form* of the matrix A_k in (1.18):

$$A_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -g\frac{sin\gamma_T}{V_T}\Delta t & 1 & 0 & 0 \\ -gcos\gamma_T \Delta t & 0 & 1 & 0 \\ V_T cos\gamma_T \Delta t & -V_T cos\gamma_T \Delta t & sin\gamma_T \Delta t & 1 \end{pmatrix} \tag{1.22}$$

Additionally, a *constant* state space transformation T of the above defined state space representation,

$$z_k = Tx_k = \begin{pmatrix} 1 & 0 & \frac{\Delta t}{2} & 0 \\ -1 & 1 & 0 & \frac{\Delta t}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

yields a (lower) *observer Hessenberg form* for A_k.

The last two forms fit in the framework of "condensed forms" that will be defined and discussed in the next chapter.

1.5. The perspectives of the two-step method

The two basic subproblems of the two-step procedure, presented in section 1.4.1, fit in the framework of *linear state filtering and estimation.* This simple formulation of both steps makes the method very attractive for *real-time operation.* The realization constitutes a challenge that will certainly stimulate a number of appealing developments.

First, this method can serve as an on-line flight test analysis procedure and hence will reduce the post-flight analysis time significantly. Furthermore, it may also reduce the flight test time, by informing the pilot whether

the manoeuvre is performed sufficiently well to achieve the required accuracy. Both aspects are very important features in the development of future civil and military aircrafts, especially because of their large flight envelopes.

Secondly, the real-time identification of a state space model for an aircraft offers several possibilities to improve the flight operation, since it allows to identify an aircraft model that ***adapts automatically*** to time varying operational circumstances, e.g. change in atmospheric conditions, engine deterioration, etc. Therefore, it has been a tool that was looked after for a long time. Until now, several projects on this topic have abandoned in spite of the use of very powerful computers [13]. One of the main reasons was that many simplifications were needed in order to solve the non-linear identification problem discussed in section 1.3.2. These simplifications are not necessary for the two-step procedure, which may therefore be considered as an interesting candidate. The real-time realization of this procedure will certainly influence the following items:

- *The aircraft design:*

  The identification of a state space model, described mathematically by equations (1.9) and (1.11) is *the key* in the explicit adaptive control scheme. This scheme, depicted in fig. 1.7, uses this information to compute a (state) feedback law. This is done according to the chosen control objective, e.g. the optimal control framework [12]. Furthermore, this closed loop regulation offers the possibility to handle the ***aircraft stability*** automatically. And in this way it allows the original aircraft design to be ***unstable***. These type of aircrafts are called ***relaxed static stability*** aircrafts. A principle that is already used in the General Dynamics YF-16 combat aircraft. However for this design *fixed* state space models were used for a number of dominant flight conditions. This design philosophy is also the main reason for the high performance characteristics and excellent manoeuvring qualities of the YF-16. But at the same time, it makes the regulation unit to become a vital element for the flight execution. This is why a failure of any part of the control system causes the loss of an aircraft.

- *The FMS-design:*

  A first requirement in the realization of the global optimization method presented in section 1.2.2.3 was the availability of a state space model of the aircraft. Since in our two-step identification method the aircraft model is no longer assumed to be *fixed*, the

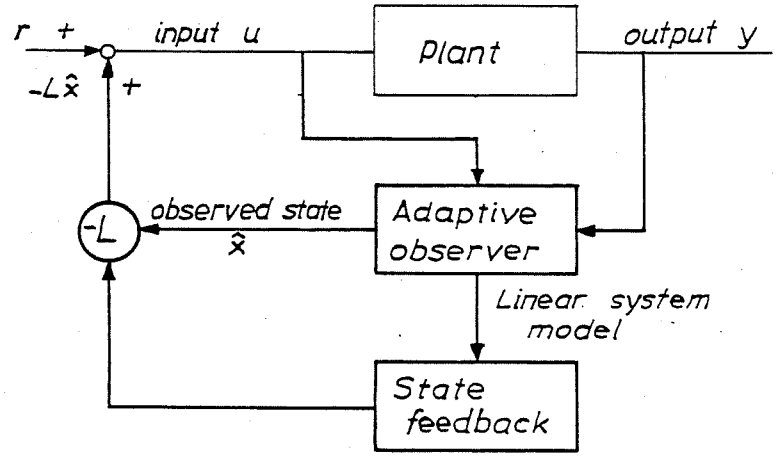


Figure 1.7: Block scheme representation of the explicit control scheme.

optimization results will improve [99], thus resulting in a more comfortable, safe and economical flight.

- *Aircraft navigation performances:*

  The realization of the first step of the two-step method will allow to reconstruct the aircraft's flight path accurately without the need of other navigational information sources. The avoidance of e.g. radio navigation is especially of great interest in military aviation. Since the aircraft can operate completely independently.

  Also, for civil applications the realization of this single step can also be very interesting. For example in taking air photos of a region that has to be mapped. Here, deviations from the desired flight-path perturb the results of the air photo. Therefore, very time consuming ground based correction methods were needed to correct for these errors, see e.g. [121]. However, the accurate knowledge of the aircraft's actual flight-path will avoid (or certainly reduce) the need of this post-flight analysis.

1.6. The description of the research goals

The simplicity of the two-step procedure and the many perspectives described in the previous section motivate to make the method real-time applicable. This requires research into three main areas:

- digital technology
- sensor technology
- algorithmic design

The research performed in this thesis concentrates on the third aspect. More specifically, algorithms will be constructed for both steps of the two-step procedure that, due to the real-time operation, have to deal with the following *algorithmic requirements*:

1. *Algebraic efficiency:*

   For the present airborn flight computers, this will still remain a very important algorithmic aspect. For example in the case of the ROLM1602B airborn computer, used by the Dutch Aerospace laboratory NLR, a *single precision* addition using software floating-point routines requires $300\mu s$. For the *double precision* case this becomes even 5 times worse. Of course, the fast developments in micro electronics will make this requirement less stringent.

2. *Algorithmic reliability and accuracy:*

   These aspects often go hand in hand. In the acquisition of real-time measurements special care has to be taken in the presence of *measurement errors*. Their magnitude of course depend on the measurement system used. However, the manipulation of the data by the chosen algorithmic implementation can also deteriorate the desired results. This effect has to be kept within certain tolerances, and should be minimized. These aspect will *remain important*, independent of the evolution of digital technology or sensor technology. They require:

   - the development or use of *reliable software* which minimizes the error increase throughout the computations involved. This aspect is even stressed when preference is given to single precision computations in order to speed up the computations.

   - a careful *modelling* and treatment of the different *error sources*.

   The same requirements remain valid for the automatic treatment of the time-varying conditions.

These algorithmic requirements will be investigated especially for the linear filtering and estimation problems that occur in the two-step procedure. The results related to this investigation will be checked on digitally simulated flight test data using the aircraft model derived for the one-step identification method as described in section 1.3.2. The unknown entries of the system matrices are taken from flight test results performed with the high wing DHC-2 Beaver aircraft, see fig. 1.8 [72]. This is the laboratory aircraft of the Delft University of Technology.

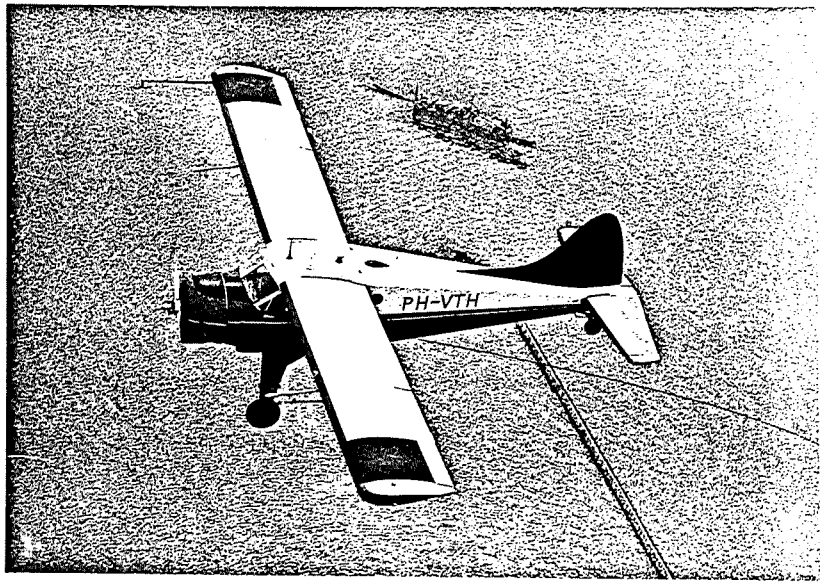


Figure 1.8: The DHC-2 Beaver aircraft.

Chapter 2.

The use of condensed forms in linear system theory

2.1. Introduction

The state space description (1.14) and (1.16) derived for the basic flight path reconstruction problem is determined by the matrix triplet $\{A_{fp}, B_{fp}, C_{fp}\}$. It has been shown that for the discrete time case, this triplet could be transformed by simple and constant similarity transformations to another triplet, exhibiting a "special structure". Moreover, the transformations used have a condition number one or in the vicinity of one. In this way, the obtained special forms of the aircraft model form a subset of the more general class of the so-called *condensed forms*. The main advantage for our flight-path reconstruction problem is that these forms were derived *analytically*, i.e. at no extra computing costs and without loss of numerical accuracy.

These condensed forms have a broad application spectrum in the development of algorithms which combine the contradicting requirements of reliability and efficiency. Many articles already appeared on this topic, see e.g. [4], [6], [25], [38], [53], [56], [73], [75], [112], [113], [122] and [123]. The specific results described in these different papers all uncover only a part of what we will call "*a new class of algorithms*". The separate presentation in these papers may give the impression that the use of condensed forms is *a formal trick*. However, in this chapter we will outline that it is not, but that it can really be considered as a *new concept* in designing algorithms.

This is done firstly by giving a survey of different algorithms in numerical analysis and system theory, where condensed forms were applied with success. And secondly, by filling gaps in this *unification* of the use of condensed forms. In this and later chapters, we shall present new algorithms and implementations based on the ideas of condensed forms.

For a long time, system scientists used other structures, such as the Jordan canonical form, to come up with very efficient and elegant solutions, see e.g. [67] and [18]. However, a lot of these techniques remained only used as interesting textbook examples. This was mainly because the nice theoretical results didn't come out in practice due to the sensitivity of the algorithm to round-off errors, or errors on the data. Mathematical investigation by numerical analysists learned that the transformations required to find such canonical forms may be *very badly* conditioned [123]. Therefore, they advised to use unitary (in the complex case) or orthogonal (in the real case) similarity transformations, and this mainly for two reasons:

1. A major reason is the numerical sensitivity of the problem at hand. The sensitivity of several problems in linear algebra can indeed be expressed in terms of certain norms, singular values or angles and each of these are in general invariant under unitary transformations. Unitary/orthogonal transformations therefore allow one to reformulate the problem in a new coordinate system (usually more amenable for solving the problem) and this *without* affecting its sensitivity. Typical examples of this are the QR decomposition [33] and the Singular Value Decomposition (SVD) [32], both used for constructing least squares solutions to linear systems: they reduce the least squares problem $\min_x \|Ax - b\|_2$ to a solvable but smaller linear system, without altering its sensitivity (the latter is e.g. not the case for methods based on solving the normal equations $A^*Ax - A^*b = 0$ [122]).

2. A second reason is the numerical stability of the algorithm used for solving the problem. Most decompositions involving unitary transformations can be obtained by a sequence of Givens reflections or Householder transformations, each of which can be performed in a numerically stable manner. The concatenation of such transformations can also be performed in a backward stable manner. Numerical errors resulting from previous steps are indeed maintained in norm throughout subsequent steps because these transformations (and their inverse) have a condition number equal to one [123].

Originally, condensed forms were used in numerical analysis. See for example in [122] and [123] for an extensive treatment in eigenvalue problems. In this area they showed to be of great interest especially in *iterative type of algorithms*. First, because for these type of algorithms the distortion of the sensitivity and loss of accuracy can become significant when using non-unitary transformations. And secondly, the reduction to a particular condensed form is usually done only once and is then maintained throughout the recursions. This resulted in considerable savings in computing time. This technique has for example been used to solve eigenvalue problems. Here the full matrix was first reduced to Hessenberg form in an initial step and then maintained in all subsequent steps of the algorithm [25], [53], [75], [122] and [123].

It is the aim of this chapter to demonstrate the usefulness of these condensed forms *in linear system theory*. This, I think, may help to promote the development of reliable algorithms in the field of a linear system theory. Until now, still articles appear where use is made of non-unitary transformations to formulate "attractive" solutions for problems where they can easily be avoided, see e.g. [18] and [84]. Therefore, we first describe in section 2.2 some particular properties of 3 basic orthogonal transformations. Their use in possible implementations of different algorithms is discussed throughout this chapter. In section 2.3 different types of condensed forms, their computation and the computational complexity for state space models defined by the quadruple $\{A, B, C, D\}$ are presented. The survey of the use of condensed forms in a broad class of problems in numerical analysis and system theory, see [97] and [98], is given in section 2.4. A separate subsection is however made for the use of these forms in the linear state observation problem. This will be very instructive for the more complex Kalman filter problem, treated in the subsequent chapters. Therefore, some experiments are performed in this subsection in order to gain insight in the numerical problems that may occur when observing the state of a linear system. In section 2.5, the extension of the use of condensed forms to the so-called *descriptor systems* is discussed. After defining the generalized condensed forms for this class of systems, we will present a *new* algorithm to solve the the generalization of the reduced order observer. Finally, section 2.6 presents some concluding remarks.

2.2. Review of basic orthogonal transformations

Here the analysis is restricted to real matrices. For the case of complex matrices the transformations are called unitary and are treated in [123].

2.2.1. Elementary Permutations

The *simplest* orthogonal transformation is the permutation of two rows or columns. Interchanging the i-th and j-th row, for $i < j$, of a general matrix A is done by pre-multiplying A with a matrix given by:

$$
\begin{array}{c}
\begin{array}{ccccc} & & \mathrm{i} & & \mathrm{j} \end{array} \\
\begin{pmatrix}
1 & & \downarrow & & \downarrow & & \\
 & \ddots & & & & & \\
 & & 1 & & & & \\
 & & 0 & & 1 & & \leftarrow \mathrm{i} \\
 & & & 1 & & & \\
 & & & & \ddots & & \\
 & & & & & 1 & \\
 & & 1 & & 0 & & \leftarrow \mathrm{j} \\
 & & & & & 1 & \\
 & & & & & & \ddots \\
 & & & & & & 1
\end{pmatrix}
\end{array}
\tag{2.1}
$$

Post multiplication with this matrix interchanges the i-th and j-th columns.

Remark 2.1: From the experience gained in completing this work, it should be recommended to invest some time in trying to exploit these elementary orthogonal transformations to simplify or completely solve the reduction of a matrix to its condensed form. Since these transformations are so simple, this work could be done *exactly*, without rounding errors. As a consequence, the use of condensed forms then becomes *extremely attractive* for such cases.

The usefulness of remark 2.1 is demonstrated in the previous chapter for the state space model used to reconstruct the flight-path.

2.2.2. The Givens transformation

A second orthogonal transformation that will be presented here is the *Givens transformation*. This transformation is characterized by the following matrix:

$$G_{ij}(\theta) = \begin{array}{c} \begin{array}{ccccccccccc} & & & & \text{i} & & & & \text{j} & & \\ \end{array} \\ \begin{pmatrix} 1 & & & & \downarrow & & & & \downarrow & & \\ & \ddots & & & & & & & & & \\ & & 1 & & & & & & & & \\ & & & cos\theta & & & & sin\theta & & & \leftarrow \text{i} \\ & & & & 1 & & & & & & \\ & & & & & \ddots & & & & & \\ & & & & & & 1 & & & & \\ & & & -sin\theta & & & & cos\theta & & & \leftarrow \text{j} \\ & & & & & & & & 1 & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & 1 \end{pmatrix} \end{array} \tag{2.2}$$

The reflection angle θ is determined by $cos^{-1}(\frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}})$ and a_{ij} represents the entry on the i-th row and j-th column of the A-matrix.

Pre-multiplication of a general matrix A by (2.2) *only* affects the i-th and j-th row of this matrix. The reflection angle θ in (2.2) can be chosen such as to annihilate an (arbitrary but single) entry of these two rows. Usually, one choses to annihilate the leading non-zero element in the j-th row. And in this case the leading element of the i-th row is called the *pivot element*. This is for instance done in the so-called QR factorization algorithm of a general matrix, where Q is *orthogonal* and R is *upper triangular*. An algorithmic implementation of this procedure is given by the following algorithm. This can for instance also be found in [92]. However, it is briefly recalled here to introduce the notation characterizing this transformation, since it will often be used in the sequel.

Algorithm 2.1:

Given a general $m \times n$ matrix A, the algorithm uses the transformation matrices G_{ij} in (2.2), such that the product $A_m = \mathcal{G}_{m-1}.\mathcal{G}_{m-2} \ldots \mathcal{G}_1.A$ is *upper trapezoidal*. The $\mathcal{G}_k$'s are an accumulation of a number of Givens rotations G_{ij} and will be defined below. At the k-th step, the matrix $A_k = \mathcal{G}_{k-1} \ldots \mathcal{G}_1.A$ has the form illustrated below for $m = 6, n = 5$ and $k = 3$:

$$A_k = \left(\begin{array}{ccc|cc} x & x & x & x & x \\ 0 & x & x & x & x \\ \hline 0 & 0 & \mathbf{x} & x & x \\ 0 & 0 & x_1 & x & x \\ 0 & 0 & x_2 & x & x \\ 0 & 0 & x_3 & x & x \end{array}\right)$$

Now we choose $\mathcal{G}_k$ to introduce zeros below the k-th diagonal element. These elements are indicated by the subscripted entries. Each of these elements are annihilated by a transformation matrix G_{ij} given in (2.2) using the bold element x as pivot element and in the order denoted by the subscripts. In this way $\mathcal{G}_k$ becomes:

$$\mathcal{G}_k = G_{k6}(\theta_3).G_{k5}(\theta_2).G_{k4}(\theta_1)$$

where the θ_i's are chosen in order to annihilate x_i. Here, we remark that the order in performing these transformations can be chosen arbitrarily for a general matrix, as long as one does not refill previously zeroed elements. We could for instance annihilate the subscripted elements in reversed order. After $m-1$ steps the product $\mathcal{G}_{m-1}.\mathcal{G}_{m-2} \ldots \mathcal{G}_1$ and the matrix A_m represent the desired information of the QR factorization.

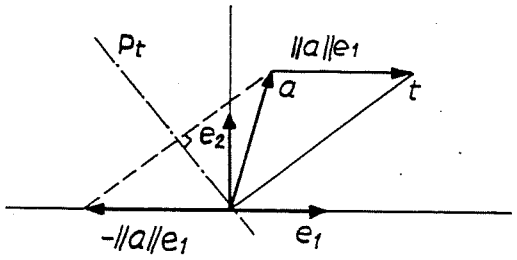
Post-multiplication acts in a similar way on the columns. When A is a square $n \times n$ matrix, this factorization requires $\frac{4}{3}n^3$ "flops", with 1 flop indicating 1 addition and 1 multiplication. On the other hand, the use of *Gaussian elimination* with partial pivoting only requires $\frac{1}{3}n^3$ flops [123].

Remark 2.2: For the case of pre-multiplication, elements with value zero occurring in both rows and having the same column number, remain zero after a Givens reflection between those rows.

Remark 2.2 may be useful to reduce the computational complexity of the use of Givens transformations or can allow to preserve some particular structure that the original matrix already possessed.

2.2.3. The Householder reflection

A third orthogonal transformation is the Householder reflection. This transformation is designed to map a chosen vector a onto a basis vector e_i of the natural orthogonal basis, but reflected from its initial orientation in the direction of that basis vector. This is schematically illustrated in figure 2.1 for the two-dimensional case. Again usually e_i is chosen to be the first basis vector e_1 of the natural orthogonal subspace one is working in. This is described in more detail in algorithm 2.2.



$\{e_1, e_2\}$ the natural orthogonal basis.

Figure 2.1: The Householder transformation in two-dimensions

From fig. 2.1 we observe that we may start by constructing the vector $t = a + \|a\|e_1$, i.e a vector with the same coordinates as a except for the first coordinate, which is $a_1 + \|a\|$. We then construct the plane P_t, perpendicular to t and passing through the origin. The mapping that reflects all vectors through this plane is given by the transformation, $H = (I - \frac{tt'}{t'a})$, where I denotes the identity matrix of appropriate order. Then, we have:

$$Ha = (I - \frac{tt'}{t'a})a = a - t = -\|a\|e_1 \qquad (2.3)$$

as desired. Since $HH' = I$, H represents an orthogonal transformation. For an arbitrary vector a, the vector t in (2.3) becomes $a + sign(a_1).\|a\|.e_1$. With $sign(a_1)$ denoting the sign of the first coordinate of the a vector in the natural orthogonal space.

This transformation can also be used to perform the QR factorization of a general matrix. An explicit algorithmic implementation is again given here in order to specify our notation.

Algorithm 2.2:

Given a general $m \times n$ matrix A, the algorithm uses the transformation matrix H given in (2.3), such that the product $A_m = H_{m-1}.H_{m-2} \ldots H_1.A$ is again *upper trapezoidal.* At the k-th step, the matrix $A_k = H_{k-1} \ldots H_1.A$ has the form illustrated below for $m = 6, n = 5$ and $k = 3$:

$$A_k = \left(\begin{array}{ccc|cc} x & x & x & x & x \\ 0 & x & x & x & x \\ \hline 0 & 0 & x & x & x \\ 0 & 0 & x_1 & x & x \\ 0 & 0 & x_1 & x & x \\ 0 & 0 & x_1 & x & x \end{array}\right)$$

Now we choose H_k to introduce zeros below the k-th diagonal element, namely the elements marked by the subscript 1. In contrast to the Givens transformation, the Householder reflection annihilates all subscripted elements at a time. This is indicated by the use of only one subscript. In more detail, at the k-th step, A_k has the form:

$$A_k = \left(\begin{array}{c|c|c} R_k & r_k & B_k \\ \hline 0 & c_k & D_k \end{array}\right)$$

where R_k is an upper triangular matrix of order $k-1$ and $c_k \in R^{(m-k+1)}$. The transformation matrix H_k only operates on the submatrix $(c_k\ D_k)$ and therefore has the form:

$$H_k = \left(\begin{array}{c|c} I_{k-1} & 0 \\ \hline 0 & \mathcal{H}_k \end{array}\right)$$

where $\mathcal{H}_k$ is constructed to reflect c_k accordingly:

$$\mathcal{H}_k = I_{m-k+1} - \frac{(c_k \pm \|c_k\|e_1)(c_k \pm \|c_k\|e_1)'}{(c_k \pm \|c_k\|e_1)'c_k}$$

with e_1 the first coordinate vector of the natural orthogonal basis for $R^{(m-k+1)\times(m-k+1)}$ and where $\pm$ indicates the sign of the first coordinate of c_k. After this transformation we obtain,

$$A_{k+1} = H_k.A_k = \left(\begin{array}{c|c|c} R_k & r_k & B_k \\ \hline 0 & \mp\|c_k\|.e_1 & \mathcal{H}_k.D_k \end{array}\right)$$

This carries the triangularization one step further. The new upper triangular matrix R_{k+1} in the following step now becomes:

$$R_{k+1} = \left(\begin{array}{c|c} R_k & r_k \\ \hline 0 & \mp\|c_k\| \end{array} \right)$$

Again after (m-1) steps the product $H_{m-1} \ldots H_1$ and R_m represents the requested information of the QR factorization. Executing this algorithm on a square $n \times n$ matrix requires $\frac{2}{3}n^3$ flops.

Remark 2.3: When *a part* of the column vector on which the Householder transformation is constructed is already zero, the corresponding rows of the transformed matrix remain *unaltered*

We end this section with the following general considerations:

1. From the above given number of flops needed to transform a general matrix to a triangular form, the Householder method is *a factor 2 faster* than the Givens reflection. This proportion is generally also observed in actual computing time.
2. In fact, the multiplication of the matrix that characterizes the different transformations, given by eq. (2.1-2.3), should never be performed explicitly. However, we didn't go into details concerning their algorithmic implementation, because many reliable (FORTRAN) implementations already exist, see e.g. LINPACK [19], which allow to use these transformations at a higher level.
3. We have indicated some interesting features of the different orthogonal transformations, which *in combination* may lead to a very fast and reliable reduction of a general matrix to a condensed form, or its QR factorization.

2.3. Condensed state-space models

The state space model of a linear system may be cast in the following general matrix notation (where λ here stands for the differential operator in the continuous time case and for the shift operator in the discrete time case):

$$\left\{ \begin{array}{rcl} \lambda x & = & Ax + Bu \\ y & = & Cx + Du \end{array} \right. \qquad (2.4)$$

Here the system matrices $\{A, B, C, D\}$, which define the system description are given for the time invariant case.

The state vector x (of dimension n) determines the *internal* behaviour of the system, while the *external* behaviour is summarized in the transfer function of the system. For the system defined by (2.4), the transfer function is,

$$C(\lambda I - A)^{-1}B + D \qquad (2.5)$$

This equation remains invariant under a *similarity transformation*, i.e. the definition of a new state vector (a new internal description) as,

$$x_t = T_1 x \qquad (2.6)$$

This may be easily verified by substituting the transformed system matrices $\{T_1 A T_1^{-1}, T_1 B, C T_1^{-1}, D\}$ in (2.5). Both system representations are often called *algebraically equivalent*. However, depending on the specific form of the matrices in $\{T_1 A T_1^{-1}, T_1 B, C T_1^{-1}, D\}$ some appealing properties towards the algorithmic implementation or accuracy of the numerical solution may be obtained. This idea in combination with transformations of the input u (of dimension m) and output y (of dimension p),

$$u_t = T_2 u \quad ; \quad y_t = T_3 y \qquad (2.7)$$

can be applied to a broad class of problems in system theory.

For a long time, the primary concern of scientists developing algorithms to solve theoretical problems in linear system theory was to use these transformations in order to derive a feasible or an attractive algorithm. For example, the transfer function calculation becomes *extremely simple* if A is reduced to a diagonal matrix. However, for general matrices this *can not* be done in a *numerical reliable way*. This simple example of the transfer function calculation illustrates very well the dilemma a system engineer is faced with. In this chapter a compromise between efficiency (or coming forward with a simple solution) and guaranteed reliability (or maintaining some numerical accuracy of the computed solution) will be presented by using "condensed forms".

Definition 2.1: The *condensed form* of a system quadruple $\{A, B, C, D\}$ is a form that contains as many zeroes as possible in the four system matrices, which are obtained by performing *unitary* state, input and output transformations.

If the *unitary* state, input and output transformations are denoted as:

$$x_t = Ux \ ; \ u_t = Vu \ ; \ y_t = Wy \tag{2.8}$$

respectively, then the corresponding quadruple of system matrices $\{A_t, B_t, C_t, D_t\}$ has the following form:

$$A_t = UAU^* \ ; \ B_t = UBV^* \ ; \ C_t = WCU^* \ ; \ D_t = WDV^* \tag{2.9}$$

where the superscript $*$ denotes the (conjugate) transpose.

In practice n is considerably larger than m and p and most of the zeroes will therefore be obtained in A_t by appropriately choosing U. Beside the numerical superiority of using unitary transformations as discussed in the introduction to transform a given problem to a more attractive formulation from the point of view of solvability, the back transformation can be easily done since the inverse of an orthogonal (unitary) matrix is simply its (conjugate) transpose.

The four main condensed forms encountered in the literature, which only require a state space transformation, are defined below. Apart from that we will give or refer to algorithms to obtain these forms.

Definition 2.2: When A_t is upper Hessenberg the quadruple $\{A_t, B_t, C_t, D\}$ is in the *Hessenberg* (condensed) *form*, illustrated for $n = 7, m = 3$ and $p = 2$:

$$\left(\begin{array}{c|c} B_t & A_t \\ \hline D & C_t \end{array}\right)_H = \left(\begin{array}{ccc|ccccccc} x & x & x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x & x & x \\ x & x & x & 0 & x & x & x & x & x & x \\ x & x & x & 0 & 0 & x & x & x & x & x \\ x & x & x & 0 & 0 & 0 & x & x & x & x \\ x & x & x & 0 & 0 & 0 & 0 & x & x & x \\ x & x & x & 0 & 0 & 0 & 0 & 0 & x & x \\ \hline x & x & x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x & x & x \end{array}\right) \tag{2.10}$$

Algorithm 2.3:

The Hessenberg form of A_t in (2.10) can be obtained with *orthogonal similarity transformations* in a direct way, i.e. after an a priori known number

of steps. Procedures based on the Givens- or Householder transformation to obtain this form are for instance given in [92] and [123].

Definition 2.3: When A_t is in upper Schur form, or upper triangular, the quadruple $\{A_t, B_t, C_t, D\}$ is in the *Schur* (condensed) *form*, again illustrated as:

$$\left(\begin{array}{c|c} B_t & A_t \\ \hline D & C_t \end{array}\right)_S = \left(\begin{array}{ccc|ccccccc} x & x & x & x & x & x & x & x & x & x \\ x & x & x & 0 & x & x & x & x & x & x \\ x & x & x & 0 & 0 & x & x & x & x & x \\ x & x & x & 0 & 0 & 0 & x & x & x & x \\ x & x & x & 0 & 0 & 0 & 0 & x & x & x \\ x & x & x & 0 & 0 & 0 & 0 & 0 & x & x \\ x & x & x & 0 & 0 & 0 & 0 & 0 & 0 & x \\ \hline x & x & x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x & x & x \end{array}\right) \tag{2.11}$$

Algorithm 2.4:

The first step in the Schur reduction is the transformation to the Hessenberg form by algorithm 2.3. The second step performs the triangularization. This occurs essentially *iteratively*. A reliable way to do this is based on the shifted (explicit or implicit) QR algorithm given by algorithm 2.7 and is described in [92] and [123]. The transformations used in this reduction are *unitary* when the real A matrix contain complex eigenvalues.

Definition 2.4: When the compound matrix $(B_t \mid A_t)$ is upper trapezoidal the quadruple $\{A_t, B_t, C_t, D\}$ is in the *controller-Hessenberg* (condensed) *form*, again illustrated as:

$$\left(\begin{array}{c|c} B_t & A_t \\ \hline D & C_t \end{array}\right)_{CH} = \left(\begin{array}{ccc|ccccccc} \overline{x} & \overline{x} & \overline{x} & x & x & x & x & x & x & x \\ 0 & \overline{x} & \overline{x} & \overline{x} & x & x & x & x & x & x \\ 0 & 0 & \overline{x} & \overline{x} & \overline{x} & x & x & x & x & x \\ 0 & 0 & 0 & \overline{x} & \overline{x} & \overline{x} & x & x & x & x \\ 0 & 0 & 0 & 0 & \overline{x} & \overline{x} & \overline{x} & x & x & x \\ 0 & 0 & 0 & 0 & 0 & \overline{x} & \overline{x} & \overline{x} & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & \overline{x} & \overline{x} & \overline{x} & x \\ \hline x & x & x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x & x & x \end{array}\right) \tag{2.12}$$

Algorithm 2.5: The first step of the algorithm computes:

$$H_1.(B \mid A).\left(\frac{I_m \mid 0}{0 \mid H_1'}\right) = \left(\frac{x \mid c'}{0 \mid A_t^1}\right)$$

where H_1 is an orthogonal transformation to annihilate the elements in the (n-1) bottom rows of the first column of the compound matrix $(B \mid A)$. The chosen post-transformation matrix $\left(\frac{I_m \mid 0}{0 \mid H_1'}\right)$ assures that the obtained zeros are not destroyed, since it does not affect the first m columns of $H_1.(B \mid A)$. After this step the matrix A_t^1 has dimensions $(n-1) \times (m + (n-1))$.
In the second step, we perform the same operation as in the first step on the submatrix A_t^1. Therefore, we construct a transformation H_2 to annihilate the elements of the (n-2) bottom rows of the first column of this submatrix. This transformation operates on the (n-1) bottom rows of the submatrix A_t^1. The post transformation than can be written as $\left(\frac{I_m \mid 0}{0 \mid H_2'}\right)$ operating on the last (n-1) columns of $H_2.A_t^1$. Therefore, again the obtained zeros in the first column of this submatrix are not destroyed. When recursively applying such transformations H_i it is easily seen that this last fact will remain true for all subsequently chosen (compound) sub-matrices. So that after (n-1) steps the form given in (2.12) results.

Remark 2.4: The choice of the name *controller Hessenberg* form in definition 2.4 may in some sense be obscure. This is because here the matrix A_t is no longer in Hessenberg form. The motivation for the choice of this name is that for the case $m = 1$ this is true. The cases $m > 1$ are then considered as an extension of this particular case.

Definition 2.5: When the compound matrix

$$\begin{pmatrix} A_t \\ C_t \end{pmatrix}$$

is upper trapezoidal the quadruple $\{A_t, B_t, C_t, D\}$ is in *observer-Hessenberg* (condensed) *form*, again illustrated as:

$$\left(\frac{B_t \mid A_t}{D \mid C_t}\right)_{OH} = \left(\begin{array}{ccc|ccccccc} x & x & x & x & x & x & x & x & x \\ x & x & x & \overline{x} & x & x & x & x & x \\ x & x & x & \overline{x} & \overline{x} & x & x & x & x \\ x & x & x & 0 & \overline{x} & \overline{x} & x & x & x \\ x & x & x & 0 & 0 & \overline{x} & \overline{x} & x & x \\ x & x & x & 0 & 0 & 0 & \overline{x} & \overline{x} & x \\ x & x & x & 0 & 0 & 0 & 0 & \overline{x} & \overline{x} \\ \hline x & x & x & 0 & 0 & 0 & 0 & 0 & \overline{x} \\ x & x & x & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right) \tag{2.13}$$

Algorithm 2.6:

Here we can use algorithm 2.5 for the compound matrix $(B \mid A)$ replaced by $(C' \mid A')$.

In the illustrations (2.10-2.13) the x's denote arbitrary elements. Notice that the first form has $\frac{(n-2)(n-1)}{2}$ zeroes, while all the other forms have the same number of zeroes, namely $\frac{n(n-1)}{2}$. Some slightly modified forms exist as well. In order to obtain the Schur form (2.11), one needs in general complex transformations even when A is real. A so-called *real Schur form* or *quasi-triangular form* for real A matrices will indeed have 2×2 bumps on the diagonal corresponding to the pairs of complex eigenvalues [25]. Its number of zeroes is thus between the forms (2.10) and (2.11). In the forms (2.12) and (2.13) some of the elements denoted by $\overline{x}$ can be zero. Therefore, the forms can be compressed further. This resulted in the so-called staircase forms, discussed in [10], [58], [73], [80], [83], [107], [108] and [119]. These variants all require rank tests to locate the non-zero elements. A possible algorithm to obtain the staircase form for the controller-Hessenberg form is given by Miminis and Paige [73] and is illustrated below, where the non-zero elements are denoted by X's:

$$\left(\begin{array}{c|c} B_t & A_t \\ \hline D_t & C_t \end{array}\right)_{CH} = \left(\begin{array}{ccc|ccccccc} 0 & X & x & x & x & x & x & x & x & x \\ 0 & 0 & X & x & x & x & x & x & x & x \\ 0 & 0 & 0 & X & x & x & x & x & x & x \\ 0 & 0 & 0 & 0 & X & x & x & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & X & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & X & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & X & x \\ \hline x & x & x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x & x & x \end{array}\right) \quad (2.14)$$

Here we always have more than $\frac{n(n-1)}{2}$ and less than $\frac{n(n+2m-1)}{2}$ zeroes, and less than $\frac{n(n+2p-1)}{2}$ for the equivalent staircase form of (2.13). Finally, one can construct *lower* condensed forms corresponding to all the above *upper* forms, by merely working on the *dual* quadruple $\{A^*, C^*, B^*, D^*\}$. Notice also that only the staircase forms require more than a state transformation U: an input transformation V is needed to obtain form (2.14) and an output transformation W is needed for the staircase form corresponding to (2.13).

The algorithms 2.3 till 2.6 to yield these different forms all have a comparable complexity. The number of flops required for the forms (2.10), (2.11), (2.12), (2.13) and (2.14) roughly are $n^2(3n+m+p)$, $n^2(5kn+p+m)$, $n^2(3n+m+p)$, $n^2(3n+m+p)$ and $n^2(3n+6m+p)$ respectively. Here, k is the average number of QR steps require to find an eigenvalue of a matrix in Hessenberg form, which is usually between 1 and 2 [122]. These operation counts include the construction of the state space transformation U (and if needed also V or W) to yield the corresponding condensed form.

2.4. Survey and unification of exploiting condensed forms

2.4.1. Some applications in numerical analysis

The different condensed forms discussed in the previous section were already in common use for a long time in numerical linear algebra [123]. Here, these different forms were generally used as part of an algorithm.

Specific techniques used herein will now briefly be described. This is because they have (had) great importance in linear system theory, as will be shown in the next subsection.

2.4.1.1. Updating Schur forms

A general matrix can be transformed by unitary (orthogonal) similarity transformations to a (real) Schur form, with the eigenvalues (or 2×2 bumps corresponding to a pair of complex eigenvalues) on its diagonal [92], [123]. For a full $n\times n$ matrix this takes order n^3, denoted as $O(n^3)$, operations. To order two adjacent eigenvalues of a Schur form reversely, again orthogonal similarity transformations could be used. This is demonstrated for the following example.

Example 2.1: For the following 2×2 triangular matrix, having its eigenvalues on the diagonal:

$$A = \begin{pmatrix} \lambda_1 & x \\ 0 & \lambda_2 \end{pmatrix} \quad (2.15)$$

the transformation matrix can be chosen accordingly to:

$$T = \begin{pmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{pmatrix} \quad (2.16)$$

with $tan\theta = \frac{(\lambda_2 - \lambda_1)}{x}$ to obtain the desired result:

$$TAT^* = \begin{pmatrix} \lambda_2 & x \\ 0 & \lambda_1 \end{pmatrix} \quad (2.17)$$

Applying (2.16) to an $n \times n$ matrix A in triangular form, takes $O(n)$ operations. In this way, one can order the eigenvalues of a Schur form in an arbitrary way by $O(kn)$ operations, where k is the number of "exchanges" of two adjacent eigenvalues in order to obtain the requested ordering.

2.4.1.2. Performing an implicit shifted QR step

The explicit form of the shifted QR algorithm is summarized in the following algorithm. This can e.g. also be found in [92].

Algorithm 2.7:

Let $A = A_0$ be an $n \times n$ matrix. The k-th iteration of the shifted QR

algorithm to find a single eigenvalue is given by the following two steps.

Step 1:

Given A_k, compute the QR factorization as follows:

$$A_k - x_k I = Q_k R_k \tag{2.18}$$

where a "good" choice of the shift x_k, given in [92] on p. 351, simply takes the diagonal element of the last row of A_k.

Step 2:

Compute A_{k+1} as:

$$A_{k+1} = R_k Q_k + x_k I \tag{2.19}$$

When A_0 is in Hessenberg form, A_k will remain in this form and the QR factorization (2.18) can be performed very efficiently. Note that from (2.18) we have $R_k = Q_k^*(A_k - x_k I)$ and hence (2.19) becomes,

$$A_{k+1} = Q_k^* A_k Q_k \tag{2.20}$$

So,that A_{k+1} is indeed unitarily similar to A_k. The above described procedure is generally performed implicitly. That implementation is therefore indicated as the implicit shifted QR method. This relies on *the fact* that if both A_{k+1} and A_k are in Hessenberg form, A_{k+1} and Q_k are uniquely determined by the first column of Q_k. (see Theorem 4.2 in [92] on p. 368). Now the shift x_k determines this first column completely.

The above described algorithms work *iteratively*. However, when the eigenvalues of A_0 are known a priori, they allow to compute the Schur form very efficiently with a prescribed ordering of the eigenvalues on its diagonal. For, if λ_1 is an eigenvalue of A_0 and if one applies the above described QR algorithm with shift $x_k = \lambda_1$, A_1 then becomes (see [92] for a proof of this):

$$A_1 = \left(\begin{array}{cccccc|c} x & x & x & x & x & x & x \\ x & x & x & x & x & x & x \\ 0 & x & x & x & x & x & x \\ 0 & 0 & x & x & x & x & x \\ 0 & 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & 0 & x & x & x \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 \end{array}\right) \tag{2.21}$$

The complexity of this single step is $O(n^2)$. Subsequently, one operates on the $(n-1) \times (n-1)$ Hessenberg submatrix in (2.21) to isolate λ_2.

2.4.1.3. Further elimination to obtain a QR decomposition

The amount of computations needed to obtain a QR decomposition of an $m \times n$ matrix may decrease when the matrix already has a special structure. E.g. when this matrix contains only k subdiagonals, the complete QR decomposition roughly requires kn^2 operations. When the QR decomposition is used for inverting square matrices or solving least squares problems, this can be done very economically if the original matrix has only a few subdiagonals (i.e. $k << n$).

2.4.2. A number of applications in system theory

2.4.2.1. Linear state variable feedback

One of the first applications of state-space methods for linear systems was that of using state feedback, where the input is defined as $u = -Lx$, to modify the poles of the closed loop system. More precisely, one can formulate the problem as follows.

If a system $\{A, B\}$ of order n, being the dimension of the state vector x, is *controllable*, then find the state feedback matrix L of dimensions $m \times n$ to allocate the eigenvalues of the closed loop system, with system matrix $(A + BL)$, at prescribed values $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_n\}$.

Here, we describe a few methods which exploit condensed forms in solving this problem.

Algorithm 2.8:

A *first* method, due to Varga [112], uses the Schur form (2.11) of the given system. By performing a feedback L_1 with only non-zero elements in the last column, the closed loop system matrix $(A_s + B_s L_1)$ is still in (upper) Schur form. With the index s denoting the Schur form. An appropriate choice of L_1 allows to assign the bottom eigenvalue of this matrix to an arbitrary value (e.g. λ_1). One of course cannot use the other columns of L_1 to place the other eigenvalues of the desired spectrum Λ, *without* destroying the triangular form of $(A_s + B_s L_1)$. But instead, one can use the *updating Schur form technique*, described in section 3.4.1.1, to move the assigned eigenvalue λ_1 to the (1,1) position of the Schur form by a unitary transformation U_1. In this new coordinate system one can now use a second feedback L_2, again with only non zero elements in the last column, to assign the bottom eigenvalue to λ_2. The closed loop system matrix in this new

coordinate system is $U_1\{A_s + B_s(L_1 + L_2U_1)\}U_1^*$. This newly assigned eigenvalue is then moved in turn to the (2,2) position of the Schur form by an updating unitary transformation U_2. This process is repeated until all n eigenvalues λ_i are assigned. The feedback matrix L then becomes:

$$L = L_1 + L_2U_1 + \ ... \ + L_n(U_{n-1}...U_1) \tag{2.22}$$

This method works as well for single input as for multi-input systems and its complexity is $O(n^3)$.

Algorithm 2.9:

A *second* method, developed by Miminis and Paige [73], uses the controller-Hessenberg form (2.12). This method relies on the shifted QR algorithm to determine the eigenvalues of a given matrix. The method consists of two parts, a forward and a backward sweep. These will be briefly described for single input systems. From (2.12) it is clear that the closed loop system matrix $(A_{ch} + b_{ch}\ell)$ is completely known except for its first row, illustrated for n = 5:

$$A_1 = A_{ch} + b_{ch}\ell = \begin{pmatrix} * & * & * & * & * \\ x_4 & \mathbf{x_4} & x & x & x \\ 0 & x_3 & \mathbf{x_3} & x & x \\ 0 & 0 & x_2 & \mathbf{x_2} & x \\ 0 & 0 & 0 & x_1 & \mathbf{x_1} \end{pmatrix} \tag{2.23}$$

where the $*$ denotes the unknown entries and x the known ones.

Step 1 of the presented shifted QR algorithm (algorithm 2.7) can now be preformed reversely. Now the shift λ_1 is given, because it has to be an eigenvalue of A_1. And the QR factorization becomes, $(A_1 - \lambda_1 I)Q_1 = R_1$. Since $(A_1 - \lambda_1 I)$ has a similar structure as illustrated in (2.23), the orthogonal transformation Q_1 can be determined completely. This can for example be done using Givens rotations. The order of annihilation is indicated in (2.23) by the subscripts and the corresponding pivot elements are indicated in bold. On the other hand, the fact that λ_1 has to be an eigenvalue of A_1 and is not an eigenvalue of A_{ch}, ensures that the (1,1) top element of R_1 must be zero. Therefore R_1 is also known except for its first row:

$$R_1 = \begin{pmatrix} 0_4 & * & * & * & * \\ 0_4 & x_3 & x & x & x \\ 0 & 0_3 & x_2 & x & x \\ 0 & 0 & 0_2 & x_1 & x \\ 0 & 0 & 0 & 0_1 & x \end{pmatrix} \tag{2.24}$$

The information of Givens transformation allows us now to perform step 2 of the shifted QR algorithm, i.e. the computation of $Q_1^*R_1 + \lambda_1 I$. The subscripts of the elements in (2.24) denote the order in which the Givens transformations are applied. And they also correspond to the transformations used in (2.23) to annihilate the elements with the same subscript. This now leads to the the following form:

$$\begin{pmatrix} \lambda_1 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{pmatrix} = \begin{pmatrix} \lambda_1 & d_1' \\ 0 & A_2 \end{pmatrix} \tag{2.25}$$

or if the previous step allocated λ_k (2.25) can be denoted more generally as:

$$\begin{pmatrix} \lambda_k & d_k' \\ 0 & A_{k+1} \end{pmatrix} \tag{2.26}$$

where d_k' is unknown and A_{k+1} is an Hessenberg matrix with a similar structure as illustrated by (2.23). With this submatrix A_{k+1} steps (2.23) to (2.25) can be repeated until λ_{n-1} is allocated. However, for this last step $A_n = \lambda_n$, which now has a known top row. With this information it is possible to reconstruct all the missing information of the previous steps using the orthogonal reductions Q_k, and this going from $k = n-1, \ ... \ , 1$. This step is called the backsweep. For k=1 the first row in (2.23) can be calculated and *hence* also ℓ. The complexity is here again $O(n^3)$. The extension of this procedure to the multivariable case is described in [73].

A *third* method, due to Kautsky, Nichols and Van Dooren [56], is specifically designed for multi-input systems, and uses the degrees of freedom that are still available in the choice of L, to minimize the *sensitivity to perturbations* of the *assigned eigenvalues* λ_i. In a sense it thus tries to find a solution that is as robust as possible. A general measure and upper bound for the sensitivities of the eigenvalues is given by the *condition number* of the matrix $\mathcal{X}$ defined in the relation, $(A + BL)\mathcal{X} = \mathcal{X}\Lambda$ [123]. In [56]

it is proved that the eigenvector x_i, comprising $\mathcal{X}$ and which corresponds to each assigned eigenvalue λ_i, must belong to the right null space $\mathcal{S}_i$ of the matrices $(A_{21} \mid A_{22} - \lambda_i I)$, where A_{21} and A_{22} are submatrices of the staircase form (2.14):

$$\left(\begin{array}{c|c|c} B_1 & A_{11} & A_{12} \\ \hline 0 & A_{21} & A_{22} \end{array}\right) = \left(\begin{array}{ccc|ccc|ccccc} 0 & X & x & x & x & x & x & x & x & x \\ 0 & 0 & X & x & x & x & x & x & x & x \\ \hline 0 & 0 & 0 & X & x & x & x & x & x & x \\ 0 & 0 & 0 & 0 & X & x & x & x & x & x \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & X & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & X & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & X & x \end{array}\right) \tag{2.27}$$

The null spaces $\mathcal{S}_i$ are obtained by computing the RQ decomposition of the matrices $(A_{21} \mid A_{22} - \lambda_i I)$ which can be done cheaply because these matrices are nearly upper triangular. After that, the core of the *iterative algorithm* consists of updating the choices of the eigenvectors x_i from these spaces $\mathcal{S}_i$ in order to minimize a certain sensitivity function, which expresses that the condition number of $\mathcal{X}$ should be as small as possible. This optimization problem would then become very expensive if one would not extensively use condensed forms. Since, here again updating QR decompositions of nearly triangular are required matrices, when using condensed forms. This makes it possible to obtain an algorithm yielding a satisfactory solution within $O(n^3 m)$ operations.

2.4.2.2. Lyapunov and Riccati equations

Lyapunov equations are used in various problems of linear system theory. For example *to check if a system is stable* without computing the eigenvalues. For a system with state space description $\{A, B, C, D\}$ and for a given matrix Q, solving the *matrix Lyapunov equation* consists in finding an $n \times n$ matrix P which satisfies:

$$A^*P + PA = -Q \tag{2.28}$$

in the continuous time case, and for the discrete time case:

$$P - A^*PA = Q \tag{2.29}$$

A discrete time system is *asymptotically* stable, i.e. $|\lambda_i(A)| < 1$, if for some choice of a nonnegative Q, the solution P of (2.29) exists, is unique and positive definite. Conversely, if a positive definite solution P exists, it is unique and $|\lambda_i(A)| < 1$.

The solution of eq. (2.28-2.29) can be determined efficiently with methods described in [4], [6] and [38]. The computational complexity is $O(n^3)$. These methods all use the Schur form of the A matrix in order to solve for P [4], [6] or its upper triangular Choleski factor [38] in a fast *recursive* manner. On the other hand, eq. (2.28-2.29) may be written as a set of linear equations in the unknown entries of P by using the sparse *Kronecker product*. Solving this set of equations indeed results in a slower method, even when using sparse matrix techniques.

The solution of the matrix Lyapunov equation can also be used in a recursive method to solve the *algebraic Riccati equation*. This will be demonstrated for the continuous time case. For this case, the equation to be solved is:

$$0 = A^*P + PA - PBR^{-1}B^*P + C^*C \tag{2.30}$$

where R is a given positive definite weighting matrix.
The solution P of this equation can be determined by solving recursively equations of the type [39]:

$$A_k^* P_{k+1} + P_{k+1} A_k = -(C^*C + P_k B R^{-1} B^* P_k) \tag{2.31}$$

where

$$A_k = A - BR^{-1}B^*P_k \tag{2.32}$$

If we choose the system in controller Hessenberg form (2.12) then A_k will have the same pattern of zeros as A. When $m = 1$, this yields automatically A_k in Hessenberg form. This form (which is required in the Schur form reduction of A_k) then does not have to be computed anymore.

2.4.2.3. Calculation of the transfer function, poles and zeros

Some interesting applications of linear system theory (still) have to be formulated in the frequency domain, e.g. the generalized optimal control problem [59], the multivariable root locus problem [60], [70]. Starting from our basic state space representation, these concepts could be used if one disposes of reliable tools to extract the required frequency domain information. In this section, the use of condensed forms is discussed to compute

reliably the transfer function, the poles and zeros of the linear system, defined by eq. (2.4)

a. The transfer function:

The transfer function of the system given by the quadruple $\{A, B, C, D\}$ is the $p \times m$ rational matrix,

$$R(\lambda) = C(\lambda I - A)^{-1}B + D \tag{2.33}$$

A single entry of this matrix is given by,

$$r_{ij}(\lambda) = c_i(\lambda I - A)^{-1}b_j + d_{ij} \tag{2.34}$$

where c_i is the i-th row of the C matrix and b_j is the j-th column of the B matrix.

Therefore, the calculation of the transfer function for the multivariable case could be recast to solving the basic problem for scalar systems. A reliable way to solve this basic problem [21], [113] is to compute the gain k_{ij}, the poles β_k^{ij} and the zeros α_k^{ij} of the scalar system $\{A, b_j, c_i, d_{ij}\}$. With this information (2.34) becomes:

$$r_{ij}(\lambda) = k_{ij}\frac{\prod_k(\lambda - \alpha_k^{ij})}{\prod_k(\lambda - \beta_k^{ij})} \tag{2.35}$$

The missing quantities $\{k_{ij}, \alpha_k^{ij}, \beta_k^{ij}\}$ in (2.35) can be computed by the following algorithm.

Algorithm 2.10:

A method, due to Varga and Sima [113], to compute these three quantities can be summarized in the following four steps:

Step 1:

In the first step the quadruple $\{A, b_j, c_i, d_{ij}\}$ is reduced to a *minimal realization* in order to eliminate all eventual pole-zero cancellation. Therefore, the controllable and observable part of the considered quadruple is extracted. This can be done by an orthogonal similarity transformation Q. The resulting quadruple $\{Q^*AQ, Q^*b_j, c_iQ, d_{ij}\}$ has the same pole-zero

structure. The calculation of the controllable part is done by transforming the pair $(b_j \mid A)$ in staircase controller Hessenberg form (2.14). During this reduction a rank detection method is used, see [113] to deduce the form illustrated below:

$$\left(\begin{array}{c|cccc|c} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ \hline 0 & 0 & 0 & 0 & 0 & x \\ \hline x & x & x & x & x & x \end{array}\right) = \left(\begin{array}{cc|c} \tilde{b}_j & \tilde{A} & x \\ \hline 0 & 0 & x \\ \hline \tilde{d}_{ij} & \tilde{c}_i & x \end{array}\right)$$

The observable part of $\{\tilde{A}, \tilde{b}_j, \tilde{c}_i, \tilde{d}_{ij}\}$ can be determined via the same procedure operating on the dual system $\{\tilde{A}', \tilde{c}_i', \tilde{b}_j', \tilde{d}_{ij}\}$. The minimal system then has the following structure:

$$\left(\begin{array}{c|c} \hat{A} & \hat{b}_j \\ \hline \hat{c}_i & \hat{d}_{ij} \end{array}\right) = \left(\begin{array}{cccc|c} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ \hline 0 & 0 & 0 & x & x \end{array}\right) \tag{2.36}$$

which is clearly the observer-Hessenberg form (2.13)

Step 2:

The system matrix $\hat{A}$ of (2.36) is used to compute the *poles* of $r_{ij}(\lambda)$. Because the realization is minimal (as guaranteed by the first step) the eigenvalues of $\hat{A}$ coincide with the poles. Since $\hat{A}$ is *already* in Hessenberg form, which is the preliminary step towards the eigenvalue computation, the pole calculation becomes very efficient.

Step 3:

The zeros of $r_{ij}(\lambda)$ are the values of λ for which (2.34) becomes zero. Analogously, these can be defined as the values of λ for which the associated *square* pencil of the transfer function r_{ij}, i.e.

$$\begin{pmatrix} \lambda I - \hat{A} & \hat{b}_j \\ \hat{c}_i & \hat{d}_{ij} \end{pmatrix} \tag{2.37}$$

becomes *singular.*
This allows to compute the zeros by solving a generalized eigenvalue problem. This becomes clear when (2.37) is formulated as $T - \lambda S$. The solution to this problem will be discussed in more detail in considering the computation of the zeros of a multivariable system in general.

Step 4:

The computation of the gain k_{ij} is performed by straightforward algebraic computations [113]:

$$k_{ij} = (\hat{c}_i(\lambda_0 I - \hat{A})^{-1}\hat{b}_j + \hat{d}_{ij})\frac{\prod_k(\lambda_0 - \beta_k^{ij})}{\prod_k(\lambda_0 - \alpha_k^{ij})} \tag{2.38}$$

for $\lambda_0 > max\{\alpha_k^{ij}, \beta_k^{ij}\}$

b. Poles and zeros:

The explicit form of the transfer function $R(\lambda)$ is not always required. An example is the root locus method, and its recent generalization to the multivariable case [60], [70]. Here, only the poles and zeros of $R(\lambda)$ are required. Recall for scalar systems, with a numerator degree of the transfer function $m < n$, m of the closed loop poles will converge to the m finite zeros as the feedback gain goes to infinity, while the remaining $n - m$ poles will converge to infinity. The use of the same ideas in the multivariable case require a proper definition of poles and zeros along with a reliable computation. The *poles* in the multivariable case can still be taken as the eigenvalues of the system matrix A. The definition of the *zeros* is less straightforward. Common in use are the *Smith zeros*, which fit in the framework of the multivariable root locus method. These are defined, see [29] and [87], as the values of λ for which the rank of $R(\lambda)$ in (2.33) drops below its normal rank, being the rank of the matrix $R(\lambda)$ for almost all values of λ. Equivalently, for these values the rank of the associated pencil $S(\lambda)$, given as:

$$S(\lambda) = \left(\begin{array}{c|c} \lambda I_n - A & B \\ \hline -C & D \end{array}\right) \tag{2.39}$$

also drops below its normal rank. A reliable method, due to Emami-Naeini and Van Dooren [21] first condenses the matrix pencil $S(\lambda)$ by orthogonal

similarity transformations on the system quadruple $\{A, B, C, D\}$ to a pencil of the form:

$$\left(\begin{array}{c|c} \lambda I_{\overline{n}} - \overline{A} & \overline{B} \\ \hline -\overline{C} & \overline{D} \end{array}\right) \tag{2.40}$$

where $\overline{D}$ is square and non-singular. For this pencil the zeros coincide with the eigenvalues of $\overline{A} - \overline{B}\,\overline{D}^{-1}\overline{C}$. This coincidence should not be used for the computation of the zeros, because of the possible bad conditioning of $\overline{D}$. Instead, one further reduces (2.40) via orthogonal transformations to:

$$\left(\begin{array}{c|c} \lambda B_f - A_f & \star \\ \hline 0 & D_f \end{array}\right) \tag{2.41}$$

where D_f will be invertible [21].
The (finite) zeros are then determined by solving the generalized eigenvalue problem, i.e. reducing $(\lambda B_f - A_f)$ to its generalized Schur form [21], [119].

Remark 2.5: The use of condensed forms (2.12) and (2.13) becomes extremely interesting for computing the poles and zeros *when* p=m=1. In both cases the system matrix A is then in Hessenberg form, which is the first step towards the Schur decomposition and thus the computation of the transfer function. If (2.13) is used, the pencil (2.39) is now also in generalized Hessenberg form. This form also resulted in the determination of the minimal realization in the explicit calculation of the transfer function, see (2.36). This particular structure of the pencil $S(\lambda)$ is the first step towards the generalized Schur decomposition. Furthermore, the the special structure of the pencil $S(\lambda)$ due to the use of (2.36) leads to a number of interesting observations. In [21] and [113] it is shown that for this structure of $S(\lambda)$, denoted as,

$$S(\lambda) = \left(\begin{array}{c|c} \lambda I - A & B \\ \hline -C & D \end{array}\right) = T - \lambda S$$

one can readily extract *the output decoupling zeros*, i.e. the Smith zeros of $\left(\begin{array}{c} \lambda I - A \\ \hline -C \end{array}\right)$, and the zeros at infinity by merely checking whether some of the elements of T in $T - \lambda S$ are zero. A similar result holds for the input decoupling zeros, i.e. the Smith zeros of $(\lambda I - A \mid B)$, when using the condensed form (2.12). By this inspection, the dimension of the (generalized) eigenvalue problem for the poles (zeros) of $S(\lambda)$ becomes smaller.

2.4.2.4. Frequency response and system response

The frequency response can be derived from the state space representation (2.4) by the following relationships:

a. for continuous time: $H_c(j\omega_k) = C(j\omega_k - A)^{-1}B + D$ (2.42)

b. for discrete time : $H_d(e^{j\omega_k}) = C(e^{j\omega_k} - A)^{-1}B + D$ (2.43)

In [61] it is shown that using the Hessenberg form (2.10) reduces considerably the amount of work to compute the frequency response for several values $k = 1, ..., N$. The inverses of the shifted A matrices in (2.42) and (2.43) normally require indeed $O(n^3)$ operations. When, on the other hand, A is in Hessenberg form the shifted A matrix is also in this form and each inverse only requires $O(n^2)$ operations. Let indeed $Q_k.R_k$ be the QR factorization of these shifted matrices in (2.42) or (2.43) then:

$$H_k = (CR_k^{-1})(Q_k^*B) + D \tag{2.44}$$

which takes $\frac{pn^2}{2}$ operations for computing CR_k^{-1}, $4mn$ for Q_k^*B (since Q_k is in fact a sequence of Givens reflections) and pmn for finally obtaining H_k. When p is small, the QR factorization takes most of the work. This can be avoided by using the Schur form (2.11). The computation of (2.42) and (2.43) always involves imaginary arithmetic. If N is very high (say more than 1000), this might still be far too expensive and using the explicit transfer function description such as defined previously will yield an $O(n)$ method.

For the computation of the system response (e.g. the step response, impulse response, transient response, etc.) one has to evaluate expressions of the type:

a. for continuous time: $y(T) = \int_o^T Ce^{A(T-t)}Bu(t)dt +$

$$Ce^{AT}x(0) \tag{2.45}$$

b. for discrete time : $y_k = \sum_{i=1}^{k} CA^{k-i}Bu(i-1) + CA^k x_0$ (2.46)

For (2.46) one has to build up the consecutive powers of A, which is cheaper for A in Schur form and, to a lesser extent, for A in Hessenberg form. For computing the exponential of a matrix in (2.45) the method of Padé approximation is particularly indicated [76], [114]. Its implementation also requires the accumulation of products of matrices which can have up to half of their elements equal to zero when a condensed form is chosen for $\{A, B, C, D\}$. For the convolution in (2.45) a numerical integration method for ordinary differential equations can be used, which again is more economical when e.g. the condensed form (2.12) is chosen, as used in [115]. Finally, the same holds also for the computation of integrals involving exponentials [114], which arise e.g. in the construction of the Gramians.

2.4.3. State observation by means of Luenberger observer

The power of state variable feedback, as discussed in section 3.4.2.1, relies on the availability of the complete state variables. In many practical applications only some linear combinations of the state are known. This is mathematically expressed by the second part of eq. (2.4). Therefore, methods have been developed *to reconstruct or observe* the complete state or missing state variables, indicated as the full order or reduced order observer problem. Pioneering work in this area has been performed by Kalman [51] and Luenberger [67]. The flight-path reconstruction problem, formulated in the first chapter, fits in the framework of both approaches. This subsection will start with the investigation of the Luenberger observer as a possible candidate to solve that particular problem. On the other hand, from the point of view of developing efficient and reliable algorithms this topic fits very well in this chapter. Since condensed forms also here have allowed us to derive new algorithms for the Luenberger observer problem. Furthermore, it can be considered as an ideal introduction to the more complicated Kalman filter problem that will be treated extensively in the following chapters.

2.4.3.1. The full order observer

The full order observer problem can be formulated as follows. For a linear system defined by (3.4) (without loss of generality the D matrix is left out in the notation here) one can construct a full-order observer:

$$\lambda z = Fz + Ky + Pu \tag{2.47}$$

using the input u and output y of system (2.4). The system (2.47) has

state dimension n. Under the assumption that the multivariable state-space system $\{A, B, C\}$ is observable the state quantities z may converge asymptotically to a particular relation between the original state variables x, denoted as Tx where T is a non-singular transformation matrix. This can be written mathematically as:

$$z \longrightarrow Tx \tag{2.48}$$

In order to obtain this, F, K, P and T have to be chosen appropriately. The full order observer configuration is illustrated in figure 2.2.

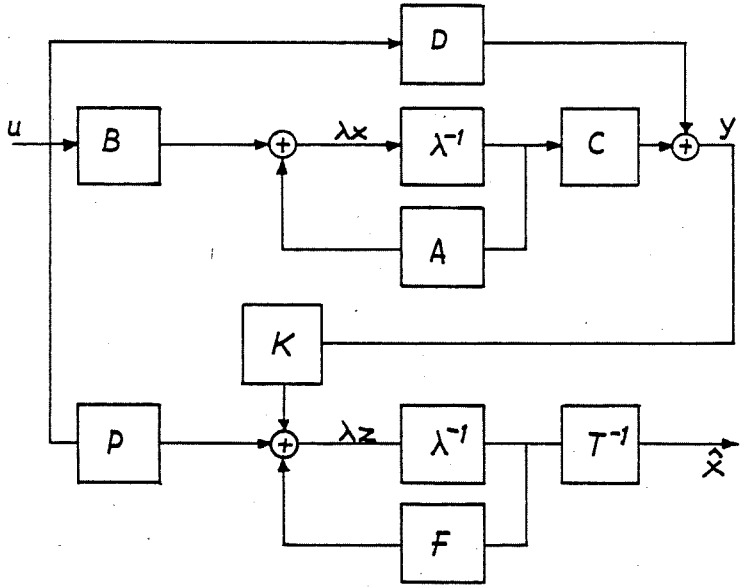


Figure 2.2: A block scheme of the full order observer

Luenberger pointed out [67] that by putting $P = TB$ for a specific choice of the matrices F and K the transformation matrix T in (2.48) can be determined by solving the following (*Sylvester*) equation:

$$TA - FT = KC \tag{2.49}$$

Generally, the matrix K is chosen arbitrarily and the matrix F is chosen so that its spectrum satisfies the following two conditions. First, it must be *stable*. In this case, convergence of the full-order observer is guaranteed, as can easily be observed from the state equations of the observer error $(Tx - z)$:

$$\lambda(Tx - z) = F(Tx - z) \tag{2.50}$$

which can be obtained by simple algebraic manipulations, using (2.4), (2.47) and (2.49). And secondly, in the original paper of Luenberger [67] it was required that the spectrum F differed from that of A in order to find a solution T for (2.49). This second condition on the spectrum of F was no longer required in later versions of the solution of (2.49) given by Luenberger [68]. Also for the algorithms given in the sequel this restriction is not necessary.

A solution of (2.49) can be obtained efficiently if A and F are put in condensed forms. This amounts to state-space transformations of the original system (2.4) and the observer (2.47), reducing (2.49) to an *equivalent numerical problem*:

$$T_{vu}A_u - F_v T_{vu} = K_v C_u \tag{2.51}$$

with

$$T_{vu} = VTU^* \;\; ; \;\; A_u = UAU^*$$

$$F_v = VFV^* \;\; ; \;\; K_v = VK \;\; ; \;\; C_u = CU^*$$

This idea is exploited in the following algorithm.

Algorithm 2.11:

Here A_u is chosen to be in upper Hessenberg and F_v in the lower Schur form, as proposed in [31]. For this choice (2.51) has the following structure, illustrated for $n = 6$:

$$T_{vu}\begin{pmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{pmatrix} - \begin{pmatrix} \lambda_1 & 0 & 0 & 0 & 0 & 0 \\ x & \lambda_2 & 0 & 0 & 0 & 0 \\ x & x & \lambda_3 & 0 & 0 & 0 \\ x & x & x & \lambda_4 & 0 & 0 \\ x & x & x & x & \lambda_5 & 0 \\ x & x & x & x & x & \lambda_6 \end{pmatrix} T_{vu} =$$

$$\begin{pmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \end{pmatrix} \tag{2.52}$$

From this form one now easily solves for T_{vu} row by row. Starting from the top row. If this first row is denoted by the row vector t_1 and kc_1 represents the first row of $K_v C_u$ in (2.51), then t_1 is solved from the following set of equations [37]:

$$t_1(A_u - \lambda_1 I) = kc_1 \tag{2.53}$$

A special class of full order observers is the *identity observer*. In this case the transformation matrix T is taken the identity matrix. From (2.49), the F matrix then becomes:

$$F = A - KC \tag{2.54}$$

Working with the dual system $\{A', C'\}$ one can use the procedure of section 3.4.2.1 to allocate the eigenvalues of F arbitrarily by a proper calculation of K.

2.4.3.2. The reduced order observer

The full order observer does not completely exploit the availability of some linear combinations of the state quantities, mathematically expressed as $y = Cx$. Using this information, it is more efficient to look for additional and independent combinations $z \rightarrow Tx$, such that the compound matrix $\begin{pmatrix} T \\ C \end{pmatrix}$ is *invertible*. This is the philosophy of the reduced order observer. Luenberger pointed out in [68] that the structure of this problem is very similar to the full order observer problem. The observer dynamics are governed by the same state equations (2.47), with state dimension (n-p). Here it is assumed that C has full row rank. The $(n-p) \times n$ transformation matrix T satisfies (2.49) with the extra constraint that:

$$\begin{pmatrix} T \\ C \end{pmatrix} \quad \text{is invertible} \tag{2.55}$$

With the observer input matrix P again taken as TB, the *observed state* is given by:

$$\hat{x} = \begin{pmatrix} T \\ C \end{pmatrix}^{-1} \begin{pmatrix} z \\ y \end{pmatrix} = (G \mid H) \begin{pmatrix} z \\ y \end{pmatrix} \tag{2.56}$$

Figure 2.3 gives a schematic representation of the reduced order observer.

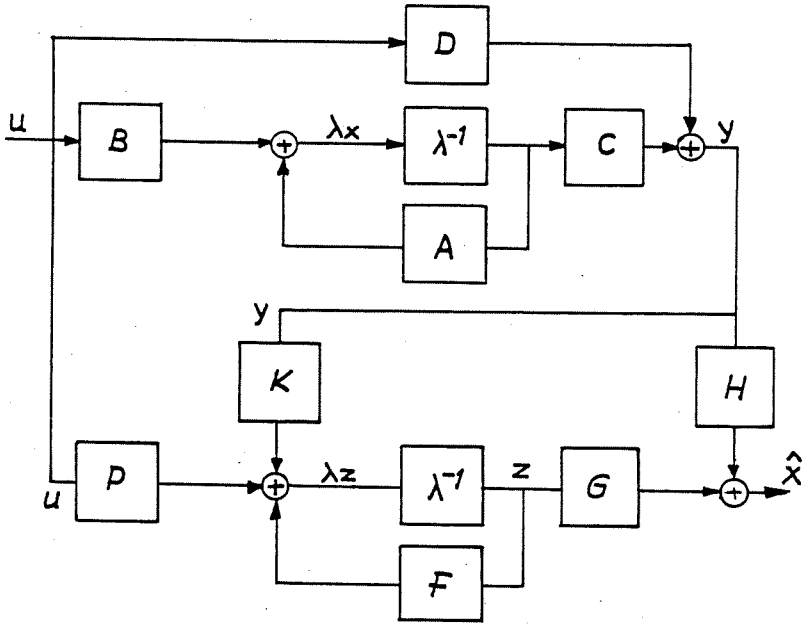


Figure 2.3: A block scheme of the reduced order observer

The solution can be formulated analogously to the full order observer problem. However, a description would be given of an alternative method in the following algorithm due to P. Van Dooren [116].

Algorithm 2.12:

Here we use the staircase observer Hessenberg form of (2.13) for the pair

(A, C) and the Schur form of F in (2.47). Furthermore, the choice of freedom in T_{vu}, F_v and K_v is restricted in this algorithm in specifying the spectrum of F_v and the structure of T_{vu}. The latter is taken as follows:

$$T_{vu} = \begin{pmatrix} 1 & & t'_1 & & \\ 0 & 1 & & t'_2 & \\ 0 & 0 & 1 & & t'_3 \\ . & . & . & . & . \\ 0 & 0 & 0 & 1 & t'_{(n-p)} \end{pmatrix} \tag{2.57}$$

This form of T_{vu} immediately guarantees that the constraint (2.55) is satisfied.

The algorithm will be presented using the following notation for F_v and K_v in the transformed form of (2.47):

$$F_v = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 & 0 \\ f'_2 & \lambda_2 & 0 & 0 & 0 \\ f'_3 & & \lambda_3 & 0 & 0 \\ . & . & . & . & . \\ & & f'_{(n-p)} & & \lambda_{(n-p)} \end{pmatrix} \; ; \; K_v = \begin{pmatrix} k'_1 \\ . \\ . \\ . \\ k'_{(n-p)} \end{pmatrix} \tag{2.58}$$

For simplicity, all matrices are assumed to be real and the Schur form of F is strictly triangular. With the notation (2.57) and (2.58), the first row of (2.51) becomes:

$$(1 \;\; t'_1) A_u - \lambda_1 (1 \;\; t'_1) = k'_1 C_u \tag{2.59}$$

and analogously for the i-th row of (2.51):

$$(0 \ldots 1 \; t'_i) A_u - (f'_i \; \lambda_i \; 0 \ldots 0) T_{vu} = k'_i C_u \tag{2.60}$$

These equations can now be written as:

$$(t'_1 \;\; k'_1) \begin{pmatrix} A_{\lambda_1} \\ \hline -C_u \end{pmatrix} = -a'_{\lambda_1} \tag{2.61}$$

and

$$(f'_i \;\; t'_i \;\; k'_i) \begin{pmatrix} -T_i \\ \hline A_{\lambda_i} \\ \hline -C_u \end{pmatrix} = -a'_{\lambda_i} \tag{2.62}$$

where A_{λ_i} and a_{λ_i} denote respectively the bottom n-i rows and the i-th row of the matrix $(A_u - \lambda_i I)$, and T_i denotes the top i-1 rows of T_{vu}.

Notice that (2.61) and (2.62) denote a system of equations of the type:

$$x'_i M_i (x_1 \ldots x_{i-1}) = a'_{\lambda_i} \tag{2.63}$$

in the unknown $x'_i = (f'_i \; t'_i \; k'_i)$ whereby M_i depends on the previous vectors x'_k for $k < i$. One can thus determine the matrices F_v, T_{vu} and K_v row by row for $i = 1, ..., (n-p)$ provided that all M_i matrices are *left invertible.* Due to the use of the staircase observer Hessenberg form, it can easily be shown that this condition is always fulfilled [116]. The *underdetermined* set of equations (2.63) can then be solved for x_i under the constraint that the norm of x_i should be *minimal.* This is a logical choice since it minimizes the norm of the matrix K_v and the off diagonal part of F_v and T_{vu}. These minimal norms have a positive effect on the condition number for the eigenvalue problem of F, defined at the end of section 2.4.2.1, and on the condition number of the inversion of the compound matrix in (2.56). Two properties that are welcome in solving the problem numerically. In order to construct the "minimum norm" solution of (2.61) and (2.62), one performs the QR decomposition of the matrices M_i. Using Householder transformations, this requires pn^2 operations per M_i matrix, because M_i has only (p-1) non-zero subdiagonals *due to* the use of the observer Hessenberg form. For the solution of the triangular system R_i and the back transformation with Q_i one approximately needs $\frac{1}{2}n^2$ and $2pn$ operations, respectively per vector x_i. Since there are (n-p) such decompositions this amounts (roughly) to:

$$(n-p)pn^2 \quad \text{flops for solving } F_v, T_{vu} \text{ and } K_v \tag{2.64}$$

Finally, for constructing G, H and P we use:

$$P = T_u U^* B \;\text{ and }\; (G \mid H) = U \begin{pmatrix} T_u \\ \hline C_u \end{pmatrix}^{-1} \tag{2.65}$$

with U given in (2.51) these computations require:

$$n^2 m + (n-p) n m \;\text{ and }\; \frac{n^3}{2} \text{ flops} \tag{2.66}$$

respectively, exploiting the triangular form of $\begin{pmatrix} T_u \\ \hline C_u \end{pmatrix}$

Remark 2.6: The presented algorithm can be applied in a very similar way for the case the eigenvalues of the observer system matrix F are *complex*. This is explicitly treated in [116].

Example 2.2: The presented algorithm for the reduced order observer problem is applied to reconstruct the flight-path, using the system model discussed in the previous chapter. This model can easily be put *at no cost* in observer Hessenberg form (2.13) and this *without loss of accuracy*. In this example we use the model (1.18) and (1.19) for a reference flight condition $V_T = 55.8\ \frac{m}{s}$ and $\gamma_T = 0.021\ rad$. The reduced order observer has dimensions (n-p) = 2. The results of the reduced order observer algorithm are computed for the spectrum of F in (2.47) chosen as $\{0.9, 0.95\}$. For an arbitrary registration of the input sequence u_k and resulting output sequence y_k over a time interval of 20 sec, the observed and "real" missing state variables, i.e. the angle of attack α and the angle of pitch θ, are pictured in fig. 2.4. This figure clearly shows the difference in speed by which convergence occurs for both state variables. This coincides with the difference specified by the eigenvalues of F. Therefore, the speed of the observer could easily be increased by choosing the spectrum of F much more inside the unit circle. Choosing the spectrum of F to be $\{0.5, 0.6\}$ the observed state variables are plotted in fig. 2.5. From figures 2.4 and 2.5 we can clearly observe that the behaviour of the estimates in both cases is the same. After a short transition period, depending on the specified dynamics of the observer, the observed state variables coincide "nearly" with the corresponding real state variables. The remaining difference after this transition period is due to the *discretization error* and will be discussed in more detail in section 4.3.3. However, fig. 2.6 shows the effect when the input signal u_k and output signal y_k are noisy for the same choice of the spectrum as in fig. 2.5. We clearly see that this has disastrous effect on the observed signals although the noise statistics of the real flight test equipment [41] was used, and this has proved to be very accurate. This may easily be observed from the covariance data of the input and output measurements given by (5.12).

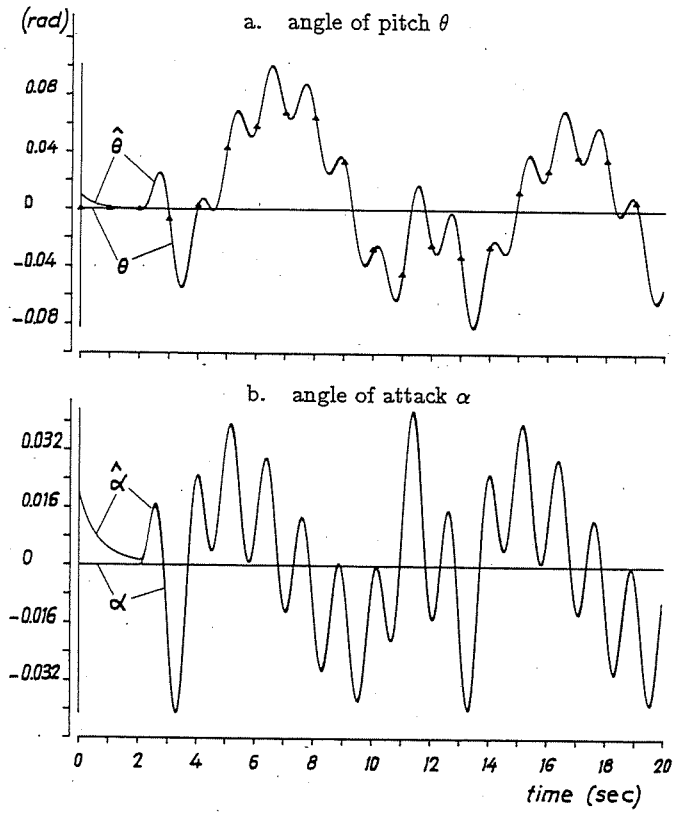


Figure 2.4: The reconstruction of the missing flight-path quantities α and θ with a reduced order observer with system dynamics $\{0.9, 0.95\}$ and using noise free measurements.

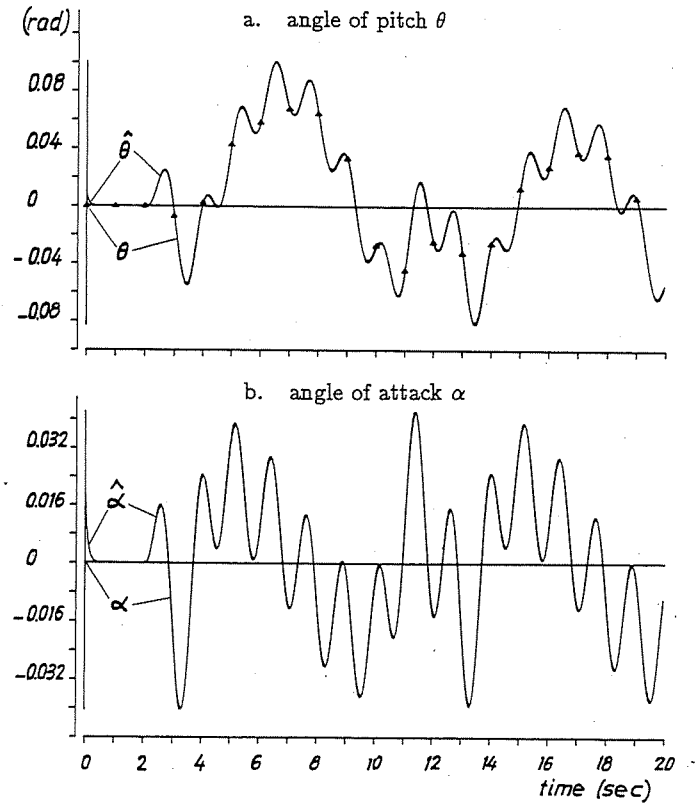


Figure 2.5: Similar experimental conditions as presented in figure 2.4 but with observer dynamics $\{0.5, 0.6\}$.

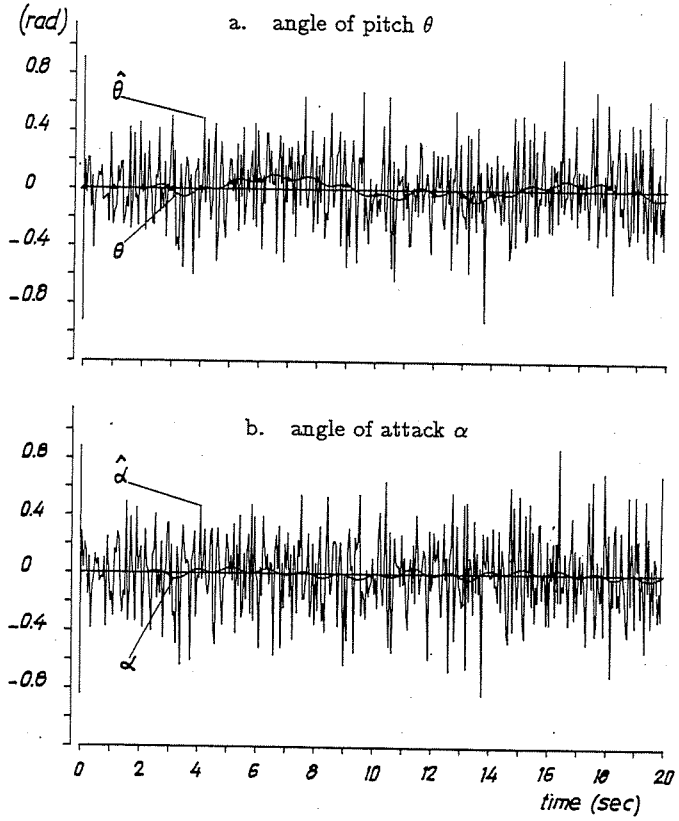


Figure 2.6: Similar experimental conditions as presented in figure 2.5 but using noisy, but still accurate, measurements.

2.5. Generalized state space descriptions

The linear state space model (2.4) results in many practical applications from linearizing a set of non-linear differential equations in some working point (or reference condition). This is considered in more detail in Appendix A for the aircraft modelling problem discussed in the previous chapter. However, as pointed out by Rosenbrock [87], this linearization can initially give rise to the following dynamic system representation:

$$\lambda E x = Ax + Bu \tag{2.67}$$

From (2.67), (2.4) can be derived if E is *non-singular*. On the other hand, formulations like (2.67) can result immediately by writing down the physical relationships. This is e.g. the case in the analysis of switched capacitors [120]. This generalization of the state space description is often indicated as the class of *descriptor systems* or *generalized state space systems* [117], [118]. They also include cases where E is *singular*.

A *new interesting research* area in the state space approach is the direct solution of system theoretical problems for the class of descriptor systems. For cases where the E matrix is singular, this is not at all a straightforward job. Here also the use of (generalized) condensed forms will prove to be a very powerful and useful concept in designing algorithms.

2.5.1. Condensed generalized state space models

For the quintuple of matrices $\{A, B, C, D, E\}$ that determines a descriptor system, the following transformations can be used to obtain a condensed form:

$$A_t = \mathcal{T} A \mathcal{U}^* \; ; \; B_t = \mathcal{T} B \mathcal{V}^* \; ; \; C_t = \mathcal{W} C \mathcal{U}^*$$

$$D_t = \mathcal{W} D \mathcal{V}^* \; ; \; E_t = \mathcal{T} E \mathcal{U}^* \tag{2.68}$$

Under these *unitary* transformations $\mathcal{T}, \mathcal{U}, \mathcal{V}, \mathcal{W}$ one obtains condensed forms similar to those discussed in section 2.3, namely:

- the *generalized Hessenberg form*, where A_t is upper Hessenberg and E_t is upper triangular
- the *generalized Schur form*, where A_t is quasi-upper triangular and E_t is upper triangular

- the *generalized controller-Hessenberg form*, where $(B_t \mid A_t)$ is upper trapezoidal and E_t is upper triangular
- the *generalized observer-Hessenberg form*, where $\begin{pmatrix} A_t \\ C_t \end{pmatrix}$ is upper trapezoidal and E_t is upper triangular

These forms are thus identical to those presented in section 2.3, see definitions (2.2-2.5), except for E_t which is upper triangular in each case, and was the identity matrix before. Variants of these forms and number of operations to obtain them are comparable to what was stated in section 2.3. Here, we will present a new algorithm to obtain *the generalized controller-Hessenberg* form. This only requires the use of the transformations T and U in (2.68).

Algorithm 2.13:

The *first part* of the algorithm simply transforms the set of equations (2.67) to another set:

$$\lambda \mathcal{T} E x = \mathcal{T} A x + \mathcal{T} B u \tag{2.69}$$

where the transformation matrix $\mathcal{T}$ is chosen so that it triangularizes E. This can be done orthogonally, using the QR factorization techniques discussed in section 2.2.

In the *second part*, the compound matrix $(\mathcal{T}B \mid \mathcal{T}A)$ is transformed to the controller Hessenberg form (2.12) while *preserving* the triangular form of $\mathcal{T}E$. This can similarly be done in (n-1) steps operating on the sequence of (compound sub-) matrices as described in algorithm 2.5. Therefore, we will only describe the first step in this second part.
Initially, the compound matrix $(\mathcal{T}B \mid \mathcal{T}A)$ and the matrix $\mathcal{T}E$ have the following structure, illustrated for $n = 7, m = 3$:

$$(\mathcal{T}B \mid \mathcal{T}A) = \left(\begin{array}{ccc|ccccccc} \mathbf{x} & x & x & x & x & x & x & x & x & x \\ x_6 & x & x & x & x & x & x & x & x & x \\ x_5 & x & x & x & x & x & x & x & x & x \\ x_4 & x & x & x & x & x & x & x & x & x \\ x_3 & x & x & x & x & x & x & x & x & x \\ x_2 & x & x & x & x & x & x & x & x & x \\ x_1 & x & x & x & x & x & x & x & x & x \end{array}\right)$$

$$\mathcal{T}E = \begin{pmatrix} x & x & x & x & x & x & x \\ 0_6 & x & x & x & x & x & x \\ 0 & 0_5 & x & x & x & x & x \\ 0 & 0 & 0_4 & x & x & x & x \\ 0 & 0 & 0 & 0_3 & x & x & x \\ 0 & 0 & 0 & 0 & 0_2 & x & x \\ 0 & 0 & 0 & 0 & 0 & 0_1 & x \end{pmatrix} \tag{2.70}$$

The annihilation of the elements in the (n-1) bottom rows in the first column of $(\mathcal{T}B \mid \mathcal{T}A)$ in (2.70) is performed by a sequence of Givens transformations, applied in the following way:

- *First*, element x_1 in (2.70) is annihilated with a Givens transformation $G_{67}(\theta)$ applied from the left on the matrices $(\mathcal{T}B \mid \mathcal{T}A)$ and $\mathcal{T}E$. This introduces a non-zero element in $\mathcal{T}E$ of (2.70), which replaces element 0_1. To annihilate this element, again a Givens transformation $G_{67}(\theta^*)$ is applied, but now from the right on the matrices $\mathcal{T}E$ and $(\mathcal{T}B \mid \mathcal{T}A)$. Therefore, this does not affect the introduced zero in x_1.
- We can continue this procedure to annihilate all elements x_i and their corresponding introduced non-zero values 0_i for $i = 1, \ldots, n-2$ in (2.70).
- At *last*, element x_6 in (2.70) is annihilated by a Givens transformation $G_{12}(\theta)$ applied similarly as G_{67}. To annihilate the introduced non-zero element 0_6, again a Givens transformation $G_{12}(\theta^*)$ is applied from the right. This transformation operates on the m+1 and m+2 columns of the so far obtained form of $(\mathcal{T}B \mid \mathcal{T}A)$. Therefore, we observe that all transformations applied from the right do not affect the first m columns of the considered compound matrix $(\mathcal{T}B \mid \mathcal{T}A)$ as in algorithm 2.5.

Operating in this way guarantees that this last fact will remain true for the subsequent (compound sub-) matrices of the transformed form of $(\mathcal{T}B \mid \mathcal{T}A)$ as considered in algorithm 2.5. Therefore, after (n-1) steps the controller Hessenberg form of the pair (B_t, A_t) as given in (2.12) will result.

A *third part*, can compute the staircase form of (B_t, A_t) with the same structure as in (2.12) by the algorithm developed by Miminis and Paige [73]. Hereto, the orthogonal transformations $\mathcal{U}$ (applied as described in the second part) and $\mathcal{V}$ defined in (2.68) are required.

The use of these forms may lead to straightforward generalizations of ex-

isting algorithms, developed for the classical state space formulation (2.4). This is demonstrated in the next subsection for a key problem in this thesis, namely observing the state of a linear system. A *new generalization* is formulated for the reduced order observer.

2.5.2. The generalized reduced order observer

Under the assumption that the system (2.67) is *observable*, see definition 2.6, one can construct a reduced order observer of the form:

$$\lambda S z = F z + K y + P u \tag{2.71}$$

Definition 2.6: A descriptor system of the form (2.67) is called *observable* if and only if the pencil

$$\begin{pmatrix} \lambda E - A \\ C \end{pmatrix}$$

has full column rank for all values of λ, including $\lambda = \infty$ [118].

Remark 2.7: A sufficient condition that the pencil in definition 2.6 has full column rank at ∞ is that the compound matrix $\begin{pmatrix} E \\ C \end{pmatrix}$ has full column rank. This is sometimes used as an alternative definition of observability at ∞ of descriptor systems, see also [118]. Furthermore, this full rank condition is preserved under the unitary transformations $\mathcal{T}, \mathcal{U}, \mathcal{V}$ and $\mathcal{W}$ defined in (2.68).

The observability condition allows to construct the generalized observer-Hessenberg form, displayed for $n = 6, p = 2$ and $m = 3$:

$$(E_t)_{OH} = \left(\begin{array}{cccccc} X & x & x & x & x & x \\ 0 & X & x & x & x & x \\ 0 & 0 & X & x & x & x \\ 0 & 0 & 0 & X & x & x \\ \hline 0 & 0 & 0 & 0 & \overline{x} & \overline{x} \\ 0 & 0 & 0 & 0 & 0 & \overline{x} \end{array}\right) = \left(\begin{array}{c} E_t^* \\ \hline X \end{array}\right)$$

$$\left(\frac{B_t \mid A_t}{D \mid C_t}\right)_{OH} = \left(\begin{array}{ccc|ccccc} x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x \\ x & x & x & 0 & x & x & x & x \\ x & x & x & 0 & 0 & x & x & x \\ x & x & x & 0 & 0 & 0 & x & x \\ \hline x & x & x & 0 & 0 & 0 & 0 & X \\ x & x & x & 0 & 0 & 0 & 0 & X \end{array}\right) \qquad (2.72)$$

Based on remark 2.7 the compound matrix $\left(\frac{E_t^*}{C_t}\right)$ has full rank, implying that the elements denoted by X in (2.72) are non-zero. For this condition it will be proved in theorem 2.1 that the state dimension of the observer (2.71) is again (n-p).

The state z of the observer (2.71) then supplies an extra set of linear combinations of x, namely $z \rightarrow Tx$, so that the observed state can be calculated from:

$$\begin{pmatrix} T \\ C \end{pmatrix} \hat{x} = \begin{pmatrix} z \\ y \end{pmatrix} \qquad (2.73)$$

The solution of this set of equations converges asymptotically to the real state x if S, F, K, P and T are appropriately chosen. A possible procedure is described in the following theorem. Here we use the definition of generalized eigenvalues.

Definition 2.7: The *generalized eigenvalues* of a descriptor system given by (2.67) are the values of $\lambda = \frac{\nu}{\mu}$ for which $(\mu A - \nu E)$ becomes singular.

Theorem 2.1: Let T be an $(n-p) \times p$ matrix, such that

$$T = XE \quad ; \quad \begin{pmatrix} T \\ C \end{pmatrix} \text{ is invertible} \qquad (2.74)$$

with X satisfying the following generalized Sylvester equation:

$$SXA - FXE = KC \qquad (2.75)$$

Then putting,

$$P = SXB \qquad (2.76)$$

we have that,

$$\lambda S(Tx - z) = F(Tx - z) \qquad (2.77)$$

A solution X of (2.75) does exist for a specific choice of the generalized eigenvalues of the observer (2.71). Convergence of the reduced order observer is thus obtained when the observer is chosen to be stable. This implies that the generalized eigenvalues, defined by definition 2.7, of the observer have to be *finite* (to lie inside the unit circle for the discrete time case and in the open left half plane for the continuous time case). Equivalently, the matrix S in (2.71) must be *invertible*.

Proof:

The proof of theorem 2.1 will be given for the system (2.67) in generalized staircase observer Hessenberg form, which may be derived in a complete similar manner as presented in algorithm 2.13. Again, using appropriate (unitary) coordinate transformations of (2.71) we can assume S and F to be *lower triangular*. In this form the generalized eigenvalues of the observer are easily specified by the diagonal elements of S and F. These are denoted by $\{\mu_i\}$ and $\{\nu_i\}$ respectively. Furthermore, if we use a similar notation for the X, S, F and K matrices as before:

$$X = \begin{pmatrix} 1 & & x_1' & & \\ 0 & 1 & & x_2' & \\ 0 & 0 & 1 & & x_3' \\ . & . & . & . & . \\ 0 & 0 & 0 & 1 & x_r' \end{pmatrix} \; ; \; S = \begin{pmatrix} \mu_1 & 0 & 0 & 0 & 0 \\ s_2' & \mu_2 & 0 & 0 & 0 \\ s_3' & & \mu_3 & 0 & 0 \\ . & . & . & . & . \\ & & s_r' & & \mu_r \end{pmatrix}$$

$$F = \begin{pmatrix} \nu_1 & 0 & 0 & 0 & 0 \\ f_2' & \nu_2 & 0 & 0 & 0 \\ f_3' & & \nu_3 & 0 & 0 \\ . & . & . & . & . \\ & & f_r' & & \nu_r \end{pmatrix} \; ; \; K = \begin{pmatrix} k_1' \\ . \\ . \\ . \\ k_r' \end{pmatrix} \qquad (2.78)$$

then the first row of (2.75) can be written as:

$$(x_1' \mid k_1') \left(\frac{A_{\mu\nu_1}}{-C}\right) = -a'_{\mu\nu_1} \qquad (2.79)$$

and the i-th row becomes:

$$(s_i' \mid f_i' \mid x_i' \mid k_i') \left(\begin{array}{c} X_i A \\ \hline -X_i E \\ \hline A_{\mu\nu_i} \\ \hline -C \end{array}\right) = -a'_{\mu\nu_i} \qquad (2.80)$$

where $A_{\mu\nu_i}$ and $a_{\mu\nu_i}$ denote respectively the bottom (n-i) rows and the i-th row of the matrix $(\mu_i A - \nu_i E)$, and X_i denotes the top (i-1) rows of X. These two equations (2.79) and (2.80) can be written in the form:

$$X_i' M_i(X_1, \ldots, X_{(i-1)}) = a'_{\mu\nu_i} \tag{2.81}$$

Here also the choice of the *staircase* observer Hessenberg form of the system (2.67) assures that the M_i matrices in (2.81) have full column rank. And again there is no requirement that the generalized eigenvalues of the observer (2.71) have to differ from those of the original system (2.67). The above condition assures that the M_i matrices are *left invertible*. This allows us to solve (2.81) again in *"minimum norm" sense*. By construction the solution X of dimension $(n-p) \times n$ in (2.78) has full column rank. Now, if $\left(\frac{E_t^*}{C_t}\right)$ has full rank, then it is assured that the compound matrix $\left(\frac{T}{C}\right)$ is invertible, which proves the theorem.

■

The proof of theorem 2.1 at the same time contains an algorithm to construct a generalized observer of the form (2.71).

Remark 2.8: Again a similar procedure can be used if the generalized eigenvalues of the observer (2.71) are *complex*.

2.6. Concluding remarks

It has been shown in this chapter that "condensed" forms are a very useful and powerful tool in the development of efficient and reliable solutions to many different problems in system theory based on a state space model. These forms either lead to novel algorithms (as shown in sections 2.4.2.1 and 2.4.3) or allow to speed up existing algorithms considerably (as demonstrated in sections 2.4.2.2- 2.4.2.4). The latter is especially experienced in algorithms of iterative (or recursive) type, where similar matrix operations have to be performed many times. This is also required in the recursive solution of the *Riccati equation*, which e.g. characterizes the Kalman filter. The discussion of *new and more efficient* implementations of this filter is postponed to chapter 4.

In the presentation of the different algorithms we emphasized the use of unitary (orthogonal) transformations because of their great importance in preserving the sensitivity of the considered problems and in allowing to

derive numerically stable algorithms. Furthermore, it is demonstrated that the use of (generalized) condensed forms can extend the ideas presented in this chapter to other (generalized) problems. On the other hand, the experiments with the flight-path reconstruction problem demonstrated that the Luenberger approach is *not useful* for practical application.

Chapter 3.

A new approach towards Kalman Filtering

3.1. Introduction

In the previous chapter we presented the so-called *identity observer* as a special case of the full-order observer. This implementation, given in figure 3.1., was completely determined by the gain matrix K of the innovation signal, i.e. the difference between the measured output y and the reconstructed output $C\hat{x}$. Techniques discussed in section 2.4.2.1 enable to determine such a gain matrix for a specific choice of the filter dynamics. For the case were the observed signals are *noise free*, it has been shown that the observer can be made arbitrarily fast by moving its eigenvalues far into the unit circle. This, however, makes the filter very sensitive when the observations *contain measurement noise*. Experiments performed at the end of in section 2.4.3 even showed that the observed or reconstructed state variables became *very inaccurate* under the presence of measurement errors. A *trade-off* between speed and accuracy of the observer is given by the so-called **Kalman filter**. Here the gain matrix K is tuned "optimally" taking into account the magnitude (the statistics) of the measurement errors. This optimality is specified as the *minimum variance estimate* of the reconstructed variables.

This "optimal" filter is often derived, as well in the continuous time case as in the discrete time case, in the framework of statistical analysis. This was also done by Kalman in his original paper of 1960 [51]. It is espe-

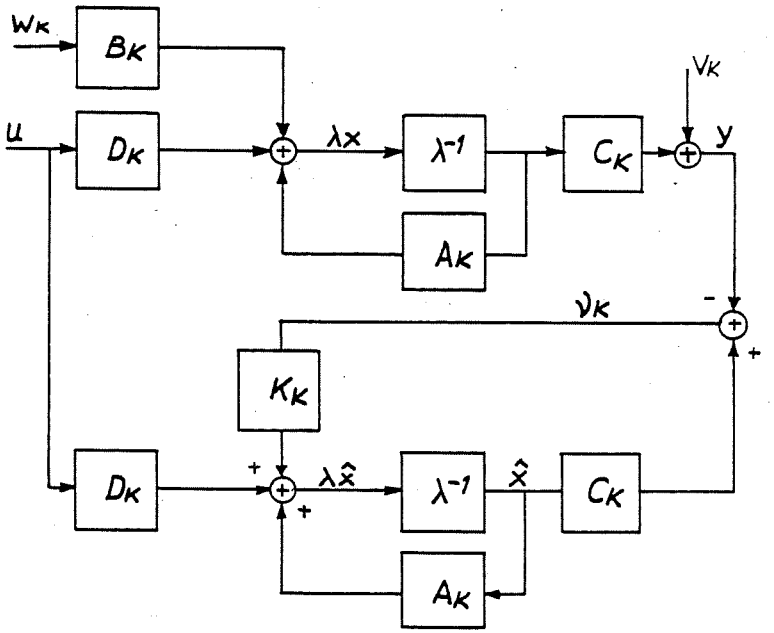


Figure 3.1: A block scheme representation of the Kalman filter

cially based on the *conditional probability theory*. One of the assumptions used in this derivation is that the measurement errors have a *Gaussian distribution*. However, practical use of the Kalman filter has shown that this is a rather *restrictive* assumption. This is because the filter remains *the most optimal* linear *filter* also in the case of non-Gaussian signals (see e.g. [1]). Furthermore, the theoretical foundations used from statistics are quite involved and complex. Extensive use of the Kalman filter has demonstrated that the framework of statistical analysis is only one out of a set of different ways to derive the "optimal" filter structure. Many *alternative* theoretical derivations of the filter have been presented in literature. This resulted certainly in an increased use of the filter in a broad spectrum of applications and led to new insights and new applications better suited to

a certain class of problems (or better understandable for a certain class of scientists). An overview of these different theoretical approaches, would be very interesting, but would lead us to far. To name only a few, there is e.g. the innovation approach [48], scattering theory [28] and the work of Duncan and Horn [20]. In [20] a relation is formulated between the minimum variance estimation problem for general time variant discrete systems and *weighted least squares*. This *simple* and *well known* framework of regression analysis will certainly contribute to new developments in the area. Recently, C.C. Paige [81], used this approach in combination with a generalization of the covariance representation to derive a new filter structure. This generalization of the covariance representation of stochastic variables is based on a numerical motivation. Section 3.4 will present this generalization and its usefulness. The new filter structure will be presented in section 3.5. Section 3.6 more deeply analyzes this filter. Here we show that the filter represents a whole class of new square root filters, of which the existing square root covariance filter and square root information filter are extremes. We start this chapter with the formulation of the "optimal filtering" problem and the notation used herein.

3.2. Notation and preliminaries

In this section we introduce our notation and list the statistical assumptions made about the model and observation process. Consider the discrete time-varying linear system,

$$x_{k+1} = A_k x_k + B_k w_k + D_k u_k \tag{3.1}$$

and the linear observation process,

$$y_k = C_k x_k + v_k \tag{3.2}$$

where,

- x_k is the state vector to be estimated (dimension n)
- u_k is the deterministic input vector (dimension r)
- y_k is the measurement vector (dimension p)
- w_k is the process noise (dimension m)
- v_k is the measurement noise (dimension p)

- A_k is the (non-singular) transition matrix
- B_k is the input noise distribution matrix
- C_k is the observation matrix
- D_k is the input signal matrix

The process noise and measurement noise sequences are assumed zero mean and uncorrelated :

$$E\{w_k\} = 0\ ,\ E\{v_k\} = 0\ ,\ E\{w_k v_j'\} = 0 \tag{3.3}$$

with covariances :

$$E\{w_j w_k'\} = Q_k \delta_{jk}\ ,\ E\{v_j v_k'\} = R_k \delta_{jk} \tag{3.4}$$

where $E\{.\}$ denotes the mathematical expectation and δ_{ij} is the kronecker delta, which is 1 for $j = i$ and 0 otherwise. Suppose further that the initial state x_0 is a random variable with mean $\overline{x}_0$:

$$E\{x_0\} = \overline{x}_0 \tag{3.5}$$

and covariance:

$$E\{(x_0 - \overline{x}_0)(x_0 - \overline{x}_0)'\} = P_0 \tag{3.6}$$

and independent of $\{w_k\}$ and $\{v_k\}$.

3.3. The discrete Kalman filter problem

Let the discrete time system (3.1) and (3.2) be given and the system matrices $\{A_k, B_k, C_k, D_k\}$ and covariance matrices $\{P_0, Q_k, R_k\}$ be known, then the problem is to compute the *minimum variance estimate* of the stochastic variable x_k, provided y_1 up to y_j have been measured:

$$\hat{x}_{k|j} = \hat{x}_{k|y_1, \ldots, y_j} \tag{3.7}$$

and *associated error covariance matrix*:

$$P_{k|j} = E\{(x_k - \hat{x}_{k|j})(x_k - \hat{x}_{k|j})'\} \tag{3.8}$$

From this problem formulation three alternative types of estimation problems can be defined.

Definition 3.1: The estimate $\hat{x}_{k|j}$ in (3.7) is called the *filtered estimate* when it is based on the measurements $y_1, \ldots, y_j$ for $j = k$.

Definition 3.2: The *smoothed estimate* differs form the filtered estimate in that the estimate $\hat{x}_{k|j}$ needs not to become available at time k, and measurements received later than time k, i.e. for $j > k$, can be used in obtaining the estimate $\hat{x}_{k|j}$.

Definition 3.3: For the *predicted estimate* the aim is to obtain at time j information about $\hat{x}_{k|j}$ for $k > j$.

In real-time applications, the problem is generally restricted to the determination of the filtered estimate $\hat{x}_{k|k}$ and the one-step predicted estimate $\hat{x}_{k|k-1}$, a special case of the definition 3.3. These quantities have to be determined in a *recursive way.*

The above cited problem can be formulated mathematically as:

$$\min_{\hat{x}_{k|j}} E\{\|x_k - \hat{x}_{k|j}\|^2 \mid y_1, \ldots, y_j\} \tag{3.9}$$

Definition 3.4: The trace of matrix M, denoted by $tr(M)$, is the sum of all the diagonal elements of M.

Using this definition, we can reformulate criterion (3.9) analogously as,

$$\min_{\hat{x}_{k|j}} tr(E\{(x_k - \hat{x}_{k|j})(x_k - \hat{x}_{k|j})'\}) \tag{3.10}$$

Where we used the property of the tr-operator in definition 3.4 that the $tr(AB) = tr(BA)$.

Formulating criterion (3.9) as a classical least squares problem, namely $\min_{\hat{x}_{k|j}} \|x_k - \hat{x}_{k|j}\|^2$ subject to (3.1) and (3.2) has no sense, since x_k is an unknown random variable. Therefore, the conditional expectation operator $E\{ \cdot \mid y_1, \ldots, y_j\}$ is used. This explicitly expresses the conditioning of the estimate on the measurements $y_1, \ldots, y_j$. The choice to measure the error associated with an estimate as $\|x - \hat{x}\|^2$, rather than say $\|x - \hat{x}\|$, $\|x - \hat{x}\|^4$ or $\cosh \|x - \hat{x}\|^3$ is somewhat arbitrary. However, as it will turn out, this particular measure gives rise to a *linear structure* of the estimator. Before going into details of the solution of the Kalman filter problem, we first sketch the solution using conditional probability theory. Here, it is assumed that the considered stochastic signals in the problem are *Gaussian.* In this case their mean and covariance uniquely determine their probability density function. Since the Gaussian property is preserved by

linear operation, the conditional probability density function of the state variable x conditioned on the measurements y, denoted as $p_{x|y}\{x \mid y\}$, is also gaussian. An explicit form for $p_{x|y}\{x \mid y\}$ can then be derived using the assumption that the different stochastic variables considered in the problem are independent and using the rules of the conditional probability density theory. The conditional mean of $p_{x|y}\{x \mid y\}$ and the associated error covariance matrix determine the solution to the Kalman filter problem. This approach is followed and described very clearly in [1]. However, as is the view of the writer of this thesis, this goes along with a number of lengthy derivations. These derivations are first based on statistical analysis, which requires an extensive exposure on itself, and secondly do not contribute to the insight of the problem. This is mainly because this mathematical framework is only used as a *tool* to derive a solution to the Kalman filter problem. Therefore, I think, the latter should be made as simple as possible. Furthermore, the restrictions to gaussian stochastic signals in order to come to a solution in the statistical framework are not necessary. It is known (see e.g. [1]) that the derived filter structure remains the *best*, in minimum variance sense, among the class of linear filters when the stochastic signals are *not Gaussian.* Another disadvantage of this approach is that they lead to a recursive formulation of $P_{k|k}$ and/or $P_{k|k-1}$ and the estimates $\hat{x}_{k|k}$ and/or $\hat{x}_{k|k-1}$ for $k = 1, 2, \ldots$. Other recursive implementations, such as the information filter, the square root filters discussed later on in this chapter are not at all *a straightforward* consequence of the previous relationships. In the following sections, a method based on the simple least squares method that overcomes a large number of disadvantages of the previous approach will be presented. This presentation will start with a generalization of the covariance data representation and is based on the work of C.C. Paige [81].

3.4. A general covariance representation

The information assumed in the problem formulation of the previous section about the stochastic variables was their mean and covariance matrix. Combine (3.5) and (3.6), this can be denoted shortly as follows:

$$x_0 \frown (\overline{x}_0, P_0) \tag{3.11}$$

Generally, it is further assumed that the covariance matrices $\{P_0, Q_k, R_k\}$ are *positive definite.* Practical applications where this condition is violated require special care. For example, the singularity of Q_k requires the linear

dependent components in w_k to be removed first. Therefore, this does not restrict the generality of the latter assumption. On the other hand, the regularity of R_k rules out the possibility of including perfect measurements not corrupted by noise. The particular case of perfect measurements leads to special adaptations of the different Kalman filter implementations, e.g. the use of the Moore-Penrose inverse, see e.g. [5]. A similar problem may occur for P_0, when some of the initial conditions are exactly known. On the opposite, it might be possible that no information is available about the initial conditions. For this case, the norm of the covariance matrix will be infinity. On can then use the inverse of the covariance matrix in an information filter implementation of the Kalman filter. The treatment of both cases, exactly known and completely unknown initial conditions, in a combined way is not possible with an existing Kalman filter implementation.

To handle the above cited problems with covariance data in a unique way, a modified covariance representation of the stochastic variables will be given. First, consider the case where P_0 in (3.11) is regular. This allows to factor P_0 as follows:

$$P_0 = S_0 S_0' \tag{3.12}$$

where the Choleski factor S_0 may be chosen *upper* or *lower* triangular. This freedom of choice will be exploited, in chapter 4, to develop fast Kalman filter implementations. Using factorization (3.12), (3.11) can be expressed by the following equation:

$$x_0 = \overline{x}_0 + S_0 \tilde{u}_0 \quad \text{with} \quad \tilde{u}_0 \frown (0, I) \tag{3.13}$$

Analogously, for the regular case we may consider the factorization of the inverse covariance matrix, as:

$$P_0^{-1} = U_0' U_0 \tag{3.14}$$

to represent (3.11) by,

$$U_0 x_0 = \overline{b}_0 + \tilde{u}_0 \quad \text{with} \quad \overline{b}_0 = U_0 \overline{x}_0 \quad \text{and} \quad \tilde{u}_0 \frown (0, I) \tag{3.15}$$

Equations (3.13) and (3.15) are algebraically equivalent *when* S_0 and U_0 are square and non-singular. But even then they will probably lead to numerically different results. For example when U_0 has norm 1 but is ill-conditioned with respect to inversion, then $S_0 = U_0^{-1}$ will have a very large norm. Representation (3.13) allows linear combinations of the elements of

x_0 to be constant (known *a priori*) having an S_0 which does not have full row rank. On the other hand, (3.15) allows for the possibility that U_0 has less than column rank. In this case no information is available of that part of x in the null-space of U_0, denoted by $\mathcal{N}(U_0)$ (where the null-space of a matrix, also called its kernel, is defined in (1.17)).

The obvious generalization is to replace both (3.13) and (3.15) by,

$$U_0 x_0 = \overline{b}_0 + S_0 \tilde{u}_0 \quad \text{with} \quad \tilde{u}_0 \frown (0, I) \tag{3.16}$$

where ideally the norms of S_0 and U_0 are of reasonable size. Representations (3.13) and (3.15) are a special case of (3.16). For $U_0 = I$ we have (3.13) and for $S_0 = I$ we have (3.15). However, (3.16) covers cases which neither (3.13) nor (3.15) can handle. This is demonstrated by the following example.

Example 3.1: If we take U_0 and S_0 in (3.16) as,

$$U_0 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \; ; \; S_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

then the sum of the first two elements of x_0 are considered to be *random*, their difference to be *constant* and there is *no information* about the third element.

The same freedom can be incorporated in the presentation of the covariance data of the process noise w_k and the measurement noise v_k. If we furthermore include the possibility of correlation between w_k and v_k, this can be combined as follows:

$$-\begin{pmatrix} R_k^\ell & 0 \\ N_k^\ell & Q_k^\ell \end{pmatrix} \begin{pmatrix} v_k \\ w_k \end{pmatrix} = \begin{pmatrix} R_k^r & 0 \\ N_k^r & Q_k^r \end{pmatrix} \begin{pmatrix} \tilde{v}_k \\ \tilde{w}_k \end{pmatrix}$$

$$\text{with} \quad \begin{pmatrix} \tilde{v}_k \\ \tilde{w}_k \end{pmatrix} \frown (0, I) \tag{3.17}$$

where the supperscripts ℓ and r stand for left and right factor respectively. Of course, this would introduce far more flexibility than is usually needed, but it is included here to demonstrate its general applicability.

3.5. A general, recursive algorithm for the Kalman filter problem

The subsequent formulation of the Kalman filter problem, along with the recursive solution to this problem are based on the recent work of C.C. Paige [81].

3.5.1. The Kalman filter problem as a weighted least squares problem

The main advantage of the given general covariance representation in the previous section is that now the different equations occurring in the Kalman filter problem can be formulated in a *unified manner.*
Therefore, we briefly recall those equations. First, for the initial conditions, the process and measurement noise we have (3.16) and (3.17) written as:

$$\hat{b}_0 = \hat{U}_0 x_0 + \hat{S}_0 \tilde{u}_0 \tag{3.18}$$

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} R_k^\ell & 0 \\ N_k^\ell & Q_k^\ell \end{pmatrix} \begin{pmatrix} v_k \\ w_k \end{pmatrix} + \begin{pmatrix} R_k^r & 0 \\ N_k^r & Q_k^r \end{pmatrix} \begin{pmatrix} \tilde{v}_k \\ \tilde{w}_k \end{pmatrix} \tag{3.19}$$

In a similar way, we can write down the state equations (3.1):

$$-D_k u_k = A_k x_k + B_k w_k - E_k x_{k+1} \tag{3.20}$$

given for descriptor systems, defined in Chapter 2, and the output equation (3.2):

$$y_k = C_k x_k + v_k \tag{3.21}$$

The uniformity between equations (3.18-3.21) immediately becomes clear when writing them down as the following set of equations:

$$\begin{pmatrix} \hat{b}_0 \\ y_0 \\ 0 \\ -D_0 u_0 \\ 0 \\ y_1 \\ . \\ . \end{pmatrix} = \begin{pmatrix} \hat{U}_0 & & & & & & & \\ C_0 & I & & & & & & \\ 0 & R_0^\ell & & & & & & \\ A_0 & 0 & B_0 & -E_0 & & & & \\ & N_0^\ell & Q_0^\ell & 0 & & & & \\ & & & C_1 & I & . & . & \\ & & & . & . & . & . & \\ & & & . & . & . & . & \end{pmatrix} \begin{pmatrix} x_0 \\ v_0 \\ w_0 \\ x_1 \\ v_1 \\ . \\ . \end{pmatrix} +$$

$$\begin{pmatrix} \hat{S}_0 & & & & \\ 0 & 0 & & & \\ & R_0^r & & & \\ & 0 & & & \\ 0 & N_0^r & Q_0^r & & \\ & 0 & 0 & & \\ & & & R_1^r & \\ . & . & . & . & \end{pmatrix} \begin{pmatrix} \tilde{u}_0 \\ \tilde{v}_0 \\ \tilde{w}_0 \\ \tilde{v}_1 \\ . \\ . \end{pmatrix} \tag{3.22}$$

where all stochastic quantities denoted by $(\tilde{.})$ are zero mean with unit covariance and are uncorrelated. Obviously, this can be denoted as:

$$y = Fx + Lu \qquad u \frown (0, I) \tag{3.23}$$

The minimum variance estimate of x is now determined by the solution of the following weighted least squares problem:

$$\min_x u'u \quad \text{subject to} \quad y = Fx + Lu \tag{3.24}$$

This problem is clearly the classical least squares problem when $L = I$. The solution of this problem comes down to solving an overdetermined set of equations (3.23). As in the classical least squares problem, orthogonal transformations can be used to solve the more general problem (3.24) in a reliable way. A description of such a technique, for general F and L matrices is given in [79] and [57]. These methods would of course apply to (3.24), allowing to determine at the same time the smoothed variables (see e.g. [82]). However, it will be shown that the *special structure* of F and L in (3.23) allow to solve (3.24) *recursively.* Therefore, the matrices F and L need not to be stored as full matrices. On the other hand, we point out that no products of matrices appear in (3.22), but that only the original data appears as the subblocks of F and L. This is a very important consequence that assures that if a numerical stable algorithm can be designed to solve problem (3.24), at the same time the *maximal achievable accuracy* is obtained. However, the practical design of algorithms has to take into account the aspect of *efficiency.* Since this of course depends on the structure of the given problem, a number of implementations will be discussed in the following sections.

3.5.2. A recursive solution

Note first that (3.18) has the same form as (3.23). The initial estimate

$\hat{x}_{0|-1}$ for x_0 is of course the trivial solution to the subproblem:

$$\min_{x_0} \tilde{u}_0' \tilde{u}_0 \quad \text{subject to} \quad \hat{b}_0 = \hat{U}_0 x_0 + \hat{S}_0 \tilde{u}_0 \tag{3.25}$$

which for U_0 having full row rank is obtained by solving:

$$\hat{U}_0 \hat{x}_{0|-1} = \hat{b}_0 \tag{3.26}$$

The recursive solution consists of finding similar expressions for each x_k seperately. Starting from the set of equations (3.23) with structure (3.22), we observe that this is far more general than usually needed. Therefore, a numerically stable algorithm for this case would be *very inefficient* for most practical uses. As a result, we will present an algorithm for some more specific cases. This will clarify the steps that have to be executed when one wants to design a stable and efficient algorithm for its own specification.

The problem to be considered is with full general initial conditions, but with *positive definite* covariance matrices of the noise w_k and v_k. This allows to work with the *Choleski factors* of these matrices only. Let these be defined as follows:

$$Q_k = Q_k^{1/2}[Q_k^{1/2}]' \;\; ; \;\; R_k = R_k^{1/2}[R_k^{1/2}]' \tag{3.27}$$

Remark 3.1: Notice that historically $Q_k^{1/2}$ and $R_k^{1/2}$ have erroneously been called "square roots" instead of "Choleski factors". However, we will maintain the adjective "square root" because of the familiarity that they have acquired.

It is possible to construct a new set of equations similar to (3.23) for these conditions in a straightforward manner. They can also be derived from (3.22), which then illustrates the numerical properties of the approach used. The considered problem here corresponds to $R_k^\ell = I_p$, $N_k^\ell = 0$, $Q_k^\ell = I_m$ and $R_k^r = R_k^{1/2}$, $N_k^r = N_k^r$, $Q_k^r = Q_k^{1/2}$. The first step in going form the general formulation (3.22) to the considered problem is to eliminate the third block row of (3.22). Therefore, we transform the non-zero matrices in the second and third row of (3.22) as follows:

$$\begin{pmatrix} 0 & I_p \\ I_p & -I_p \end{pmatrix} \left(\begin{array}{cc|cc|c} y_0 & | & C_0 & I_p & | & 0 \\ 0 & | & 0 & I_p & | & R_0^{1/2} \end{array}\right) = \left(\begin{array}{cc|cc|c} 0 & | & 0 & I_p & | & R_0^{1/2} \\ y_0 & | & C_0 & 0 & | & -R_0^{1/2} \end{array}\right) \tag{3.28}$$

The unit matrix in the right hand side of (3.28) has only zero matrices above and below it. Therefore, the new second row in the transformed form of (3.22) does not affect the solution of (3.23) and can thus be dropped, along with the column in which this I_p lies. Secondly we can proceed analogously to eliminate the original fifth row in (3.22). For this, we transform the non-zero matrices in the original fourth and fifth row as follows:

$$\begin{pmatrix} 0 & I_m \\ I_n & -B_0 \end{pmatrix} \left(\begin{array}{cc|ccc|cc} -D_0 u_0 & | & A_0 & B_0 & -E_0 & | & 0 & 0 \\ 0 & | & 0 & I_m & 0 & | & N_0^r & Q_0^{1/2} \end{array}\right) =$$

$$\left(\begin{array}{ccc|ccc|cc} 0 & | & 0 & I_m & 0 & | & N_0^r & Q_0^{1/2} \\ -D_0 u_0 & | & A_0 & 0 & -E_0 & | & -B_0 N_0^r & -B_0 Q_0^{1/2} \end{array}\right) \tag{3.29}$$

With the same argument to that given for (3.28), the column and row containing the unit matrix I_m in the right hand side of (3.29) can be dropped. The same method can be used to eliminate all the other R_k^ℓ, N_k^ℓ and Q_k^ℓ. This gives rise to the following transformed form of (3.22):

$$\begin{pmatrix} \hat{b}_0 \\ y_0 \\ -D_0 u_0 \\ y_1 \\ -D_1 u_1 \\ \cdot \\ \cdot \end{pmatrix} = \begin{pmatrix} \hat{U}_0 & & & & \\ C_0 & & & & \\ A_0 & -E_0 & & & \\ & C_1 & & & \\ & A_1 & -E_1 & & \\ & & \cdot & & \\ & & & & \cdot \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \end{pmatrix} +$$

$$\begin{pmatrix} \hat{S}_0 & & & & & \\ & -R_0^{1/2} & & & & \\ & -B_0 N_0^r & -B_0 Q_0^{1/2} & & & \\ & & & -R_1^{1/2} & & \\ & & & -B_1 N_1^r & -B_1 Q_1^{1/2} & \\ & & & \cdot & \cdot & \\ & & & \cdot & \cdot & \end{pmatrix} \begin{pmatrix} \tilde{u}_0 \\ \tilde{v}_0 \\ \tilde{w}_0 \\ \tilde{v}_1 \\ \cdot \\ \cdot \end{pmatrix} \tag{3.30}$$

which can again be written as:

$$y = Fx + Lu \tag{3.31}$$

Remark 3.2: The transformation used to annihilate the original fifth row in (3.22), satisfies:

$$\begin{pmatrix} 0 & I_m \\ I_n & -B_0 \end{pmatrix} \begin{pmatrix} B_0 & I_n \\ I_m & 0 \end{pmatrix} = \begin{pmatrix} I_m & 0 \\ 0 & I_n \end{pmatrix} \tag{3.32}$$

and so this transformation matrix is well conditioned if $\|B_0\|$ is not too large, i.e. in the vicinity of one, no matter what the conditioning of B_0 itself is.
By this transformation, the system matrices F and L in (3.31) are no longer constructed from the original data only, but contain the products $B_j N_j^r$ and $B_j Q_j^{1/2}$. The matrices are however more compact. It becomes clear that we have here *sacrified to a certain extent* accuracy for efficiency.
The numerical error analysis performed by C.C. Paige in [78] shows that the use of well conditioned left transformations of (3.23) does *not contribute* to numerical instability.

Remark 3.3: Having made the transformation (3.29), the row containing the information on w_0 is apparently dropped, and this may raise some doubts. However, it is not really dropped, it is just not needed in solving (3.24). When this has been solved, the dropped row can then be used for computing the original w_0 in (3.22), if it is needed.

Solving (3.24) in a recursive manner consists, for each time instant, in a measurement update and a time update.

A. The measurement update

Given the structure of (3.30), in this step a left and right transformation are performed, which introduce zeros and full row or column matrices in certain subblocks of F and L of (3.31). First, the left transformation results in:

$$\left(\begin{array}{c|c} \hat{T}_0 & 0 \\ \hline 0 & I \end{array}\right) \left(\begin{array}{ccc} \hat{U}_0 & \hat{S}_0 & 0 \\ C_0 & 0 & -R_0^{1/2} \\ \hline A_0 & 0 & -B_0 N_0^r \end{array}\right) = \left(\begin{array}{ccc} 0 & \star & \star \\ \overline{U}_0 & \star & \star \\ \hline A_0 & 0 & -B_0 N_0^r \end{array}\right) \tag{3.33}$$

where $\overline{U}_0$ has *full row rank* and $\star$ denotes the resulted matrix quantities in that particular block row and column after transformation.

Secondly the right transformation results in:

$$\left(\begin{array}{c|cc} 0 & \star & \star \\ \overline{U}_0 & \star & \star \\ \hline A_0 & 0 & -B_0 N_0^r \end{array}\right) \left(\begin{array}{c|c} I & 0 \\ 0 & \hat{P}_0 \end{array}\right) = \left(\begin{array}{c|cc} 0 & \tilde{S}_0 & 0 \\ \overline{U}_0 & \tilde{R}_0 & \overline{R}_0 \\ \hline A_0 & \tilde{N}_0 & \overline{N}_0 \end{array}\right) \tag{3.34}$$

where now $\tilde{S}_0$ has *full column rank.*
The transformation matrix $\hat{T}_0$ *can be taken* orthogonal, or a well conditioned matrix, following remark 3.2. $\hat{P}_0$, on the other hand *must be* orthogonal, to preserve the optimization problem (3.24). Both transformations affect the corresponding parts of y and u in (3.24) as follows:

$$\hat{T}_0 \begin{pmatrix} \hat{b}_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} r_0 \\ \tilde{b}_0 \end{pmatrix} \quad ; \quad \hat{P}_0' \begin{pmatrix} \tilde{u}_0 \\ \tilde{v}_0 \end{pmatrix} = \begin{pmatrix} \tilde{v}_0^1 \\ \tilde{u}_0^1 \end{pmatrix} \tag{3.35}$$

From the second part, it becomes clear that $u'u$ in (3.24) remains unaltered. So that we are still solving the same optimization problem.
With these two transformations, the set of equations becomes:

$$\begin{pmatrix} r_0 \\ \tilde{b}_0 \\ -D_0 u_0 \\ \cdot \\ \cdot \end{pmatrix} = \begin{pmatrix} 0 & 0 & & \\ \overline{U}_0 & 0 & & \\ A_0 & -E_0 & & \\ & & \cdot & \\ & & & \cdot \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \end{pmatrix} +$$

$$\begin{pmatrix} \tilde{S}_0 & 0 & 0 & \\ \tilde{R}_0 & \overline{R}_0 & 0 & \\ \tilde{N}_0 & \overline{N}_0 & -B_0 Q_0^{1/2} & \\ & & & \cdot \\ & & & & \cdot \end{pmatrix} \begin{pmatrix} \tilde{v}_0^1 \\ \tilde{u}_0^1 \\ \tilde{w}_0 \\ \cdot \\ \cdot \end{pmatrix} \tag{3.36}$$

Since $\tilde{S}_0$ has full column rank we can compute $\tilde{v}_0^1$ uniquely from:

$$\tilde{S}_0 \tilde{v}_0^1 = r_0 \tag{3.37}$$

Using this solution, we define:

$$\overline{b}_0 = \tilde{b}_0 - \tilde{R}_0 \tilde{v}_0^1 \text{ and } \overline{d}_0 = -D_0 u_0 - \tilde{N}_0 \tilde{v}_0^1 \tag{3.38}$$

which gives for our model:

$$\begin{pmatrix} \bar{b}_0 \\ \bar{d}_0 \\ y_1 \\ \cdot \\ \cdot \end{pmatrix} = \begin{pmatrix} \overline{U}_0 & 0 & & \\ A_0 & -E_0 & & \\ 0 & C_1 & & \\ & & & \cdot \\ & & & \cdot \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \end{pmatrix} +$$

$$\begin{pmatrix} \overline{R}_0 & 0 & & \\ \overline{N}_0 & -B_0 Q_0^{1/2} & & \\ & & -R_1^{1/2} & \\ & & \cdot & \\ & & & \cdot \end{pmatrix} \begin{pmatrix} \tilde{u}_0^1 \\ \tilde{w}_0 \\ \cdot \\ \cdot \end{pmatrix} \tag{3.39}$$

Observe, that the first block row indicates the updated version of (3.18):

$$\bar{b}_0 = \overline{U}_0 x_0 + \overline{R}_0 \tilde{u}_0^1 \quad \text{with} \quad \tilde{u}_0^1 \frown (0, I) \tag{3.40}$$

which include the first measurement y_0. The solution $\hat{x}_{0|0}$, i.e. the filtered estimate, is then given by solving:

$$\bar{b}_0 = \overline{U}_0 \hat{x}_{0|0} \tag{3.41}$$

This can be computed if $\overline{U}_0$ has full row rank.

B. The time update

In the time update, the matrices A_0 and $\overline{N}_0$ in the second block row of (3.39) are eliminated. This may be achieved as follows.
First, a left transformation is performed:

$$\overline{T}_0 \left(\begin{array}{cc|cc} \bar{b}_0 & \overline{U}_0 & 0 & \overline{R}_0 & 0 \\ \bar{d}_0 & A_0 & -E_0 & \overline{N}_0 & -B_0 Q_0^{1/2} \end{array} \right) = \left(\begin{array}{ccc|cc} \star & U_0 & U_{01} & \star & \star \\ \hat{b}_1 & 0 & \hat{U}_1 & \star & \star \end{array} \right) \tag{3.42}$$

which makes U_0 full row rank.
And secondly,

$$\left(\begin{array}{cc|cc} U_0 & U_{01} & \star & \star \\ 0 & U_1 & \star & \star \end{array} \right) \left(\begin{array}{cc|cc} I & 0 & & \\ 0 & \overline{P}_0 & & \end{array} \right) = \left(\begin{array}{cc|cc} U_0 & U_{01} & S_0 & S_{01} \\ 0 & \hat{U}_1 & 0 & \hat{S}_1 \end{array} \right) \tag{3.43}$$

with the corresponding part of u in (3.24) given as:

$$\overline{P}_0' \begin{pmatrix} \tilde{u}_0^1 \\ \tilde{w}_0 \end{pmatrix} = \begin{pmatrix} u_0 \\ \tilde{u}_1 \end{pmatrix}$$

For $\overline{T}_0$ and $\overline{P}_0$, the same comments apply as in the measurement step. After this step the transformed model has the following structure:

$$\begin{pmatrix} \star \\ \hat{b}_1 \\ y_1 \\ -D_1 u_1 \\ \vdots \\ \cdot \end{pmatrix} = \left(\begin{array}{ccc} U_0 & U_{01} & \\ \hline 0 & \hat{U}_1 & \\ 0 & C_1 & \\ 0 & A_1 & -E_1 \\ & | & \\ & | & \end{array} \right) \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \end{pmatrix} +$$

$$\left(\begin{array}{cccc} S_0 & S_{01} & & \\ \hline 0 & \hat{S}_1 & & \\ 0 & 0 & -R_1^{1/2} & \\ 0 & 0 & -B_1 N_1^T & -B_1 Q_1^{1/2} \\ & | & & \cdot \\ & | & & \end{array} \right) \begin{pmatrix} u_0 \\ \tilde{u}_1 \\ \tilde{v}_1 \\ \tilde{w}_1 \\ \cdot \\ \cdot \end{pmatrix} \tag{3.44}$$

Since U_0 has full row rank and x_0 does not affect the following block rows of (3.44), the first block row and column may be deleted. The remaining problem has exactly the same form as the original problem (3.30), and so one step has been completed.

The one-step predicted estimate for x_1, satisfies:

$$\hat{b}_1 = \hat{U}_1 x_1 + \hat{S}_1 \tilde{u}_1 \tag{3.45}$$

and may be computed form:

$$\hat{b}_1 = \hat{U}_1 \hat{x}_{1|0} \tag{3.46}$$

Remark 3.4: The given estimate in (3.41) for the measurement step and for the one-step predicted estimate in (3.46) for the time step need not to be computed explicitly. As a matter of fact, during some time intervals, insufficient information may be available to determine these estimates uniquely. This does not restrict the generality of the presented method, since the given forms (3.40) and (3.45) can easily be included in subsequent recursions. After a certain number of steps, the estimates can again be determined uniquely, when the system and measurement equations supply sufficient information.

3.6. The derivation of two extremes of the general algorithm

In this section we will show that the general Kalman filter algorithm, described in the previous section, covers a whole class of reliable and recursive solutions to some particular Kalman filter problem formulation. This is done by deriving the two extremes of this class, which at the same time are existing algorithms. Namely, when using *only* covariance matrices the Square Root Covariance filter is obtained. And when using *only* inverse covariance matrices, the Square Root Information filter results. Other possible algorithms of this class form a mixture of these two extremes.

3.6.1. The Square Root Covariance filter

Here, the initial conditions are given as:

$$x_0 = \hat{x}_{0|-1} - \hat{S}_0\tilde{u}_0. \quad \tilde{u}_0 \frown (0, I) \tag{3.47}$$

and the process and measurement noise as:

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} v_k \\ w_k \end{pmatrix} - \begin{pmatrix} R_k^{1/2} & 0 \\ 0 & Q_k^{1/2} \end{pmatrix} \begin{pmatrix} \tilde{v}_k \\ \tilde{w}_k \end{pmatrix} \quad \begin{pmatrix} \tilde{v}_k \\ \tilde{w}_k \end{pmatrix} \frown (0, I) \tag{3.48}$$

The process noise and measurement noise given hereby could immediately be substituted in the state equations (3.1) and measurement equations (3.2), to result in the following set of equations:

$$\begin{pmatrix} \hat{x}_{0|-1} \\ y_0 \\ -D_0u_0 \\ y_1 \\ \cdot \\ \cdot \end{pmatrix} = \begin{pmatrix} I & & & \\ C_0 & & & \\ A_0 & -I & & \\ & C_1 & & \\ & & \cdot & \\ & & & \cdot \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \end{pmatrix} + \begin{pmatrix} \hat{S}_0 & & & \\ 0 & R_0^{1/2} & & \\ 0 & 0 & B_0Q_0^{1/2} & \\ & & \cdot & \\ & & & \cdot \end{pmatrix} \begin{pmatrix} \tilde{u}_0 \\ \tilde{v}_0 \\ \tilde{w}_0 \\ \cdot \\ \cdot \end{pmatrix} \tag{3.49}$$

again denoted in shorthand as:

$$y = Fx + Lu \quad u \frown (0, I) \tag{3.50}$$

Using the technique described in section 3.5 for this set of equations, one recursion of the square root covariance filter can be described by the following two steps.

A. The measurement update

This step operates on the first two block rows of F and the first two block columns of L in (3.50).
First, we use the following transformation to eliminate C_0 in F:

$$\begin{pmatrix} C_0 & -I \\ I & 0 \end{pmatrix} \left(\begin{array}{c|c|cc} \hat{x}_{0|-1} & I & \hat{S}_0 & 0 \\ y_0 & C_0 & 0 & R_0^{1/2} \\ \hline -D_0u_0 & A_0 & 0 & 0 \end{array}\right) = \left(\begin{array}{c|c|cc} C_0\hat{x}_{0|-1} - y_0 & 0 & C_0\hat{S}_0 & -R_0^{1/2} \\ \hat{x}_{0|-1} & I & \hat{S}_0 & 0 \\ \hline -D_0u_0 & A_0 & 0 & 0 \end{array}\right) \tag{3.51}$$

where $C_0\hat{x}_{0|-1} - y_0$ is the *innovation* at the time instant 0, denoted as ν_0 in figure 3.1. Secondly, $R_0^{1/2}$ in the right hand side of (3.51) is annihilated as follows:

$$\left(\begin{array}{c|c|cc} \nu_0 & 0 & C_0\hat{S}_0 & -R_0^{1/2} \\ \hat{x}_{0|-1} & I & \hat{S}_0 & 0 \\ -D_0u_0 & A_0 & 0 & 0 \end{array}\right) \left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & I & 0 \\ \hline 0 & 0 & \hat{P}_0 \end{array}\right) = \left(\begin{array}{c|c|cc} \nu_0 & 0 & (R_0^e)^{1/2} & 0 \\ \hat{x}_{0|-1} & I & G_0 & \overline{S}_0 \\ -D_0u_0 & A_0 & 0 & 0 \end{array}\right) \tag{3.52}$$

where again $\hat{P}_0$ is orthogonal and $(R_0^e)^{1/2}$, *automatically* obtained from the right hand side of (3.52), is the square root of the innovation covariance matrix of ν_0. Along with equation (3.38), the filtered estimate is given as:

$$\hat{x}_{0|0} = \hat{x}_{0|-1} - G_0(R_0^e)^{-1/2}\nu_0 \tag{3.53}$$

where the product $G_0(R_0^e)^{-1/2}$ is called the *Kalman gain* (represented by K_k in figure 3.1).
With (3.53), the second block row of (3.52) becomes:

$$\hat{x}_{0|0} = x_0 + S_0\tilde{u}_0^1 \quad \tilde{u}_0^1 \frown (0, I) \tag{3.54}$$

from which follows that $\overline{S}_0$ is the Choleski factor of the covariance matrix of the filtered estimate $P_{0|0} = E\{(x_0 - \hat{x}_{0|0})(x_0 - \hat{x}_{0|0})'\}$.

Remark 3.5: Computationally, the measurement update directly consists of (3.52) and (3.53) only. However, for a more detailed algorithmic description we refer to chapter 4.

B. The time update

The first part here eliminates A_0 in the third block row of (3.49) as follows:

$$\left(\begin{array}{cc} I & 0 \\ -A_0 & I \end{array}\right)\left(\begin{array}{cc|cc} \hat{x}_{0|0} & & I & 0 \\ -D_0u_0 & & A_0 & -I \end{array}\left|\begin{array}{cc} \overline{S}_0 & 0 \\ 0 & B_0Q_0^{1/2} \end{array}\right.\right) =$$

$$\left(\begin{array}{cc|cc} \hat{x}_{0|0} & & I & 0 \\ A_0\hat{x}_{0|0} + D_0u_0 & & 0 & I \end{array}\left|\begin{array}{cc} \overline{S}_0 & 0 \\ A_0\overline{S}_0 & B_0Q_0^{1/2} \end{array}\right.\right) \tag{3.55}$$

Secondly, one performs an orthogonal transformation $\overline{P}_0$ at the left:

$$(A_0\overline{S}_0 \;\; B_0Q_0^{1/2})\overline{P}_0 = (\hat{S}_1 \;\; 0) \tag{3.56}$$

such that $\hat{S}_1$ has full column rank. This converts (3.55) into:

$$\left(\begin{array}{cc|cc} \hat{x}_{0|0} & & I & 0 \\ A_0\hat{x}_{0|0} + D_0u_0 & & 0 & I \end{array}\left|\begin{array}{cc} \overline{S}_0 & 0 \\ A_0\overline{S}_0 & B_0Q_0^{1/2} \end{array}\right.\right)\left(\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & I & 0 \\ \hline 0 & 0 & \overline{P}_0 \end{array}\right) =$$

$$\left(\begin{array}{cc|cc} \hat{x}_{0|0} & & I & 0 \\ A_0\hat{x}_{0|0} + D_0u_0 & & 0 & I \end{array}\left|\begin{array}{cc} S_{01} & S_0 \\ \hat{S}_1 & 0 \end{array}\right.\right) \tag{3.57}$$

After this time update, (3.49) becomes:

$$\left(\begin{array}{c} \hat{x}_{0|0} \\ A_0\hat{x}_{0|0} + D_0u_0 \\ y_1 \\ -D_1u_1 \\ \cdot \\ \cdot \end{array}\right) = \left(\begin{array}{ccc} I & & \\ 0 & I & \\ 0 & C_1 & \\ 0 & A_1 & -I \\ & & \cdot \\ & & \cdot \end{array}\right)\left(\begin{array}{c} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \end{array}\right) +$$

$$\left(\begin{array}{cccc} S_0 & S_{01} & & \\ 0 & \hat{S}_1 & & \\ & & R_1^{1/2} & \\ & & \cdot & \\ & & & \cdot \end{array}\right)\left(\begin{array}{c} \tilde{u}_0^1 \\ \tilde{u}_1 \\ \tilde{v}_1 \\ \tilde{w}_1 \\ \cdot \\ \cdot \end{array}\right) \tag{3.58}$$

This is analogous to (3.44) and one recursion is completed. The second block row of (3.58) determines the one-step predicted estimate $\hat{x}_{1|0}$:

$$\hat{x}_{1|0} = A_0\hat{x}_{0|0} + D_0u_0 \tag{3.59}$$

while the square root of its covariance matrix $P_{1|0}$ given by $\hat{S}_1$.

Remark 3.6: Computationally, the only equations required to characterize the time-step of the square root covariance filter are (3.56) and (3.59).

3.6.2. The Square Root Information filter

The opposite of the previous filter is the square root information filter. Here, all the covariance data are given via the inverse covariance matrices. And, more particularly, we now start with the filtered estimate, and first perform the time update. Furthermore, it will be shown that a combined measurement and time update will automatically result. The derivation of the filter goes as follows.
The required set of equations similar to (3.23) follows straightforwardly from the initial conditions:

$$\overline{S_0}^{-1}x_0 = \overline{S_0}^{-1}\hat{x}_{0|0} - \tilde{u}_0 \quad \tilde{u}_0 \frown (0, I) \tag{3.60}$$

where $\overline{S_0}^{-1}$ is the square root of the inverse covariance matrix of $\hat{x}_{0|0}$ and from the process and measurement noise covariance representation:

$$\left(\begin{array}{c} 0 \\ 0 \end{array}\right) = \left(\begin{array}{cc} R_k^{-1/2} & 0 \\ 0 & Q_k^{-1/2} \end{array}\right)\left(\begin{array}{c} v_k \\ w_k \end{array}\right) + \left(\begin{array}{c} \tilde{v}_k \\ \tilde{w}_k \end{array}\right) \quad \left(\begin{array}{c} \tilde{v}_k \\ \tilde{w}_k \end{array}\right) \frown (0, I) \tag{3.61}$$

In contrast to the square root covariance filter, now only the representation of v_k could directly be included in the measurement equation (3.2), resulting in:

$$R_k^{-1/2}y_k = R_k^{-1/2}C_kx_k + \tilde{v}_k \tag{3.62}$$

The above representation combined with the state equations gives:

$$\begin{pmatrix} \overline{S_0}^{-1}\hat{x}_{0|0} \\ -D_0u_0 \\ 0 \\ R_1^{-1/2}y_1 \\ . \\ . \end{pmatrix} = \begin{pmatrix} \overline{S_0}^{-1} & 0 & 0 & \\ A_0 & B_0 & -I & \\ 0 & Q_0^{-1/2} & 0 & \\ 0 & 0 & R_1^{-1/2}C_1 & \\ & & & . \end{pmatrix} \begin{pmatrix} x_0 \\ w_0 \\ x_1 \\ w_1 \\ . \\ . \end{pmatrix} + \begin{pmatrix} \tilde{u}_0 \\ 0 \\ \tilde{w}_0 \\ \tilde{v}_1 \\ . \\ . \end{pmatrix} \tag{3.63}$$

Remark 3.7: This particular set of equations exactly corresponds with the set of equations occurring in the classical least squares problem, briefly denoted as:

$$\min_x v'v \quad \text{subject to:} \quad y = Fx + v \tag{3.64}$$

And a reliable way to solve this problem, is to transform this set of equations in (3.64) by an orthogonal transformation, applied in front, to give a new set of the following structure:

$$\begin{pmatrix} y_1^\star \\ y_2^\star \end{pmatrix} = \begin{pmatrix} R^\star \\ 0 \end{pmatrix} x + \begin{pmatrix} v_1^\star \\ v_2^\star \end{pmatrix} \tag{3.65}$$

Here the index $\star$ denotes the transformed quantities. If in (3.65) $R^\star$ is square and invertible, we can solve for the estimated value of x. With this particular solution the performance of the estimation criterion in (3.64) becomes

$$(y_2^\star)'y_2^\star$$

which is the minimum value of the optimization problem in the 2-norm sense. In a similar way we can interpret the previously handled optimization problems.

Using this remark, we can solve the square root information problem. However, to increase the efficiency we first *decouple* the first and second block row in (3.63). This is done by multiplying the second row by A_0^{-1} and then performing the following transformation:

$$\begin{pmatrix} I & 0 \\ -I & \overline{S_0}^{-1} \end{pmatrix} \left(\begin{array}{cc|ccc} \overline{S_0}^{-1}\hat{x}_{0|0} & & \overline{S_0}^{-1} & 0 & 0 & \tilde{u}_0 \\ -A_0^{-1}D_0u_0 & & I & A_0^{-1}B_0 & -A_0^{-1} & 0 \end{array} \right) =$$

$$\left(\begin{array}{cc|ccc} \overline{S_0}^{-1}\hat{x}_{0|0} & & \overline{S_0}^{-1} & 0 & 0 & \tilde{u}_0 \\ -\overline{S_0}^{-1}(\hat{x}_{0|0} + A_0^{-1}D_0u_0) & & 0 & \overline{S_0}^{-1}A_0^{-1}B_0 & -\overline{S_0}^{-1}A_0^{-1} & -\tilde{u}_0 \end{array} \right) \tag{3.66}$$

This allows to drop the first block row. Observing the second row of the r.h.s. of (3.66), it is important to remark that the v-vector of the sub-optimization problem of (3.64) is unaltered. Now, to be compatible with the notation of the covariance data, given previously, we denote $\overline{S_i}^{-1}$ by $\overline{U}_i$. After this, we apply the orthogonal transformation P_0 as follows:

$$P_0 \left(\begin{array}{cc|c} \overline{U}_0A_0^{-1}B_0 & \overline{U}_0A_0^{-1} & \overline{U}_0(\hat{x}_{0|0} + A_0^{-1}D_0u_0) \\ Q_0^{-1/2} & 0 & 0 \\ 0 & R_1^{-1/2}C_1 & R_1^{-1/2}y_1 \end{array} \right) =$$

$$\left(\begin{array}{cc|c} (Q_0^e)^{-1/2} & Q_{01} & \overline{w}_0 \\ 0 & \overline{U}_1 & \hat{s}_{1|1} \\ 0 & 0 & \nu_1 \end{array} \right) \tag{3.67}$$

with $\hat{s}_{1|1} = \overline{U}_1\hat{x}_{1|1}$. This allows to solve for the filtered state estimate $\hat{x}_{1|1}$ from:

$$\overline{U}_1\hat{x}_{1|1} = \hat{s}_{1|1} \tag{3.68}$$

if $\overline{U}_1$ is invertible. Here, $\overline{U}_1$ is the square root of the inverse of the covariance matrix $P_{1|1}$.

3.7. Concluding remarks

In this chapter a new theoretical derivation of the Kalman filter is presented. This method is on the one hand very instructive since it is only based on the classical least squares method, and on the other hand, acts as a "general unified recipe of a cookbook to prepare a meal (*to derive a filter implementation*) best suited to one's own taste on a particular occasion (*best suited from the point of view of efficiency and reliability for a particular set of system conditions*)". Furthermore, the presented technique covers all existing implementations applicable to the general *discrete time-varying* case. This is shown by deriving the square root covariance filter and the square root information filter implementations. From these filters the original Kalman filter, which formulates a recursion for the full state

error covariance matrix or its inverse, is derived in a much more natural way (simply by multiplying out). The opposite direction as followed by many authors e.g. [1], [51] etc. is not that straightforward. This will be indicated explicitly in the next chapter.

Chapter 4.

The Reliability and Efficiency of different Kalman filter algorithms

4.1. Introduction

Since the appearance of Kalman's 1960 paper [51], the so-called Kalman filter (KF) has been applied successfully to many practical problems, especially in aeronautical and aerospace applications. As applications became more numerous, some pitfalls of the KF were discovered as well. Here we think especially of the problem of *divergence*, which is the loss of any practical value of the numerical estimates (i.e. the difference between the true parameter values and their estimates becomes unacceptably large). Reasons for this may be of two different natures: first, the *lack of reliability of the numerical algorithm* may lead to the loss of symmetry and/or positivity of the state covariance matrix P, and, secondly, *inaccurate modeling* of the system under consideration as reported e.g. by J. Jazwinski [43].

Therefore, a lot of effort has been spent on modifying the numerical implementation of the KF. Many of these modifications were based on heuristics, as e.g. in the stabilized KF [9], or when averaging the off-diagonal elements of P during each recursion of the KF, or also the conventional KF with lower bounding [14]. The latter illustrates nicely the *inefficiency* of such heuristic procedures. In this particular modification, a lower bound is specified for the diagonal elements of P and an upper bound for the off-diagonal elements of P. These two bounds are evaluated by a "cut and try" method in a time consuming simulation study. Even when "good" bounds

have been obtained for the particular application, there is still no guarantee that the filter will work for slightly modified experimental conditions, which makes such a heuristic procedure rather inattractive. The lack of absolute certainty combined with the increased computational burden (to compensate for the numerical errors performed by the algorithm) makes the use of such a method impossible for real-time applications.

Later on, more reliable KF implementations were described such as the square root filters (SRF) proposed by Potter in 1963 [85]. For these filters the reliability of the filter estimates is expected to be better because of the use of numerically stable orthogonal transformations for each recursion step (e.g. Householder, Givens or modified Gram-Schmidt methods). Claims of better numerical performance were, however, mainly based on experimental results and not on an error analysis. On the other hand, the SRF implementations require more computations than the conventional KF [52]. This problem of cost efficiency on itself gave rise to the development of modified versions of the SRF's as e.g. the UDU'-algorithms [8] and the Chandrasekhar form [49]. These implementations can be made as efficient as the conventional KF, or for the Chandrasekhar SRF even more efficient for some special experimental conditions.

In this chapter we reconsider the above two issues and derive some new results giving new and/or better insights into the numerical performance of the following four KF implementations:

- The Conventional Kalman Filter
- The Square Root Covariance Filter
- The Chandrasekhar Square Root Filter
- The Square Root Information Filter

After a brief description of these filters in section 4.2, we first consider the issue of numerical robustness. Therefore we perform in section 4.3 a detailed analysis of the error propagation due to three different kind of error sources for the above four KF implementations. The subsequent considered error sources are *round-off, modeling errors and discretization errors.* It will be shown that the *approximations* derived for the first two error sources fit in the same framework, while the third error is treated in some way separately. Such an analysis shortcuts the need of a laborious simulation study in which all possible experimental conditions are examined. Secondly, the issue of the complexity of these implementations is addressed. Since the Chandrasekhar implementation is only valid for the time-invariant optimal

filtering problem, the question of efficiency is restricted to that case. In section 4.4, new variants of the conventional KF, the square root covariance filter and the square root information filter are presented that are mainly restricted to the time-invariant case. These implementations are based on the condensed forms (2.11), (2.12) and (2.13), discussed in chapter 2. These forms are shown to yield considerable savings in computations during the subsequent filter recursion. Again we stress the fact that since unitary transformations can be used to obtain the initial condensed form, *no loss of accuracy* is induced by the required coordinate transformation.

Section 4.5 then outlines a comparison between the different filter implementations using the results of the theoretical error analysis of section 4.3 and the operation counts for the new filter implementations of section 4.4. For simulations illustrating the analysis and the operation counts of the different methods we refer to the next chapter and [99]. Here very clear indications are given about which KF implementation is to be preferred depending on the problem at hand. Finally we end with some concluding remarks in section 4.6.

4.2. The four different Kalman Filter implementations

The discrete KF problem with notation of the system model to match have been defined in the previous chapter, see sections 3.2 and 3.3. In contrast to the general method presented in chapter 3 to solve the KF problem, we will consider in this chapter three different types of *classic* KF implementations, which are valid for the general *time-variant* case. Hence, following the concluding remarks of the previous chapter, these implementations can be derived in a straightforward way from the general algorithm of section 3.5.2. This is shown for the square root covariance filter and the square root information filter in sections 3.6.1 and 3.6.2 respectively. In the sequel, we will recall the recursions for these 3 KF implementations. The fourth KF implementation distinguishes itself from the previous ones, in that it is only valid for the *time-invariant* case. Therefore, this implementation can not be derived from the presented general algorithm of section 3.5.2 *in a straightforward manner*. Hence, the recursions of this method will be treated here to some larger extent.

4.2.1. The Square Root Covariance Filter (SRCF)

The measurement and time step derived for the SRCF in section 3.6.1 can easily be given in a combined way. In this case, the combined recursion propagates the Choleski factors of the error covariance matrix $P_{k|k-1}$:

$$P_{k|k-1} = S_k . S_k' \tag{4.1}$$

where S_k is chosen to be lower triangular. Since in this combined recursion, we will only work with the Choleski factor of the one-step predicted error covariance matrix we drop the $(\hat{\cdot})$ and $(\bar{\cdot})$ notation we used in the previous chapter to distinguish the filtered and one-step predicted quantities. Computationally, such a combined measurement and time update for S_k results by combining equations (3.52) and (3.56):

$$\underbrace{\begin{pmatrix} R_k^{1/2} & C_k S_k & 0 \\ 0 & A_k S_k & B_k Q_k^{1/2} \end{pmatrix}}_{\text{(pre-array)}} . U_1 = \underbrace{\begin{pmatrix} R_k^{e1/2} & 0 & 0 \\ G_k & S_{k+1} & 0 \end{pmatrix}}_{\text{(post-array)}} \tag{4.2}$$

On the other hand, the combination of equations (3.53) and (3.59) results in an update for the one-step predicted state estimate:

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k-1} - G_k R_k^{e-1/2} (C_k \hat{x}_{k|k-1} - y_k) + D_k u_k \tag{4.3}$$

U_1 in (4.2) is an arbitrary orthogonal transformation that triangularizes the pre-array. Such a triangularization can e.g. be obtained using Householder transformations, see section 2.2.3 and [33]. This recursion is now initiated with $\hat{x}_{0|-1}$ and the Choleski factor S_0 of $P_{0|-1}$ as defined in (4.1). The number of flops needed for (4.2-4.3) is given in the following table:

symbol	Mathematical expression	number of flops
M_{11}	$R_k^{1/2}$	
M_{23}	$B_k Q_k^{1/2}$	$nm^2/2$
M_{12}	$C_k S_k$	$n^2p/2$
M_{22}	$A_k S_k$	$n^3/2$
	annihilation M_{12} (Householder)	$2n^2p+np^2$
	further reduction (Householder)	$2/3n^3+n^2m$
$diag\{P_{k+1\|k}\}$	$diag\{S_{k+1}.S'_{k+1}\}$	$n^2/2$
$\hat{x}_{k+1\|k}$	$A_k.\hat{x}_{k\|k-1}$	
	$-G_k.R_k^{e-1/2}.(C_k.\hat{x}_{k\|k-1}-y_k)$	$n^2+2np+p^2/2$
	$+D_k.u_k$	$+nr$

Total number of flops: [1] $\frac{7}{6}n^3+n^2(\frac{5p}{2}+m)+n(p^2+\frac{m^2}{2})$

Table 4.1: Numerical implementation of the SRCF

In order to reduce the amount of work, we only computed the diagonal elements of the covariance matrix $P_{k+1|k}$, since usually $diag\{P_{k+1|k}\}$ carries enough information about the estimate $\hat{x}_{k+1|k}$. For this reason our operation counts differ e.g. from those of [52].

4.2.2. The Square Root Information Filter (SRIF)

The information filter accentuates the recursive least squares nature of filtering [6], [1]. This is indicated clearly for the SRIF derived in section 3.6.2. Here the SRIF propagates the Choleski factor of $P_{k|k}^{-1}$ using the Choleski factor of the inverses of the process- and measurement noise covariance matrices:

$$P_{k|k}^{-1} = T'_k.T_k \tag{4.4}$$

$$Q_k^{-1} = [Q_k^{-1/2}]'.Q_k^{-1/2} \tag{4.5}$$

$$R_k^{-1} = [R_k^{-1/2}]'.R_k^{-1/2} \tag{4.6}$$

[1] The total number of flops does not include second order terms

where the right factors are all chosen to be upper triangular. The Choleski factor T_k of $P_{k|k}^{-1}$, defined in (4.4), can be taken as $(\overline{U})$ in (3.67), which for this notation becomes:

$$U_2.\underbrace{\begin{pmatrix} Q_k^{-1/2} & 0 & 0 \\ T_k A_k^{-1} B_k & T_k A_k^{-1} & \hat{s}_{k|k} \\ 0 & R_{k+1}^{-1/2} C_{k+1} & R_{k+1}^{-1/2} y_{k+1} \end{pmatrix}}_{\text{(pre-array)}} = \underbrace{\begin{pmatrix} Q_k^{e\ -1/2} & * & * \\ 0 & T_{k+1} & \hat{s}_{k+1|k+1} \\ 0 & 0 & \nu_{k+1} \end{pmatrix}}_{\text{(post-array)}} \tag{4.7}$$

with $\hat{s}_{i|i} = T_i \hat{x}_{i|i}$.

This filter recursion is initiated with $\hat{x}_{0|0}$ and the Choleski factor T_0 of $P_{0|0}^{-1}$ as defined in (4.4). An operation count of this filter is given below:

symbol	Mathematical expression	number of flops
M_{11}	$Q_k^{-1/2}$	
M_{21}	$T_k A_k^{-1} B_k$	$3/2n^2m$
M_{22}	$T_k A_k^{-1}$	$n^3/2$
M_{32}	$R_{k+1}^{-1/2} C_{k+1}$	$np^2/2$
M_{23}	$T_k \hat{x}_{k\|k}$	$n^2/2$
M_{33}	$R_{k+1}^{-1/2} y_{k+1}$	$p^2/2$
	annihilation M_{21} (Householder)	$2n^2m+nm^2$
	further reduction (Householder)	$2/3n^3+n^2p$
$\hat{x}_{k+1\|k+1}$	$T_{k+1}^{-1}.(T_{k+1}\hat{x}_{k+1\|k+1})+D_k.u_k$	$n^2/2+nr$

Total number of flops: $\frac{7}{6}n^3+n^2(p+\frac{7m}{2})+n(\frac{p^2}{2}+m^2)$

Table 4.2: Numerical implementation of the SRIF

Here we did not count the operations needed for the inversion and/or factorization of Q_k, R_k and A_k (for the time-invariant case e.g. these are

computed only once) and the filtered state estimate is computed by,

$$\hat{x}_{k+1|k+1} = T_{k+1}^{-1}(\hat{s}_{k+1|k+1}) + D_k u_k \tag{4.8}$$

4.2.3. The Conventional Kalman Filter (CKF)

The conventional KF indicates a whole set of KF implementations which in their "covariance form" propagate the state error covariance matrix. As indicated in the concluding remarks of the previous chapter, these filter recursions can easily be derived from its square root variant, i.e. the SRCF. Hereto, both sides of equation (4.2) are multiplied by their transposes, resulting in:

$$\begin{pmatrix} R_k + C_k P_{k|k-1} C_k' & C_k P_{k|k-1} A_k' \\ A_k P_{k|k-1} C_k' & A_k P_{k|k-1} A_k' + B_k Q_k B_k' \end{pmatrix} =$$

$$\begin{pmatrix} R_k^e & (R_k^e)^{1/2} G_k' \\ G_k (R_k^e)^{1/2} & P_{k+1|k} + G_k G_k' \end{pmatrix} \tag{4.9}$$

where we used (4.1).

From eq. (4.9) we then find:

$$R_k^e = R_k + C_k P_{k|k-1} C_k' \tag{4.10}$$

$$G_k = A_k P_{k|k-1} C_k' (R_k^e)^{-1/2} \tag{4.11}$$

$$P_{k+1|k} = A_k P_{k|k-1} A_k' + B_k Q_k B_k' - G_k G_k' \tag{4.12}$$

This set of equations (4.10-4.12) have been implemented in various different forms, possessing different properties towards efficiency and reliability, see e.g. [1]. The implementation that is used in this thesis to represent the CKF is summarized by the following separated measurement and time update equations:

Measurement update - filtered estimate

$$R_k^e = R_k + C_k P_{k|k-1} C_k' \tag{4.13}$$

$$K_k = A_k P_{k|k-1} C_k' [R_k^e]^{-1} \tag{4.14}$$

$$P_{k|k} = [I - P_{k|k-1} C_k' [R_k^e]^{-1} C_k] P_{k|k-1} \tag{4.15}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} - P_{k|k-1} C_k' [R_k^e]^{-1} [C_k \hat{x}_{k|k-1} - y_k] \tag{4.16}$$

Time update - predicted estimate

$$P_{k+1|k} = A_k P_{k|k} A_k' + B_k Q_k B_k' \tag{4.17}$$

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k} + D_k u_k \tag{4.18}$$

The number of operations needed for each step are summarized in the following table of operations:

symbol	Mathematical expression	number of flops
CP	$C_k.P_{k\|k-1}$	n^2p
R_k^e	$CP.[C_k]' + R_k$	$np^2/2$
K_k	$A_k.CP'.[R_k^e]^{-1}$	$p^3/6+np^2+n^2p$
$P_{k+1\|k}$	$(A_k.P_{k\|k-1} - K_k.CP).A_k'$	$1/2n^2(3n+2p)$
	$+B_k.Q_k.B_k'$	$1/2n^2m + nm^2$
$\hat{x}_{k+1\|k}$	$A_k.\hat{x}_{k\|k-1} - K_k.(C_k.\hat{x}_{k\|k-1} - y_k)$	n^2+2np
	$+D_k.u_k$	$+nr$

Total number of flops: $\frac{3}{2}n^3 + n^2(3p + \frac{m}{2}) + n(\frac{3p^2}{2} + m^2) + \frac{p^3}{6}$

Table 4.3: Numerical implementation of the CKF

4.2.4. The Chandrasekhar Square Root Filter (CSRF)

If the system model (3.1-3.2) is *time-invariant*, it is possible to formulate recursions for the increment of the state error covariance matrix only [49]. The increment is defined as follows:

$$inc.P_k = P_{k+1|k} - P_{k|k-1} \tag{4.19}$$

From equation (4.12) we immediately have a difference equation for $inc.P_k$ given as:

$$inc.P_{k+1} = Ainc.P_k A' - G_k G_k' + G_{k-1} G_{k-1}' \tag{4.20}$$

The CSRF formulates a recursion for the "square root" of the $inc.P_k$. However, it is difficult to speak here about the square root or Choleski factor

since in general the $inc.P_k$ is not positive definite. Therefore an alternative factorization is used:

$$inc.P_k = L_k.\underbrace{\begin{pmatrix} I_{n_1} & 0 \\ 0 & -I_{n_2} \end{pmatrix}}_{\Sigma}.L_k' \tag{4.21}$$

where the rank of $inc.P_k$ is $n_1 + n_2$ and Σ is called its signature matrix. The CSRF propagates recursions for L_k, through the following scheme [74]:

$$\underbrace{\begin{pmatrix} R_{k-1}^{e1/2} & CL_{k-1} \\ G_{k-1} & AL_{k-1} \end{pmatrix}}_{\text{(pre-array)}}.U_3 = \underbrace{\begin{pmatrix} R_k^{e1/2} & 0 \\ G_k & L_k \end{pmatrix}}_{\text{(post-array)}} \tag{4.22}$$

with $L_0 \Sigma L_0' = P_{1|0} - P_{0|-1}$. Here U_3 is a Σ_p - unitary transformation, i.e. $U_3 \Sigma_p U_3' = \Sigma_p$ with

$$\Sigma_p = \begin{pmatrix} I_p & 0 \\ 0 & \Sigma \end{pmatrix} \tag{4.23}$$

The square root formulation (4.22) can be verified by multiplying both sides of (4.22) with their transposes. This indeed results in the difference equation (4.20), apart from redefining R_k^e and G_k for the time invariant case. The transformation U_3 is easily constructed using "skew Householder" transformations (using the indefinite Σ-norm) and requires as many operations as the classical Householder transformations. An explicit discussion of the skew Householder transformation is given in Appendix B or [74]. However, numerical errors may blow up in proportion to the condition number $\kappa(U_3)$ of the transformation that is applied and this may become very large for some problems. The estimate $\hat{x}_{k+1|k}$ is again computed from its previous value via

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k-1} - G_k R_k^{e-1/2}(C\hat{x}_{k|k-1} - y_k) + Du_k \tag{4.24}$$

For this implementation the operation count is summarized in the following table:

symbol	Mathematical expression	number of flops
M_{11}	$R_{k-1}^{e1/2}$	
M_{21}	G_{k-1}	
M_{12}	CL_{k-1}	$(n_1+n_2)np$
M_{22}	AL_{k-1}	$(n_1+n_2)n^2$
	annihilation M_{12} (skew Householder)	$(n_1+n_2)(p^2+2np)$
$diag\{P_{k+1\|k}\}$	$diag\{P_{k\|k-1} + L_k.\Sigma.L_k'\}$	$(n_1+n_2)n$
$\hat{x}_{k+1\|k}$	$A.\hat{x}_{k\|k-1}$	
	$-G_k.R_k^{e-1/2}.(C.\hat{x}_{k\|k-1} - y_k)$	$n^2+2np+p^2/2$
	$+D.u_k$	$+nr$

Total number of flops: $(n_1+n_2)(n^2+3np+p^2)$

Table 4.4: Numerical implementation of the CSRF

4.3. Theoretical Error analysis

4.3.1. Round-off errors

In this section we first analyze the effect of rounding errors on Kalman filtering in the four different implementations described above. The analysis is split in three parts:

- what bounds can be obtained for the errors performed in step k
- how do errors performed in step k propagate in subsequent steps
- how do errors performed in different steps interact and accumulate

Although this appears to be the logical order in which one should treat the problem of error build-up in KF, we first look at the second aspect, which is also the only one that has been studied in the literature so far. Therefore, we first need the following lemma.

Lemma 4.1: (The matrix inversion lemma)
Let A and C be non-singular $m \times m$ and $n \times n$ matrices, respectively, then the following formula holds:

$$(A+BCD)^{-1} = A^{-1} - A^{-1}B(DA^{-1}B + C^{-1})^{-1}DA^{-1} \tag{4.25}$$

Proof: This formula is simply verified by using the definition of the inverse of a matrix.

■

Lemma 4.2: Let A be a square non-singular matrix with smallest singular value σ_{min} and let E be a perturbation of the order of

$$\delta = \|E\|_2 << \sigma_{min}(A) \tag{4.26}$$

Then

$$\begin{aligned}(A+E)^{-1} &= A^{-1} - A^{-1}E(A+E)^{-1} = A^{-1} - (A+E)^{-1}EA^{-1} \\ &= A^{-1} + \Delta_1\end{aligned} \tag{4.27}$$

$$\begin{aligned}(A+E)^{-1} &= A^{-1} - A^{-1}EA^{-1} + A^{-1}E(A+E)^{-1}EA^{-1} \\ &= A^{-1} - A^{-1}EA^{-1} + \Delta_2\end{aligned} \tag{4.28}$$

where

$$\|\Delta_1\|_2 \leq \delta/\sigma_{min}(\sigma_{min} - \delta) = O(\delta) \tag{4.29}$$

$$\|\Delta_2\|_2 \leq \delta^2/\sigma_{min}^2(\sigma_{min} - \delta) = O(\delta^2) \tag{4.30}$$

Notice that when A and E are symmetric, the first and second order approximations (4.27) and (4.28) are also symmetric

Proof: Using lemma 4.1 with B and D taken equal to the identity matrix and $C = E$, equation (4.27) results. A second application of the matrix inversion lemma 4.1 then gives (4.28).

■

4.3.1.1. Propagation of errors using infinite arithmetic

We now thus consider the propagation of errors from step k to step $k+1$ *when no additional errors are performed during that update.* We denote the quantities in the computer with an upperbar, i.e. $\overline{P}_{k|k-1}$, $\overline{\hat{x}}_{k|k-1}$, $\overline{G}_k$, $\overline{S}_k$, $\overline{T}_k$, $\overline{R}_k^{e1/2}$, $\overline{F}_k$, $\overline{K}_k$, or $\overline{L}_k$, depending on the algorithm.

For the CKF, let $\delta P_{k|k-1}$ and $\delta x_{k|k-1}$ be the accumulated errors in step k, then:

$$\overline{P}_{k|k-1} = P_{k|k-1} + \delta P_{k|k-1}\ ,\ \overline{\hat{x}}_{k|k-1} = \hat{x}_{k|k-1} + \delta\hat{x}_{k|k-1} \tag{4.31}$$

By using Lemma 4.2 for the inverse of:

$$\overline{R}_k^e = R_k^e + C_k\delta P_{k|k-1}C_k' \tag{4.32}$$

we find

$$[\overline{R}_k^e]^{-1} = [R_k^e]^{-1} - [R_k^e]^{-1}C_k\delta P_{k|k-1}C_k'[R_k^e]^{-1} + O(\delta^2) \tag{4.33}$$

From this one then derives:

$$\begin{aligned}\overline{K}_k &= A_k\overline{P}_{k|k-1}C_k'\overline{R}_k^{e-1} \\ \delta K_k &= A_k\delta P_{k|k-1}C_k'R_k^{e-1} \\ &\quad -A_kP_{k|k-1}C_k'R_k^{e-1}C_k\delta P_{k|k-1}C_k'R_k^{e-1} + O(\delta^2) \\ &= F_k\delta P_{k|k-1}C_k'R_k^{e-1} + O(\delta^2)\end{aligned} \tag{4.34}$$

where

$$F_k = A_k(I - P_{k|k-1}C_k'R_k^{e-1}C_k) = A_k - K_kC_k \tag{4.35}$$

and (assuming $\overline{P}_{k|k-1}$ is not symmetric):

$$\begin{aligned}\overline{P}_{k+1|k} &= A_k(\overline{P}_{k|k-1} - \overline{P}'_{k|k-1}C_k'\overline{R}_k^{e-1}C_k\overline{P}_{k|k-1})A_k' + B_kQ_kB_k' \\ \delta P_{k+1|k} &= A_k(\delta P_{k|k-1} - \delta P'_{k|k-1}C_k'R_k^{e-1}C_kP_{k|k-1} \\ &\quad -P_{k|k-1}C_k'R_k^{e-1}C_k\delta P_{k|k-1} \\ &\quad +P_{k|k-1}C_k'R_k^{e-1}C_k\delta P_{k|k-1}C_k'R_k^{e-1}C_kP_{k|k-1})A_k' + O(\delta^2) \\ &= (A_k - K_kC_k)\delta P_{k|k-1}(A_k' - C_k'K_k') \\ &\quad +A_k(\delta P_{k|k-1} - \delta P'_{k|k-1})C_k'K_k' + O(\delta^2) \\ &= F_k\delta P_{k|k-1}F_k' + A_k(\delta P_{k|k-1} - \delta P'_{k|k-1})A_k' \\ &\quad -A_k(\delta P_{k|k-1} - \delta P'_{k|k-1})F_k' + O(\delta^2)\end{aligned} \tag{4.36}$$

For the estimate $\hat{x}_{k+1|k}$ we have:

$$\begin{aligned}\overline{\hat{x}}_{k+1|k} &= \overline{F}_k\overline{\hat{x}}_{k|k-1} + \overline{K}_ky_k + D_ku_k \\ \delta\hat{x}_{k+1|k} &= F_k\delta\hat{x}_{k|k-1} + \delta F_k\hat{x}_{k|k-1} + \delta K_ky_k + O(\delta^2) \\ &= F_k\delta\hat{x}_{k|k-1} + \delta K_k(y_k - C_k\hat{x}_{k|k-1}) + O(\delta^2) \\ &= F_k[\delta\hat{x}_{k|k-1} + \delta P_{k|k-1}C_k'R_k^{e-1}(y_k - C_k\hat{x}_{k|k-1})] + O(\delta^2)\end{aligned} \tag{4.37}$$

When on the other hand $\delta P_{k|k}$ and $\hat{x}_{k|k}$ are given, one derives analogously,

$$\delta P_{k+1|k+1} = \tilde{F}_k\delta P_{k|k}\tilde{F}_k' + A_k(\delta P_{k|k} - \delta P'_{k|k})A_k' - A_k(\delta P_{k|k} - \delta P'_{k|k})\tilde{F}_k' + O(\delta^2) \tag{4.38}$$

where

$$\tilde{F}_k = (I - P_{k|k-1}C_k'R_k^{e-1}C_k)A_k \tag{4.39}$$

has the same spectrum as F_k since $F_k A_k = A_k \tilde{F}_k$ [29].

We thus find that when $\delta P_{k|k-1}$ or $\delta P_{k|k}$ is symmetric, only the first term in (4.36) or (4.38) remains and the error propagation behaves roughly as:

$$||\delta P_{k+1|k}||_2 \approx ||F_k||_2^2 . ||\delta P_{k|k-1}||_2 = \gamma_k^2 . ||\delta P_{k|k-1}||_2 \tag{4.40}$$

$$||\delta P_{k+1|k+1}||_2 \approx ||\tilde{F}_k||_2^2 . ||\delta P_{k|k}||_2 = \tilde{\gamma}_k^2 . ||\delta P_{k|k}||_2 \tag{4.41}$$

which are decreasing in time when F_k and $\tilde{F}_k$ are contractions (i.e. when γ_k and $\tilde{\gamma}_k < 1$). The latter is usually the case when the matrices A_k, B_k, C_k, Q_k and R_k do not vary too wildly in time [1]. For the time-invariant case one can improve on this by saying that F_k and $\tilde{F}_k$ tend to the constant matrices F_∞ and $\tilde{F}_\infty$ respectively, with (equal) spectral radius $\rho_\infty < 1$ and one then has for some appropriate matrix norm [7]:

$$||\delta P_{k+1|k}|| \approx \rho_\infty^2 . ||\delta P_{k|k-1}|| \tag{4.42}$$

$$||\delta P_{k+1|k+1}|| \approx \rho_\infty^2 . ||\delta P_{k|k}|| \tag{4.43}$$

for sufficiently large k. Notice that ρ_∞ is smaller than γ_∞ or $\tilde{\gamma}_\infty$, whence (4.42-4.43) are better bounds than (4.40-4.41). Using this, it then also follows from (4.42-4.43) that all three errors $\delta P_{k|k-1}$, δK_k and $\delta \hat{x}_{k|k-1}$ are decreasing in time when no additional errors are performed. The fact that past errors are weighted in such a manner is the main reason why Kalman filtering does not diverge in presence of rounding errors: the total error at step $k+1$ indeed mainly depends on the (new) errors performed in the updating of the equations and *not* on past errors as we will see later on.

The property (4.40-4.43) was already observed before [43], again for symmetric $\delta P_{k|k-1}$. However if symmetry is removed, divergence may occur when A_k (i.e. the original plant) is unstable. Indeed, from (4.36)(4.38) we see that when A_k is unstable the larger part of the error is skew symmetric:

$$\delta P_{k+1|k} \approx A_k . (\delta P_{k|k-1} - \delta P'_{k|k-1}) . A'_k \tag{4.44}$$

$$\delta P_{k+1|k+1} \approx A_k . (\delta P_{k|k} - \delta P'_{k|k}) . A'_k \tag{4.45}$$

and the lack of symmetry *diverges* as k increases. A simple way to avoid this is to symmetrize $\overline{P}_{k|k-1}$ or $\overline{P}_{k|k}$ at each step by averaging it with its transpose. This makes the errors symmetric and the largest terms in (4.36)(4.38) disappear ! This "trick" has been used already before [9] without proper understanding of its cause and effect.

For the SRCF one inherently computes the same equations as the CKF but in a different way. Starting with errors δS_k and $\delta \hat{x}_{k|k-1}$ and assuming the updating equations to be performed exactly, we have that (4.34),(4.35),(4.37),(4.40) and (4.42) still hold, whereby

$$\delta P_{k|k-1} = S_k . \delta S'_k + \delta S_k . S'_k + O(\delta^2) \tag{4.46}$$

which is clearly symmetric by construction. Here also this ensures the convergence to zero of $\delta P_{k|k-1}$ and hence of δS_k, δK_k and $\delta \hat{x}_{k|k-1}$ if γ_k is sufficiently bounded in the time-varying case.

For the SRIF we start with errors δT_k and $\delta \hat{x}_{k|k}$ and use the equality:

$$\delta P_{k|k}^{-1} = T'_k . \delta T_k + \delta T'_k . T_k + O(\delta^2) \tag{4.47}$$

to relate this problem to the CKF as well. Since the SRIF update is exactly equal to an update of $P_{k|k}$ and $\hat{x}_{k|k}$ as in the CKF, the equations (4.38)(4.40) still hold where now the symmetry of $\delta P_{k|k}$ is ensured because of (4.47). From this it follows that $\delta P_{k|k}$ and $\delta \hat{x}_{k|k}$, and therefore also δT_k, converge to zero as k increases, provided γ_k is sufficiently bounded in the time-varying case.

Finally for the CSRF we start with errors δL_{k-1}, δG_{k-1}, $\delta R_{k-1}^{e1/2}$ and $\delta \hat{x}_{k|k-1}$. Because of these errors, (4.22) is perturbed *exactly* as follows:

$$\begin{pmatrix} R_{k-1}^{e1/2} + \delta R_{k-1}^{e1/2} & C(L_{k-1} + \delta L_{k-1}) \\ G_{k-1} + \delta G_{k-1} & A(L_{k-1} + \delta L_{k-1}) \end{pmatrix} . \overline{U}_3 = \begin{pmatrix} R_k^{e1/2} + \delta R_k^{e1/2} & 0 \\ G_k + \delta G_k & L_k + \delta L_k \end{pmatrix} \tag{4.48}$$

where $\overline{U}_3$ is also Σ_p-unitary. When $\lambda = ||C.L_{k-1}|| << ||R_{k-1}^{e1/2}||$ (which is satisfied when k is sufficiently large), Lemma C.3 yields after some manipulations:

$$\begin{pmatrix} \delta R_{k-1}^{e1/2} & C\delta L_{k-1} \\ \delta G_{k-1} & A\delta L_{k-1} \end{pmatrix} . U_3 = \begin{pmatrix} \delta R_k^{e1/2} & 0 \\ \delta G_k & \delta L_k \end{pmatrix} + O(\delta . \lambda) \tag{4.49}$$

Now the $(1,1)$ and $(1,2)$ blocks of U'_3 are easily checked to be given by $R_k^{e-1/2} . R_{k-1}^{e1/2}$ and $R_k^{e-1/2} . C . L_{k-1} . \Sigma$, respectively. From this, one then derives that for k sufficiently large

$$\begin{aligned} \delta R_k^{e1/2} &= \delta R_{k-1}^{e1/2} . [R_k^{e-1/2} . R_{k-1}^{e1/2}]' + \\ &\quad C . \delta L_{k-1} . [R_k^{e-1/2} . C . L_{k-1} . \Sigma]' + O(\delta . \lambda) \\ &= \delta R_{k-1}^{e1/2} . [R_k^{e-1/2} . R_{k-1}^{e1/2}]' + O(\delta . \lambda) \end{aligned} \tag{4.50}$$

$$\begin{aligned}\delta G_k &= \delta G_{k-1}.[R_k^{e-1/2}.R_{k-1}^{e1/2}]' + A.\delta L_{k-1}.[R_k^{e-1/2}.C.L_{k-1}.\Sigma]' + O(\delta.\lambda) \\ &= \delta G_{k-1}.[R_k^{e-1/2}.R_{k-1}^{e1/2}]' + O(\delta.\lambda)\end{aligned} \tag{4.51}$$

Here again thus the errors $\delta R_{k-1}^{e1/2}$ and δG_{k-1} are multiplied with the matrix $[R_k^{e-1/2}.R_{k-1}^{e1/2}]'$ at each step. When Σ is the identity matrix (i.e. when $inc.P_k$ is non-negative) this is a contraction since $R_k^e = R_{k-1}^e + C.L_{k-1}.L_{k-1}'.C'$. From this, we then derive similar formulas for the propagation of δK_k and $\delta\hat{x}_{k+1|k}$. Using lemma 4.2 for the perturbation of the inverse in $K_k = G_k.R_k^{e-1/2}$, we find:

$$\begin{aligned}\delta K_k &= \delta G_k.R_k^{e-1/2} - G_k.R_k^{e-1/2}.\delta R_k^{e-1/2}.R_k^{e-1/2} + O(\delta^2) \\ &= \delta G_k.R_k^{e-1/2} - K_k.\delta R_k^{e-1/2}.R_k^{e-1/2} + O(\delta^2)\end{aligned} \tag{4.52}$$

Using (4.50)(4.51) and the fact that for large k, $K_k = K_{k-1} + O(\lambda)$, we then obtain

$$\begin{aligned}\delta K_k &= \delta G_{k-1}.[R_k^{e-1/2}.R_{k-1}^{e1/2}]'.R_k^{e-1/2} \\ &\quad + K_{k-1}.\delta R_{k-1}^{e1/2}.[R_k^{e-1/2}.R_{k-1}^{e1/2}]'.R_k^{e-1/2} + O(\delta.\lambda) \\ &= \delta G_{k-1}.R_{k-1}^{e-1/2}.[R_{k-1}^e.R_k^{e-1}] \\ &\quad + K_{k-1}.\delta R_{k-1}^{e1/2}.R_{k-1}^{e-1/2}.[R_{k-1}^e.R_k^{e-1}] + O(\delta.\lambda)\end{aligned} \tag{4.53}$$

which because of (4.52) decremented by 1 becomes:

$$\delta K_k = \delta K_{k-1}.[R_{k-1}^e.R_k^{e-1}] + O(\delta.\lambda) \tag{4.54}$$

Using (4.24) we then also obtain from this:

$$\delta\hat{x}_{k+1|k} = F_k.\delta\hat{x}_{k|k-1} + \delta K_k.(y_k - C.\hat{x}_{k|k-1}) + O(\delta^2) \tag{4.55}$$

For the same reason as above, the matrix $[R_{k-1}^e.R_k^{e-1}]$ is a contraction when $\Sigma = I$, which guarantees the convergence to zero of δK_k and $\delta\hat{x}_{k+1|k}$. Notice however, that here the contraction becomes closer to the identity matrix as k increases, which suggests that the inherent decay of errors performed in previous steps will be smaller for this filter. Besides that, nothing is claimed about δL_k or $\delta P_{k+1|k}$, but apparently these are less important for this implementation of the KF since they do not directly affect the precision of the estimate $\hat{x}_{k+1|k}$. Moreover, when Σ is not the identity matrix, the above matrix has norm larger than 1 and divergence may be expected. This has also been observed experimentally as shown in chapter 5 and [99].

4.3.1.2. The numerical errors performed in one single step

We now turn our attention to the numerical errors performed in one single step k. Bounds for these errors are derived in the following theorem.

Theorem 4.1: Let us denote the *norms* of the absolute errors due to round-off during the construction of $P_{k+1|k}$, K_k, $\hat{x}_{k+1|k}$, S_k, T_k, $P_{k+1|k+1}^{-1}$ and $\hat{x}_{k|k}$ by Δ_p, Δ_k, Δ_x, Δ_s, Δ_t, Δ_{pinv} and Δ_x [2], respectively, we obtain the following upper bounds (where all norms are 2-norms):

1. CKF

$$\begin{aligned}\Delta_p &\leq \epsilon_1.\sigma_1^2/\sigma_p^2.\|P_{k+1|k}\| \\ \Delta_k &\leq \epsilon_2.\sigma_1^2/\sigma_p^2.\|K_k\| \\ \Delta_x &\leq \epsilon_3.(\|F_k\|.\|\hat{x}_{k|k-1}\| + \|K_k\|.\|y_k\| + \|D_k\|.\|u_k\|) + \\ &\quad \Delta_k.(\|C_k\|.\|\hat{x}_{k|k-1}\| + \|y_k\|)\end{aligned}$$

2. SRCF

$$\begin{aligned}\Delta_s &\leq \epsilon_4.(1+\sigma_1/\sigma_p).\|S_{k+1}\|/cos\phi_1 \\ \Delta_p &\leq \epsilon_5.(1+\sigma_1/\sigma_p).\|P_{k+1|k}\|/cos\phi_1 \\ \Delta_k &\leq \epsilon_6/\sigma_p.(\sigma_1/\sigma_p.\|S_{k+1}\| + \sigma_1.\|G_k\| + \|S_{k+1}\|/cos\phi_1) \\ \Delta_x &\leq \epsilon_7.(\|F_k\|.\|\hat{x}_{k|k-1}\| + \|K_k\|.\|y_k\| + \|D_k\|.\|u_k\|) + \\ &\quad \Delta_k.(\|C_k\|.\|\hat{x}_{k|k-1}\| + \|y_k\|)\end{aligned}$$

3. CSRF

$$\begin{aligned}\Delta_k &\leq \epsilon_8.\kappa(U_3)/\sigma_p.(\sigma_1/\sigma_p.\|L_k\| + \sigma_1.\|G_k\| + \|L_k\|/cos\phi_2) \\ \Delta_x &\leq \epsilon_9.(\|F_k\|.\|\hat{x}_{k|k-1}\| + \|K_k\|.\|y_k\| + \|D\|.\|u_k\|) + \\ &\quad \Delta_k.(\|C\|.\|\hat{x}_{k|k-1}\| + \|y_k\|)\end{aligned}$$

4. SRIF

$$\begin{aligned}\Delta_t &\leq \epsilon_{10}.\{\kappa(A_k) + \kappa(R_k^{1/2}) + \\ &\quad \tau_1/\tau_m.[\kappa(Q_k^{1/2}) + \kappa(A_k)]\}.\|T_{k+1}\|/cos\phi_3 \\ \Delta_{pinv} &\leq \epsilon_{11}.\{\kappa(A_k) + \kappa(R_k^{1/2}) + \\ &\quad \tau_1/\tau_m.[\kappa(Q_k^{1/2}) + \kappa(A_k)]\}.\|P_{k+1|k+1}^{-1}\|/cos\phi_3 \\ \Delta_p &\leq \Delta_{pinv}.\|P_{k+1|k+1}\|^2 \\ \Delta_x &\leq \epsilon_{12}.\|D_k\|.\|u_k\| + \\ &\quad \Delta_t.[\kappa^2(T_{k+1}).\|\nu_{k+1}\| + \kappa(T_{k+1}).\|\hat{x}_{k+1|k+1}\| + \|\nu_{k+1}\|/cos\phi_4]\end{aligned}$$

where σ_i and τ_i are the i-th singular value of $R_k^{e1/2}$ and $Q_{k+1}^{e-1/2}$ respectively, ϵ_i are constants close to the machine precision ϵ and $cos\phi_i$ are defined as follows

$$cos\phi_1 = \|S_{k+1}\|/\|\,[G_k|S_{k+1}]\,\|$$

[2] The same symbol Δ_x is used to indicate the error on the filtered state estimate and on the one-step predicted state estimate. However, the precise meaning follows clearly form the context

$$cos\phi_2 \;=\; \|L_k\|/\|\,[G_k|L_k]\,\|$$

$$cos\phi_3 \;=\; \|T_{k+1}\|/\|\begin{pmatrix} T_k A_k^{-1} \\ R_{k+1}^{-1/2} C_{k+1} \end{pmatrix}\|$$

$$cos\phi_4 \;=\; \|\nu_{k+1}\|/\|\begin{pmatrix} T_{k+1}\hat{x}_{k+1|k+1} \\ \nu_{k+1} \end{pmatrix}\|$$

and are usually close to 1.

Proof:

(1)CKF. Using Lemma C.1 the errors performed when constructing the matrix R_k^e can be bounded by $\epsilon_m.\|R_k^e\|$ and those for its inverse by $\epsilon_m.\kappa(R_k^e).\|R_k^e\|$. By again applying Lemma C.1 several times one finally obtains all bounds for Δ_p, Δ_k and Δ_x, as given above.

(2)SRCF. The bounds for Δ_k and Δ_s follow directly from Lemma C.2 since K_k and S_{k+1} are the least square solution and the residual, respectively, of the problem $[\ A\ |\ B\ ]$ where A' and B' are the top and bottom block rows of the prearray (4.2). The matrix A_1 of Lemma C.2 is here the matrix $R_k^{e1/2}$. The bound for Δ_p then follows directly from the bound for Δ_s using (4.46) and the fact that $\|S_{k+1}\|^2 = \|P_{k+1|k}\|$. Finally, the bound for Δ_x is obtained from that for Δ_k and from using Lemma C.1 several times.

(3)CSRF. For the case $\Sigma = I$, one obtains the bound for Δ_k as for the SRCF, from the observation that K_k is the least squares solution of the problem $[\ A\ |\ B\ ]$ where A' and B' are the top and bottom block rows of the prearray (4.22). The matrix A_1 of Lemma C.2 is here also the matrix $R_k^{e1/2}$. When $\Sigma \neq I$, this bound is multiplied by $\kappa(U_3)$ from the following observation. We can use Lemma C.1 to bound the errors in constructing the prearray (which we call M_k) by $\epsilon_m.\|M_k\|$, and those in constructing the postarray (which we call N_k) by

$$\epsilon_m.\|U_3\|.\|M_k\| \;=\; \epsilon_m.\|U_3\|.\|N_k.U_3^{-1}\| \;\leq\; \epsilon_m.\kappa(U_3).\|N_k\|$$

In terms of N_k we are now again in a problem of classical least squares and errors in M_k and N_k are related by a factor $\kappa(U_3)$, whence the bound for Δ_k for general Σ. The bound for Δ_x is then obtained by repeatedly using Lemma C.1 as for the SRCF.

(4)SRIF. As above, T_{k+1} is the residual of a least squares problem where A and B are the first and second block columns of the prearray (4.7).

The relative backward errors (δ_a and δ_b in the appendix) in these matrices A and B are, according to Lemma C.1, bounded by $\kappa(R_k^{1/2}) + \kappa(A_k)$ and $\kappa(Q_k^{1/2}) + \kappa(A_k)$, respectively. Using this and Lemma C.2 we then obtain the bound for Δ_t. The bound for Δ_{pinv} is then obtained from that for Δ_t using (4.47) and the fact that $\|T_{k+1}\|^2 = \|P_{k+1|k+1}^{-1}\|$. The bound for Δ_p is then on its turn obtained from that for Δ_p using Lemma 4.2. Finally, $\hat{x}_{k+1|k+1}$ is the least squares solution of the bottom 2×2 block in the post-array, which on itself is a residue (much as T_{k+1}) and is therefore only known with Δ_t precision. Using Lemma C.2 we thereby obtain the bounds for Δ_x.

Here again we should point out that all bounds hold for several norms when appropriately adapting the constants ϵ_i (see appendix C).

■

These bounds are crude simplifications of the complicated process of rounding errors in linear algebra, but are often a good indication of what can go wrong in these problems (see e.g. [88] and [94]).

Corollary 4.1: The above theorem also gives bounds on the errors due to model deviations δA_k, δB_k, δC_k, δD_k, δQ_k and δR_k as follows. Let η be the relative size of these errors, i.e. $\|\delta M\| \leq \eta\|M\|$ for M equal to each of the above model matrices, then the above bounds hold when replacing the ϵ_i by numbers η_i which are now all of the order of η.

Proof: The model errors can indeed be interpreted as backward errors on the matrices A_k, etc., but then on a machine of precision η. The same analysis then holds, but with ϵ replaced by η.

■

This corollary indicates that the effect of the modeling errors on the numerical errors performed in one single step, can be treated as round-off errors. However, the propagation of these kind of errors is different. This will be addressed more concretely in section 4.3.2.

Remark 4.1: It is interesting to note here that the bounds derived in Theorem 4.1 disprove in a certain sense a result that was often used to claim the numerical supremacy of the SRF's, namely that the sensitivity (i.e. the condition number) of $P_{k+1|k}$, K_k and $\hat{x}_{k+1|k}$ as computed by the SRF's is the square root of that of the same quantities computed via the CKF. Apparently, this can only be claimed for $P_{k+1|k}$ and *not* for K_k or $\hat{x}_{k+1|k}$, as follows from a quick comparison of the CKF and the SRF's in Theorem 4.1. This is also illustrated via simulations in chapter 5 and [99].

4.3.1.3. Accumulation and interaction of the round-off errors

Theorem 4.1 is now used together with the analysis of the *propagation* of errors through the recursion of the KF to yield bounds on the *total* error of the different filters at a given step k, which we denote by the prefix δ_{tot} instead of δ.

For this we first turn to the (symmetrized) CKF. For the total error $\delta_{tot}P_{k+1|k}$ we then have according to (4.34)(4.36)(4.37)(4.40) and Theorem 4.1 (for any *consistent* norm [92]):

$$\|\delta_{tot}P_{k+1|k}\| \leq \gamma_k^2.\|\delta_{tot}P_{k|k-1}\| + \overline{\Delta}_p \tag{4.56}$$

$$\|\delta_{tot}K_k\| \leq c_1.\gamma_k.\|\delta_{tot}P_{k|k-1}\| + \overline{\Delta}_k \tag{4.57}$$

$$\|\delta_{tot}\hat{x}_{k+1|k}\| \leq \gamma_k.\{\|\delta_{tot}\hat{x}_{k|k-1}\| + c_2.\|\delta_{tot}P_{k|k-1}\|\} + \overline{\Delta}_x \tag{4.58}$$

Here the upperbar on the Δ's indicate that these are not the exact bounds of Theorem 4.1 (which are derived under the assumption that the computations up to step k are exact), but analogous bounds derived for the perturbed results stored in computer at step k. Under the assumption that at step k the accumulated errors are still of the order of the errors estimated in Theorem 4.1, one easily finds that the Δ- and $\overline{\Delta}$-quantities are $O(\delta^2)$-close to each other. It is thus reasonable to assume that they are equal to each other. Denoting by Δ_{tot} the norm of the corresponding matrix δ_{tot}, then finally yields:

$$\begin{pmatrix} \Delta_{tot}P_{k+1|k} \\ \Delta_{tot}K_k \\ \Delta_{tot}\hat{x}_{k+1|k} \end{pmatrix} \leq \gamma_k. \begin{pmatrix} \gamma_k & 0 & 0 \\ c_1 & 0 & 0 \\ c_2 & 0 & 1 \end{pmatrix} . \begin{pmatrix} \Delta_{tot}P_{k|k-1} \\ \Delta_{tot}K_{k-1} \\ \Delta_{tot}\hat{x}_{k|k-1} \end{pmatrix} + \begin{pmatrix} \Delta_p \\ \Delta_k \\ \Delta_x \end{pmatrix} \tag{4.59}$$

where the inequality is meant elementwise. From this one then easily sees that the total errors will remain of the order of the local errors as long as the norms γ_k do not remain too large for a long period of time. For a time-invariant system, γ_k can be replaced by ρ_k — if the norm is chosen appropriately as discussed in (4.42) —, which then becomes eventually smaller than 1. Comparable results can also be stated about the γ_k if the time variations in the model are sufficiently smooth.

Using the above inequality recursively from 0 to ∞ one finally obtains

$$\begin{pmatrix} \Delta_{tot}P_\infty \\ \Delta_{tot}K_\infty \\ \Delta_{tot}\hat{x}_\infty \end{pmatrix} \leq \begin{pmatrix} 1/(1-\hat{\gamma}^2) & 0 & 0 \\ c_1\hat{\gamma}/(1-\hat{\gamma}^2) & 1 & 0 \\ c_2\hat{\gamma}/((1-\hat{\gamma}^2)(1-\hat{\gamma})) & 0 & 1/(1-\hat{\gamma}) \end{pmatrix} . \begin{pmatrix} \Delta_p \\ \Delta_k \\ \Delta_x \end{pmatrix} \tag{4.60}$$

if $\hat{\gamma} < 1$, where $\hat{\gamma}$ is the largest of the γ_k's. When γ_k tends to a fixed value γ_∞ it is easily shown that $\hat{\gamma}$ can be replaced by γ_∞ in (4.60), since the contributing terms to the summation are those with growing index k. For a *time-invariant* system, finally, this can then be replaced by ρ_∞ as was remarked earlier, and the condition $\hat{\gamma} = \rho_\infty < 1$ is then always satisfied.

For the SRCF, one uses the relation to the CKF (as far as the propagation of errors from one step to another is concerned) to derive (4.60) in an analogous fashion, but now with Δ_p, Δ_k and Δ_x appropriately adapted for the SRCF as in Theorem 4.1. Finally, for the SRIF one also obtains analogously the top and bottom inequalities of (4.60) for Δ_p and Δ_x adapted for the SRIF as in Theorem 4.1 and where now $\hat{\gamma}$ is the largest of the $\tilde{\gamma}_k$'s. Upon convergence, the same remarks hold as above for replacing $\hat{\gamma}$ by $\tilde{\gamma}_\infty$ and ρ_∞. Finally for the CSRF, we can only derive from (4.54)(4.55) a recursion of the type:

$$\begin{pmatrix} \Delta_{tot}K_k \\ \Delta_{tot}\hat{x}_{k+1|k} \end{pmatrix} \leq \begin{pmatrix} \beta_k & 0 \\ c_2 & \gamma_k \end{pmatrix} . \begin{pmatrix} \Delta_{tot}K_{k-1} \\ \Delta_{tot}\hat{x}_{k|k-1} \end{pmatrix} + \begin{pmatrix} \Delta_k \\ \Delta_x \end{pmatrix} \tag{4.61}$$

where $\beta_k = \|R_{k-1}^e.R_k^{e-1}\|_2$. Recursive summation of these inequalities as was done to obtain (4.60), only converges here — for both $\Delta_{tot}K_\infty$ and $\Delta_{tot}\hat{x}_\infty$ — when the β_k increase sufficiently slow to 1 as k grows. We remark here that these are only upper bounds, but the fact that they may diverge does indeed indicate that for the CSRF numerical problems are more likely to occur.

4.3.2. Modeling errors

In applying the KF to a specific system, the dynamical system model $\{A_k, B_k, C_k\}$, noise statistics $\{Q_k, R_k\}$ and a priori data $\{x_{0|-1}, P_0\}$ must be specified. Since the system model is usually an approximation of a physical situation, the model parameters and noise statistics are seldom exact. That is, the system model used in constructing the filter differs from the actual system that generates the observations. It is clear that an inexact filter model will *degrade* the filter performances.

In the literature several authors have considered this filter degradation and developed error models for particular modeling errors. Because this effect is very important in practical applications investigations were performed not long after Kalman's paper [51]. A.H. Jazwinski, gave a very nice overview of these different contributions. In his book [43] he combined

these results in one general theorem that formulates a single error model for the *propagation* of the different modeling errors. More recently, these results have been reconfirmed by B.D.O. Anderson in [1].

All these investigations are restricted to the study of the propagation of the different kind of modeling errors in the KF recursion. This was done without taking into account the effect of the numerical implementation chosen. More importantly, the interaction of the errors caused by the implementation used with the propagation of the modeling errors wasn't considered at all. This problem, however, very nicely fits into the framework developed in the previous subsection. This will be demonstrated for the error on the transition matrix A_k (for the other modeling errors which can be derived in a very similar way we refer to [43] and [1]) and only for the CKF. We will also restrict our derivations to the effect on the state error covariance matrix.

Let δA_k be the error on the system matrix in step k, then we denote the transition matrix of the model used in the KF as:

$$\overline{A}_k = A_k + \delta A_k \tag{4.62}$$

with $\|\delta A_k\| = O(\delta)$.
Since we are no longer working with the real system model, the calculated matrix $\overline{P}_{k+1|k}$, which is computed by:

$$\overline{P}_{k+1|k} = \overline{A}_k(\overline{P}_{k|k-1} - \overline{P}_{k|k-1}C_k'\overline{R}_k^{e-1}C_k\overline{P}_{k|k-1})\overline{A}_k' + B_kQ_kB_k' \tag{4.63}$$

is no longer an error covariance matrix.
The real state error covariance matrix, however satisfies a similar difference equation:

$$P_{k+1|k} = A_k(P_{k|k-1} - P_{k|k-1}C_k'R_k^{e-1}C_kP_{k|k-1})A_k' + B_kQ_kB_k' \tag{4.64}$$

The difference between equations (4.63) and (4.64), denoted by $\delta P_{k+1|k}$, can be computed in a similar way as outlined in section 4.3.1.1, where δ is taken as defined in (4.62).
If we assume that $\delta P_{k|k-1}$ remains symmetric, the following error propagation model results:

$$\delta P_{k+1|k} = F_k\delta P_{k|k-1}F_k' + \delta A_kP_{k|k-1}F_k' + F_kP_{k|k-1}\delta A_k' + O(\delta^2) \tag{4.65}$$

So, the error propagation behaves roughly as:

$$\|\delta P_{k+1|k}\| \approx \gamma_k^2\|\delta P_{k|k-1}\| + c.\gamma_k\|P_{k|k-1}\|.O(\delta) \tag{4.66}$$

with c a constant of the order of unity.
Model (4.66) again learns us that when $\delta P_{k|k-1}$ is symmetric the difference $\delta P_{k|k-1}$ decreases in time when no additional errors are performed.

Remark 4.2: From [43] and [1] it is observed that a similar stabilization of the errors remains valid for the errors on the other system model matrices, noise statistics and a priori data. This is not true if unknown zero bias errors occur on the input signal u_k of the state space model (3.1). Their presence can lead to severe deterioration of the reconstructed state variables. A general treatment of this kind of error source is given in the next chapter.

To formulate an upperbound for the errors made during each recursion by the KF implementation used, we can use corollary 4.1. If we restrict to the symmetrized CKF, we are now able in combination with model (4.66) to write down an upperbound for the total error due to δA_k:

$$\|\delta_{tot}P_{k+1|k}\| \leq \gamma_k^2\|\delta P_{k|k-1}\| + c.\gamma_k\|P_{k|k-1}\|.O(\delta) + \overline{\Delta}_p^{\eta} \tag{4.67}$$

where $\overline{\Delta}_p^{\eta}$ is $\overline{\Delta}_p$ used in theorem 4.1 (1) with ϵ_i replaced by the relative error on A_k, i.e. $\|\delta A_k\| \leq \eta\|A_k\|$.

4.3.3. Discretization errors

The discrete system model (3.1) often results from discretizing a set of differential equations:

$$\dot{x} = \mathcal{A}x + \mathcal{B}u \tag{4.68}$$

where we now restrict ourselves to the time invariant case.
This is performed to solve this set of equations on a digital computer. Here, one looks for an approximation of the solution to (4.68):

$$x(t) = e^{\mathcal{A}(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{\mathcal{A}(t-\tau)}\mathcal{B}u(\tau)d\tau \tag{4.69}$$

If we consider this solution only over one sample period Δt, where we denote the values of x at the time instant $i.\Delta t$ by x_i, (4.69) can also be written as:

$$x_{k+1} = e^{\mathcal{A}\Delta t}x_k + \int_{k\Delta t}^{(k+1)\Delta t} e^{\mathcal{A}([k+1]\Delta t-\tau)}\mathcal{B}u(\tau)d\tau \tag{4.70}$$

It is a common practice to approximate x_{k+1} in (4.70) by the state $\overline{x}_{k+1}$ of a discrete time system:

$$\overline{x}_{k+1} = e^{\mathcal{A}\Delta t}\overline{x}_k + \int_0^{\Delta t} e^{\mathcal{A}\eta}\, d\eta . B u_k \qquad (4.71)$$

This corresponds to the use of a zero-order hold with no delay for the input signal, depicted in fig. 4.1. This means that:

$$u(\tau) = u(k\Delta t) \quad \text{for} \quad k\Delta t \leq \tau \leq (k+1)\Delta t$$

Equation (4.71) can easily be verified using the above condition for the input signal, see e.g. [26].

This approximation, leaving alone the question of the accuracy by which the transition matrix $e^{\mathcal{A}\Delta t}$ and the input distribution matrix

$$\int_0^{\Delta t} e^{\mathcal{A}\eta}\, d\eta . B$$

can be calculated (see e.g. [114]), induces an error in the state calculations. The effect of this approximation on any computation involving the discretized system matrices is summarized under the name *discretization error.*

The design of a KF for continuous time systems with a digital computer is based on a discretized sytem model as given by equation (4.71). In practical applications we are interested in *the influence of the choice of the discretization period* on the performance of the KF. Therefore, we will compare the calculated KF quantities based on the discrete system model (4.71) for different choices of Δt, displayed by p in fig. 4.1. (The choice of an infinitely small discretization period — without taking into account the practical achievability to perform these computations — will correspond to the continuous time equivalents). To allow comparison of the computed KF results two different values of the discretization periods p have to be chosen that are a multiplicity of each other, say Δt and $2\Delta t$. For these two different cases, the system models are:

$$\{A, B, C\} \qquad \text{for a discretization period } \Delta t \qquad (4.72)$$

with $A = e^{\mathcal{A}\Delta t}$ and $B = \int_0^{\Delta t} e^{\mathcal{A}\eta}\, d\eta . \mathcal{B}$, and

$$\{A^2, (A+I)B, C\} \text{ for a discretization period } 2\Delta t \qquad (4.73)$$

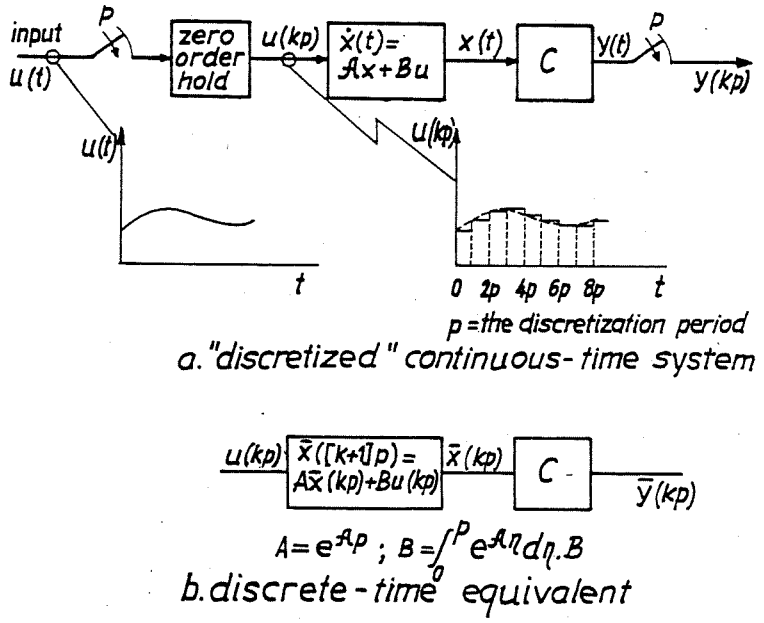


Figure 4.1: Representation of the discretization of a continuous system.

First, we consider the difference that results from the calculation of the state error covariance matrix after the first comparable recursion and using the same starting value, denoted as P_0. For two KF recursions with the system model (4.72) we obtain:

$$P_{2|1}^{\Delta t} = (A - K_1 C)(A - K_0 C) P_0 (A')^2 + (A - K_1 C) B Q B' A' + B Q B' \qquad (4.74)$$

One recursion with system model (4.73) gives:

$$P_{1|0}^{2\Delta t} = A(A - K_0 C) P_0 (A^2)' + (A + I) B Q B' (A + I)' \qquad (4.75)$$

So, the difference satisfies:

$$(P_{1|0}^{2\Delta t} - P_{2|1}^{\Delta t}) = A P_{1|0}^{\Delta t} C' (C P_{1|0}^{\Delta t} C' + R)^{-1} C P_{1|0}^{\Delta t} A' \qquad (4.76)$$

Equation (4.76) shows that the effect of using a larger discretization period can be interpreted as designing a filter with *more input noise.* This is because the right hand side of (4.76) is *non-negative definite.* For this case, the KF design is *more conservative.*

Although, the increase of the discretization period can be seen as an induction of extra input noise, the inherent stabilization of the Riccati difference equation, expressed by eq. (4.36) under the assumption that the state error covariance matrix will remain symmetric, assures that the KF recursion does not diverge by using a larger discretization period.

Remark 4.3: The effect of increased input noise by a larger discretization period will cause the calculated state error covariance matrix to remain always higher but bounded, with respect to any consistent norm, on comparable sampling intervals.

Secondly, we discuss the effect on the computation of the reconstructed state quantities. To analyze the discretization error on the reconstructed state variables, we assume that all the signals are deterministic, i.e. we will only consider the mean of the state quantities. Then equation (4.3) with the system matrices $\{A, D\}$ equal to $\{e^{\mathcal{A}p}, \int_0^p e^{\mathcal{A}\eta} d\eta . \mathcal{B}\}$, for a particular discretization period p, gives the *reconstructed state* quantity:

$$\hat{x}([k+1]p) = e^{\mathcal{A}p}\hat{x}(kp) + \int_{kp}^{[k+1]p} e^{\mathcal{A}([k+1]p-\eta)}\mathcal{B}u(kp)d\eta + K_k y_k - K_k C\hat{x}(kp) \qquad (4.77)$$

The *real state* obeys equation (4.70). Therefore, the error between the reconstructed state and the real state quantities becomes:

$$[x([k+1]p) - \hat{x}([k+1]p)] = (e^{\mathcal{A}p} - K_k C)[x(kp) - \hat{x}(kp)] + \int_{kp}^{[k+1]p} e^{\mathcal{A}([k+1]p-\eta)}\mathcal{B}[u(\eta) - u(kp)]d\eta \qquad (4.78)$$

Where the system matrix $(e^{\mathcal{A}p} - K_k C)$ is clearly the KF transition matrix (4.35) which tends to a matrix with spectral radius smaller than one [1]. Therefore, the reconstructed error due to the discretization will be *bounded.* Furthermore, a number of *qualitative results* can be given from this analysis:

- If a *larger* discretization period is used $\|K_k\|$ will normally increase, causing the KF transition matrix (4.35) to become more exponentially stable. That is moving its eigenvalues more inside the unit circle. Therefore, the reconstructed error due to discretization will *decrease faster.* On the other hand, (4.78) learns that a larger discretization period causes the second term to increase, depending on the nature of the input signal. These two contradicting facts will determine the magnitude of the *deterministic error* on the reconstructed state variables.
- From the experiments performed at the end of section 2.4.3 we have learned that a more stable "observer" transition matrix increases the sensitivity of the reconstructed state variables for measurement errors. This is explicitly derived in equation (2.76) that indicates that the uncertainty of the reconstructed state variables increases with a larger discretization period.

The practical value of these observations will be verified by experimental analysis in the following chapter.

4.4. New efficient time-invariant KF implementations

The use of the condensed forms discussed in chapter 2 in the different KF implementations leads to a considerable speed-up when the number of inputs m and/or outputs p is significantly smaller than the number of state variables n. Since this happens mostly in practice we will tacitly assume to hold when comparing various implementations. The computations that are required in CKF, equations (4.13-4.18) are mainly matrix multiplications. This is also true for the construction of the pre-array in the different SRF's. However beside these multiplications the SRF also involve unitary or Σ-unitary transformations. In the execution of the matrix multiplication it is easily seen that a lot of profit may be gained from the use of condensed forms. For the conventional KF this leads to a reduction of the most time consuming operation, namely the product $A.P_{k|k-1}.A'$, roughly by a factor of 2 if the (lower or upper) Schur form is used. About the same improvement is obtained when using the other condensed forms since A is essentially triangular in each of these cases. These operation counts for the CKF are summarized in Table 4.8.

The reduction of the number of computations by the use of condensed forms is now outlined for the SRCF. First, we consider the *lower* Schur form,

where the number of operations required to construct the pre-array (4.2) is given in table 4.5 (we only mentioned the differences with table 4.1). If we choose additionally that the factors $R^{1/2}$, $Q^{1/2}$ and S_k are lower triangular, the pre-array has the following special structure (illustrated again for the system dimensions n=6, m=3, p=2):

$$\left(\begin{array}{cc|cccccc|ccc} \mathbf{x} & 0 & x_{11} & x_9 & x_7 & x_5 & x_3 & x_1 & 0 & 0 & 0 \\ x & \mathbf{x} & x_{12} & x_{10} & x_8 & x_6 & x_4 & x_2 & 0 & 0 & 0 \\ \hline 0 & 0 & \mathbf{x} & 0 & 0 & 0 & 0 & 0 & x_{13} & x_{13} & x_{13} \\ 0 & 0 & x & \mathbf{x} & 0 & 0 & 0 & 0 & x_{14} & x_{14} & x_{14} \\ 0 & 0 & x & x & \mathbf{x} & 0 & 0 & 0 & x_{15} & x_{15} & x_{15} \\ 0 & 0 & x & x & x & \mathbf{x} & 0 & 0 & x_{16} & x_{16} & x_{16} \\ 0 & 0 & x & x & x & x & \mathbf{x} & 0 & x_{17} & x_{17} & x_{17} \\ 0 & 0 & x & x & x & x & x & \mathbf{x} & x_{18} & x_{18} & x_{18} \end{array}\right) \tag{4.79}$$

from which we see that many elements that need to be annihilated are already zero. This particular structure of the pre-array is exploited to derive an efficient triangularization, which is briefly outlined for the example. The annihilation of the elements x_1 to x_{12} of the (1,2) block in the pre-array (4.79) is done with *Givens transformations* in the order denoted by the subscripts and using the $\mathbf{x}$'s as pivots. In this way the triangular shape of the (2,2) block is preserved. At the same time, this is a very nice illustration of remark 2.2 discussed in chapter 2. Further elimination of all elements x_{13} to x_{18} is then done by using *Householder reflections*, again using the $\mathbf{x}$'s in the corresponding rows as pivots. This can be done efficiently using remark 2.3. The total cost of these transformations is given in table 4.5.

symbol	Mathematical expression	number of flops
M_{23}	$BQ^{1/2}$	–
$\begin{pmatrix} M_{12} \\ M_{22} \end{pmatrix}$	$\begin{pmatrix} CS_k \\ AS_k \end{pmatrix}$	$n^3/6+n^2p/2$
	annihilation M_{12} using Givens	$2np^2+2n^2p$
	further reduction using Householder	n^2m

Total number of flops: $\frac{1}{6}n^3 + n^2(\frac{5p}{2} + m) + n(2p^2)$

Table 4.5: Lower Schur implementation of the SRCF

Similar arguments also lead to an efficient implementation of the SRIF algorithm when using an *upper* Schur form for the model and upper triangular Choleski factors for T_k, $R^{-1/2}$ and $Q^{-1/2}$. Assuming A^{-1} (which is also upper triangular), $A^{-1}B$ and $R^{-1/2}C$ to be computed once and for all, then the work for constructing the prearray (4.7) is given in table 4.6 (where we only noted the differences with table 4.2). The triangularization is dual to that of the SRCF, since both arrays look like each others transpose, the pattern of zeros included. The operation counts are thus similar to table 4.5 (when replacing p by m) and are summarized in table 4.6.

symbol	Mathematical expression	number of flops
M_{32}	$R^{-1/2}C$	–
$(\ M_{21} \mid M_{22}\ )$	$(\ T_kA^{-1}B \mid T_kA^{-1}\ )$	$n^3/6+n^2m/2$
	annihilation M_{21} using Givens	$2nm^2+2n^2m$
	further reduction using Householder	n^2p

Total number of flops: $\frac{1}{6}n^3 + n^2(p + \frac{5m}{2}) + n(2m^2)$

Table 4.6: Upper Schur implementation of the SRIF

A further improvement can now be obtained for the SRCF using a *lower* observer-Hessenberg form for the model since now Householder transformations can be used throughout instead of (slower) Givens transformations. The pattern of zeros after a permutation (i.e. a unitary transformation) of the second and third block columns of the pre-array (4.7) now indeed looks like (for the same choice of dimensions as above):

$$\left(\begin{array}{cc|ccc|ccccc} \mathbf{x} & 0 & 0 & 0 & 0 & x_1 & 0 & 0 & 0 & 0 & 0 \\ x & \mathbf{x} & 0 & 0 & 0 & x_2 & x_2 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & \mathbf{x} & x_3 & x_3 & x_3 & x_3 & x_3 & 0 & 0 & 0 \\ 0 & 0 & x & \mathbf{x} & x_4 & x_4 & x_4 & x_4 & x_4 & 0 & 0 \\ 0 & 0 & x & x & \mathbf{x} & x_5 & x_5 & x_5 & x_5 & x_5 & 0 \\ 0 & 0 & x & x & x & \mathbf{x} & x_6 & x_6 & x_6 & x_6 & x_6 \\ 0 & 0 & x & x & x & x & \mathbf{x} & x_7 & x_7 & x_7 & x_7 \\ 0 & 0 & x & x & x & x & x & \mathbf{x} & x_8 & x_8 & x_8 \end{array}\right) \tag{4.80}$$

and all the elements x_1 up to x_8 can be eliminated using Householder reflections only (using the $\mathbf{x}$'s as pivots). The total computational cost for

this is given in table 4.7.

symbol	Mathematical expression	number of flops
M_{23}	$BQ^{1/2}$	–
$\begin{pmatrix} M_{13} \\ M_{23} \end{pmatrix}$	$\begin{pmatrix} CS_k \\ AS_k \end{pmatrix}$	$n^3/6+pn^2/2$
	further reduction using Householder	$2np^2+n^2(m+p)$ $+2/3p^3$

Total number of flops: $\frac{1}{6}n^3 + n^2(\frac{3p}{2} + m) + n(2p^2) + \frac{2p^3}{3}$

Table 4.7: Lower observer-Hessenberg implementation of the SRCF

Similar significant savings with the controller-Hessenberg form can *not* be obtained for the SRIF, for the reason that A^{-1} and $A^{-1}B$ are now full matrices despite the sparsity of A and B. The same holds for the CSRF because the L_k matrices are usually full.

filter	type	complexity
CKF	full	$(3/2)n^3 + n^2(3p + m/2) + n(3p^2/2 + m^2) + p^3/6$
	Schur	$(3/4)n^3 + n^2(5p/2 + m/2) + n(3p^2/2 + m^2) + p^3/6$
	o-Hess.	$(3/4)n^3 + n^2(7p/2 + m/2) + n(2p^2 + m^2) + p^3/6$
SRCF	full	$(7/6)n^3 + n^2(5p/2 + m) + n(p^2 + m^2/2)$
	Schur	$(1/6)n^3 + n^2(5p/2 + m) + n(2p^2)$
	o-Hess.	$(1/6)n^3 + n^2(3p/2 + m) + n(2p^2) + 2p^3/3$
SRIF	full	$(7/6)n^3 + n^2(p + 7m/2) + n(p^2/2 + m^2)$
	Schur	$(1/6)n^3 + n^2(p + 5m/2) + n(2m^2)$
	c-Hess.	essentially no reduction
CSRF	full	$(n_1 + n_2)(n^2 + 3np + p^2)$
	Schur	essentially no reduction
	o-Hess.	essentially no reduction

Table 4.8: Operation counts for the different KF's

A complete list of the total operations counts for the different methods is given in table 4.8 for comparison. Here we have not included second order terms which e.g. come from the calculation of the estimates $\hat{x}_{k+1|k}$ or $\hat{x}_{k+1|k+1}$ in these different methods.

4.5. Comparison of the different filters

In this section we compare the different filter implementations based on the error analysis of section 2.3 and the complexity analysis of section 2.4.

We first look at the time-varying case (whence excluding the CSRF). According to the error bounds of Theorem 4.1 for round-off errors and modeling errors, it appears that the SRCF has the lowest estimate for the *local* errors generated in a single step k. The *accumulated* errors during subsequent steps is governed by the norms γ_k for all three filters in a similar fashion (at least for the error on the estimate). This of course under the assumption that the "symmetrized" version of the CKF is considered, which does not require much more computational work. For practical problems (see e.g. [9], [99]) these three filters indeed seem to yield a comparable accuracy for the estimates $\hat{x}_{k+1|k}$ or $\hat{x}_{k+1|k+1}$, unless some of the "influential" parameters in the error bounds of Theorem 4.1 become critical. This is e.g. true for the SRIF which will give worse results when choosing matrices A_k, R_k or Q_k that are hard to invert. As far as R_k or Q_k are concerned, this is in a sense an artificial disadvantage since in some situations the inverses R_k^{-1} and Q_k^{-1} are the given data and the matrices R_k and Q_k have then to be computed. This then would of course disadvantage the SRCF (see [81] for by-passing this). The problem of inverting A_k, on the other hand, is always present.

For the computational cost, the SRCF again seems to have a marginal advantage over the two other methods when n is significantly larger then m and p, which is a reasonable assumption in general. As is shown in the next chapter or [73], [99], condensed forms or even the CSRF can sometimes be used in the time-varying case as well, when e.g. only some of the matrices are time-varying or when the variations are structured. In that case the latter two may yield significant savings in computing time.

For the time-invariant case, the same comments as above hold for the accuracy of the CKF, SRCF and SRIF. The fourth candidate, the CSRF, has in general a much poorer accuracy than the other three. This is now not due to some pathologically chosen parameters, but to the simple fact that the accumulation of rounding errors from one step to another is usually much more significant than for the three other filters, as was pointed out in section 4.3. This is particularly the case when the signature matrix Σ is not the identity matrix, which may then lead to divergence as also shown experimentally in the next chapter or [99].

As for the complexity, the (lower) observer-Hessenberg form of the SRCF seems to be the most appealing candidate, except when the coefficient n_1+n_2 for the CSRF is much smaller than n. This is e.g. the case when the initial covariance $P_{0|-1}$ is zero, in which case the CSRF becomes the fastest of all four filters. Although the Schur implementations of the SRCF and SRIF are almost as fast as the observer-Hessenberg implementation of the SRCF, they also have the small additional disadvantage that the original state-space transformation U for condensing the model to Schur form is more expensive than that for the other condensed forms and that the (real) Schur form is not always exactly triangular but may contain some 2×2 "bumps" on the diagonal (corresponding to complex eigenvalues of the real matrix A).

As a general conclusion, we recommend the SRCF, and its observer-Hessenberg implementation in the time-invariant case, as the optimal choice of KF implementation because of its good balance of reliability and efficiency. Other choices may of course be preferable in some specific cases because of special conditions that would then be satisfied.

4.6. Concluding remarks

In this chapter we have analyzed four different KF algorithms and some new variants for their reliability and computational efficiency. The comparison is based on an error analysis and an operation count where we have made full use of possible savings in the time-invariant case.

From the error models a better insight is also gained of which parameters influence the error propagation in the different KF algorithms that have been investigated. For the CKF and the SRCF these are:

- the condition number of the innovation signal covariance matrix R_k^e
- the spectral norm (radius) of the filter state transition matrix F_k

For the CSRF the choice of:

- the initial error covariance matrix $P_{0|-1}$ matrix
- the condition number of the innovation signal covariance matrix R_k^e

becomes critical and, for the SRIF the relevant parameters in numerical reliability are:

- the condition number of the innovation signal covariance matrix R_k^e
- the spectral norm (radius) of the filter state transition matrix F_k
- the condition number of the process noise covariance matrix Q_k and of the matrix Q_k^e
- the condition number of the system state transition matrix A_k
- the condition number of the Choleski factor of the state covariance matrix

This is moreover supported by the simulation study on flight-path reconstruction given in chapter 5 or [99].

The obtained results in this and previous chapters have now paved the way to start the development of algorithms which solve the two subproblems of the two-step identification procedure, described in section 1.4, *in real-time*. On the other hand, these two subproblems will be used to illustrate the validity of the obtained (theoretical) results.

Chapter 5.

The Flight-Path Reconstruction Module

5.1. Introduction

In this chapter we analyze the first step of the two-step identification procedure, outlined in the first chapter. In this step the *aircraft's flight-path is reconstructed.* As outlined in chapter 1, we are interested in the *accurate* determination of these quantities in *real-time.* And this for mainly two reasons:

- Firstly, the accuracy of the reconstructed flight-path mainly determines the *reliability* with which the characteristic aircraft properties and performances can be estimated in the following step of the two-step procedure.
- Secondly, the flight-path is one of the *essential information sources* in the operation of the FMS, described in chapter 1. It allows the FMS to check whether a prescribed "optimal" flight-path is actually followed by the airplane. For this case the proposed adaptive flight-path reconstruction technique is an alternative to the combination of Inertial Navigation Systems (INS) and Radio Navigation Systems (RNS) [102]. The advantage of the proposed technique is that it is designed to operate independently, and needs no correction from other navigation sources as is the case with the conventional combination of INS and RNS.

The *on-line realizability* of the new flight-path reconstruction technique

proposed in this chapter relies on 3 different factors:

- *Accurate modeling* of the aircraft system dynamics plus the error sources of the observed signals that are used to reconstruct the aircraft's flight path. This is first achieved by the use of the *inertial signals.* Hence, the entries of the system matrices of the dynamic system model used to reconstruct the aircraft's flight-path, given by (1.18-1.19), are *known very accurately.* Secondly, the lack of knowledge about the precise nature of the errors on the inertial signals can make the flight-path reconstruction results completely useless after a very short time. Therefore, these error sources require special care. This is handled in section 5.3.2.
- The use of *accurate inertial measurements.* These measurements are supplied by accelerometers and rate gyro's. A lot of effort was spent by sensor technologists to improve measurement equipment on next generation aircrafts. Helped by the fast development in micro-electronics and physics (laser technology) accurate inertial reference platforms were designed. This will become a *standard feature* on future commercial aircrafts, such as the Boeing B757 and B767 and the Airbus A310.
- The use of robust and numerically stable implementations of the chosen algorithm to reconstruct the aircraft's flight-path.

The mathematical system model (1.18-1.19) ideally fits into the framework of Kalman filtering. Furthermore, the set of *recursive* equations describing the filter are well suited for implementation on a digital computer. However, depending on the numerical implementation chosen, the calculated results may be very sensitive to *round-off errors* [52], [9] and [100]. To investigate whether the given problem suffers from this undesired phenomenon, the results of the theoretical error analysis of the error propagation due to round-off, may be used instead of a very time consuming simulation study. The practical value of these theoretical results will be verified *experimentally* for the basic flight-path reconstruction problem defined in section 1.4.1. This is done in section 5.2 for the four different KF implementations considered in the theoretical error analysis. In this section, also the results concerning the other error sources are verified. Finally, this then allows to choose the most reliable KF implementation for the problem at hand.

The real-time operation of the KF algorithm requires further adaptation of the filter to take into account:

- the variations of the aircraft's dynamics during the different phases of flight (*climb, cruise and descent*) including variations in the errors of the different sensors
- the *automatic* determination of its convergence
- the efficiency of the developed algorithm

These three topics are analyzed in section 5.3. The analysis of the first item results in a *complete adaptation* scheme that supervises the original KF. The theoretical error model is used to solve the second item. For the last item a new square root covariance filter implementation is developed for the flight-path reconstruction problem that solves this problem far more *efficiently* than the conventional KF does. Section 5.4 describes the developed flight-path reconstruction algorithm. Some concluding remarks are given in section 5.5.

5.2. Experimental evaluation of the different KF's

The objective of this section is to evaluate the reliability and the efficiency of different KF implementations. In the literature, different types of KF implementations have been compared using a selection of experimental conditions for some particular (realistic) problems [9], [52]. More general arguments for comparison are based on concepts of numerical analysis (such as conditioning) which have e.g. been used to indicate that from a numerical point of view the class of SRF's is usually superior to the CKF or heuristically derived variants of it [9]. While original implementations of SRF's were slower than the CKF, this was not true anymore for later versions of the SRF's. Some of these apply to the general time-varying case (such as e.g. the $[UDU']$-implementation of the SRCF [8]) while others only hold for the time-invariant case (such as e.g. the Chandrasekhar formulation [74]). Despite these advantages of the SRF's, the CKF is still often used for different reasons: the simplicity of the CKF formulation allows one to easily implement it on a computer with readily available software. More importantly, for several experimental conditions it will perform reasonably well as far as accuracy is concerned, and for some conditions a better performance may even be observed.

In this section we show by a series of experiments that the upper bounds

for numerical round-off developed in the previous chapter and [100] are in general reasonably close to the true error build up. These error estimates can thus be used to decide *in advance*, prior to any experimental analysis, if a filter implementation is suited for the problem under consideration.

5.2.1. Experimental set-up

The objective formulated above is demonstrated using the basic flight-path reconstruction problem, defined in section 1.4.1. In the simulation, the system model (A.30) and the observation model (A.31), derived in appendix A, are taken for a discretization period and sampling period $\Delta t = 0.05$ *sec.* Four different KF implementations are included in the experimental analysis, namely the conventional KF (CKF), the square root covariance filter (SRCF), the square root information filter (SRIF) and the Chandrasekhar square root filter (CSRF). The relevant parameters that influence the error propagation in the four different KF under consideration, as indicated by the theoretical error analysis are:

1. The choice of an initial condition for the error covariance matrix, denoted by $P_{0|-1}$.
2. The condition number of the innovation signal covariance matrix, i.e. $\kappa(R_k^e)$. In the simulations, it turns out that the initial choice of $\kappa(R_k)$ approximately determines $\kappa(R_k^e)$ during the whole run. This is partly due to the fact that $||P_{k|k-1}||$ is small compared to $||R_k||$.
3. The spectral norm γ_k and radius ρ_k of the filter state transition matrix F_k. This can be affected by "weighting" the system matrix A_k by a factor c_w.
4. The condition number of the process noise covariance matrix $\kappa(Q_k)$
5. The condition number of the system state transition matrix $\kappa(A_k)$. This is affected by the choice of a state space coordinate system.
6. The condition number of the square root of the error covariance matrix, i.e. $\kappa(S_k)$. This is hard to control a priori.

To study round-off errors, computations were carried out in double precision as well as in single precision and *double precision* results are considered to be *exact*. The round-off errors on three different quantities that result from a KF were considered in the simulations, namely:

1. on the state error covariance matrix P, denoted by $\Delta_{tot}P_{k|k-1} = ||P_{k|k-1} - \overline{P}_{k|k-1}|| = ||\delta_{tot}P_{k|k-1}||$
2. on the Kalman gain K, denoted by $\Delta_{tot}K_k = ||K_k - \overline{K}_k|| = ||\delta_{tot}K_k||$
3. on the reconstructed state quantities $\hat{x}_{k|k-1}$ or $\hat{x}_{k|k}$, denoted by $\Delta_{tot}x_k = ||\hat{x}_{k|k-1} - \overline{\hat{x}}_{k|k-1}||$ or $||\hat{x}_{k|k} - \overline{\hat{x}}_{k|k}||$, depending on the chosen algorithm (this will become clear from the context).

Here (.) and $\overline{(.)}$ denote the exact (i.e. double precision), respectively approximated (i.e. single precision) results and ||.|| is the Frobenius norm. It is noted that the SRIF does not require the Kalman gain explicitly to compute the filtered state quantities. Therefore the second parameter will not be considered for this implementation.

Since the accuracy of the first two quantities determines the accuracy of the reconstructed state, a first analysis can be *restricted* to these quantities. If conditions can be formulated under which accuracy degradation of these two quantities occurs, extensive simulation tests with the input and output time histories of the real (or simulated) system become obsolete.

Because of the inclusion of the CSRF, only the *time-invariant* case will be considered in this first analysis. The SRCF and the SRIF algorithms are closely related from numerical point of view. They are therefore first compared to the CKF and secondly to the CSRF.

5.2.2. Comparing the SRCF/SRIF with the CKF

The experimental conditions of the different tests are listed in table 5.1. From the theoretical analysis [100] it follows that the relevant parameters that influence the reliability of the CKF are $\kappa(R_k^e)$ and $\rho(F_k)$, the spectral radius of F_k. Two test were performed to analyze their effect. The magnitudes of the variables $\kappa(R^e)$ and $\rho(F)$ are indeed very close to the values of $\kappa(R)$ and $\rho(A)$ given in table 5.1. The results of these tests are graphed in figure 5.1. From this figure the following observations are made:

1. Test 1 - Fig. 5.1a: ($\rho(A) = 1.0$ and $\kappa(R) = 10^2$)

Since symmetry of the error state covariance matrix P is not preserved by the CKF, the round-off error propagation model for the local error $\delta P_{k|k-1}$ is given by (4.36), i.e.:

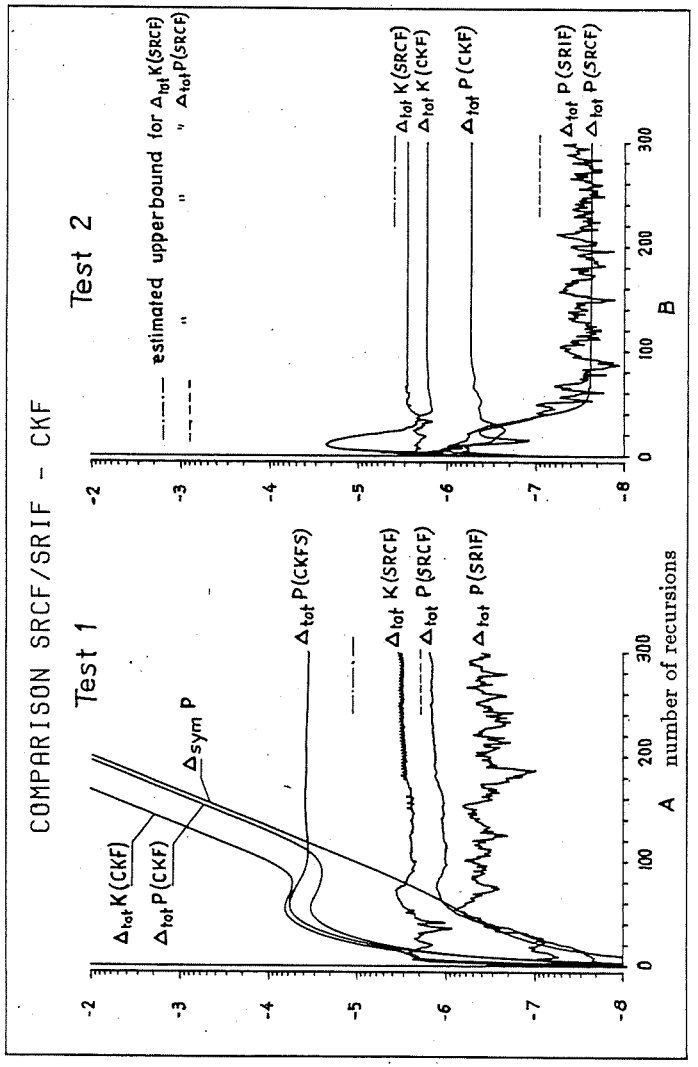


Figure 5.1: Comparison of the numerical performance of the SRCF/SRIF with the CKF.

$$\delta P_{k+1|k} \approx (A_k - K_k C_k)\delta P_{k|k-1}(A_k' - C_k' K_k') + A_k(\delta P_{k|k-1} - \delta P'_{k|k-1})C_k' K_k' \quad (5.1)$$

with K_k defined as $A_k P_{k|k-1} C_k'(R_k^e)^{-1}$. This tells us that divergence of round-off errors on P and hence on K will occur if the original system is *unstable*. This experiment confirms this divergence phenomenon also when $\rho(A) = 1.0$, as is the case for the flight-path reconstruction problem (see (1.18)) . Furthermore, it is observed from fig. 5.1a that the error on P with the CKF is almost completely determined by the *loss of symmetry*, computed by $||\overline{P}_{k|k-1} - \overline{P}'_{k|k-1}|| = \Delta_{sym} P_{k|k-1}$.
A very simple trick is usually used to solve this problem. Hereby, the error covariance matrix P is forced to become symmetric after each filter recursion by averaging the off-diagonal elements of P. This implementation ensures that the second term of the right hand side (rhs) of (5.1) vanishes. As indicated by the behaviour of $\Delta_{tot} P_{k|k-1}$ in fig.5.1a, this modified CKF implementation, denoted by CKF(S) becomes again competitive for application, even when the original system is unstable. This is obtained with a very modest increase in computational complexity. On the other hand, the round-off error on P with the CKF(S) remains higher (a factor 10) than the SRCF/SRIF combination, which are both of the same order.

2. Test 2 - Fig. 5.1b: ($\rho(A) = 0.9$ and $\kappa(R) = 10^2$)

If we make the original system stable, the CKF is numerically stable. Moreover, the accuracy with which the Kalman gain is computed is of the same order as that of the SRCF. This is in contrast with a general opinion that SRF possesses a conditioning or sensitivity that is the square root of that of the CKF, [52], [9]. From fig. 5.1b it is observed that this is *only so for* the error covariance matrix P, which confirms the upperbounds for the round-off errors as given in section 4.3.1.3. These upperbounds are:

For the error covariance matrix:

$$\Delta_{tot} P_{k+1|k} \leq \gamma_k^2 . \Delta_{tot} P_{k|k-1} + \Delta_p \quad (5.2)$$

with for the CKF: $\Delta_p \leq \epsilon_1 . \kappa(R_k^e) . ||P_{k|k-1}||$ (a)

and for the SRCF: $\Delta_p \leq \epsilon_5 . (1 + \kappa(R_k^e)^{1/2}) . ||P_{k+1|k}|| / cos\phi_1$ (b)

For the Kalman gain:

$$\Delta_{tot} K_k \leq c . \gamma_k . \Delta_{tot} P_{k|k-1} + \Delta_k \quad (5.3)$$

with for the CKF: $\Delta_k \leq \epsilon_2 . \kappa(R_k^e) . ||K_k||$ (a)

and for the SRCF: $\Delta_k \leq \epsilon_6/\sigma_p . (\frac{\sigma_1}{\sigma_p} . ||S_{k+1}|| + \sigma_1 . ||G_k|| + ||S_{k+1}||/cos\phi_1)$ (b)

Here $\kappa(R_k^e)^{1/2} = \frac{\sigma_1}{\sigma_p}$ and γ_k is the spectral norm of F_k (which can be replaced in (5.2) and (5.3) by the spectral radius ρ_k if convergence is achieved). The Kalman gain K_k and G_k are related as follows:

$$K_k = G_k . (R_k^e)^{-1/2} \quad (5.4)$$

This first comparison leads to the general observation that when the accuracy of the Kalman gain is considered no preference exists for the SRF with respect to the CKF, if the CKF is changed to the CKF(S) implementation for situations where the original system has eigenvalues *on or outside* the unit circle. However, the experimental results demonstrate that for the latter conditions the loss of accuracy with the CKF(S) is higher than the SRF's.

5.2.3. Comparison SRCF/SRIF with CSRF

The upperbound for the round-off errors of the Kalman gain and the state estimate $\hat{x}_{k+1|k}$ computed by the CSRF (for large k) are given by (4.61) or:

$$\Delta_{tot} K_k \leq \beta_k . \Delta_{tot} K_{k-1} + \Delta_k \quad (5.5)$$

with $\Delta_k \leq \epsilon_8 . \kappa(U_2)/\sigma_p (\frac{\sigma_1}{\sigma_p} ||L_k|| + \sigma_1 ||G_k|| + ||L_k||/cos\phi_2)$

$$\Delta_{tot} \hat{x}_{k+1|k} \leq c . \Delta_{tot} K_{k-1} + \gamma_k \Delta_{tot} \hat{x}_{k|k-1} + \Delta_x \quad (5.6)$$

with $\Delta_x \leq \epsilon_9 (||F_k|| . ||\hat{x}_{k|k-1}|| + ||K_k|| . ||y_k|| + ||D|| . ||u_k||) + \Delta_k . (||C|| . ||\hat{x}_{k|k-1}|| + ||y_k||)$

This model indicates that the error propagation is convergent when $\beta_k = ||R_{k-1}^e . (R_k^e)^{-1}|| < 1$, which is the case only if the signature matrix Σ is the identity matrix I. Note that the error variation $\Delta_{tot} K_k$ is now weighted by β_k (instead of γ_k for the other filters) which even for $\Sigma = I$ becomes

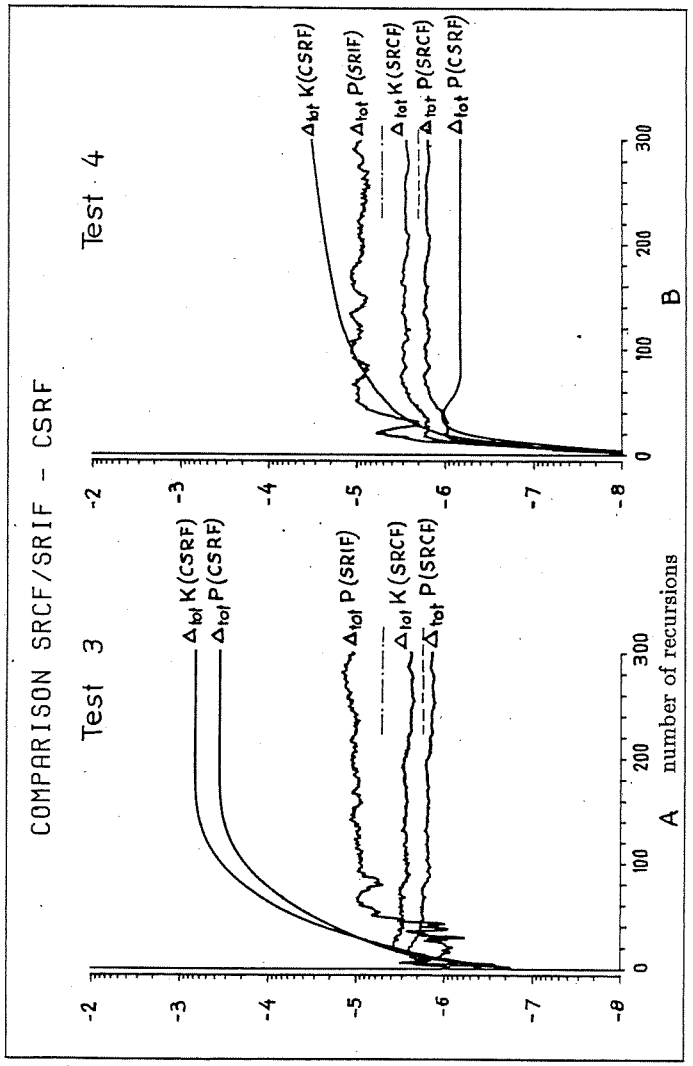


Figure 5.2: Comparison of the numerical performance of the SRCF/SRIF with the CSRF.

very close to 1 unlike γ_k. This is also the main reason for the poor numerical behaviour of this filter. When $\Sigma \neq I$, which depends on the choice of $P_{0|-1} \neq 0$, β_k is larger than 1 and $\kappa(U_2)$ *may* also become large. Both these phenomena have a negative influence on the above bounds and may eventually cause divergence. Apart from the ***numerical sensitivity*** introduced by the choice of $P_{0|-1}$, it also ***influences*** the ***efficiency*** of the CSRF implementation. This is indicated by the parameter $\alpha = (n_1 + n_2)$, i.e. the rank of $incP_0 = P_{1|0} - P_{0|-1}$, in the operation count of the CSRF, given in table 4.4:

$$\alpha.(n^2 + p^2 + 3np) + n^2 + 2np + \frac{p^2}{2} + nr + \alpha n \qquad (5.7)$$

where the system dimensions are defined in section 4.2. and α lies in the interval $[n, min(m, p)]$.
The influence of the choice of $P_{0|-1}$ is analyzed by the following two tests.

1. Test 3 - Fig. 5.2a: ($P_{0|-1} \neq 0$, $\rho(A) = 1.0$ and $\kappa(R) = 1.0$)

The choice of $P_{0|-1} \neq 0$ influences the CSRF implementation ***negatively.*** First, in this experiment the computational efficiency decreases compared to the case $P_{0|-1} = 0$, discussed in the following test (see also table 5.3). This is because α in (5.7) becomes greater than p or m. This was the case for all the tests performed with $P_{0|-1} \neq 0$. Secondly, the transformations used in each recursion to triangularize the pre-array are Σ-unitary, what causes them to have a condition number > 1. This is due to the fact that $incP_0$ is not definite. From fig. 5.2a, this negative effect is clearly observed. Both the error levels on P and K are larger by a factor 10^2 compared to the SRCF or SRIF. For the covariance type algorithms considered here, it is observed that the error on the Kalman gain is always higher than the error on the state error covariance matrix. This is partly due to the extra calculation needed for the Kalman gain by (5.4). Here the condition number of $(R_k^e)^{1/2}$ determines the loss of accuracy.

2. Test 4 - Fig. 5.2b: ($P_{0|-1} = 0$, $\rho(A) = 1.0$ and $\kappa(R) = 1.0$)

For this case $incP_0 = B.Q.B'$ is positive definite, causing the transformations used in each recursion to be ***unitary.*** On the other hand $\alpha = m$ in (5.7) makes the CSRF more efficient compared to the new SRCF/SRIF implementation based on the "condensed" system representations.
From the experimental results in fig. 5.2b we observe that the error on P is very small, while the error on K is much higher compared to the round-off

level for the SRCF calculations. Furthermore, the errors on K with the CSRF *increase very slowly* because the coefficient β_k becomes very close to 1. This is due to the fact that for the CSRF round-off errors are carried along on three matrices, namely G_k, $(R_k^e)^{1/2}$ and L_k, while for the SRCF/SRIF combination errors are carried along only on the Choleski factors of P or P^{-1}. For the error on L_k (supposing $incP_k$ factored as $L_k L_k'$) this effect does not cause the errors on P_k:

$$P_k = \sum_{i=0}^{k} L_i L_i' \tag{5.8}$$

to accumulate because L_k converges rapidly enough to zero such that the accumulated errors on P_k:

$$\Delta_{tot} P_k = \sum_{i=0}^{k} (L_i . \Delta_{tot} L_i' + \Delta_{tot} L_i . L_i') \tag{5.9}$$

also converge if the $\Delta_{tot} L_i$ are not too large. The absolute value of the total error on $(R_k^e)^{1/2}$ and G_k remain much higher. This is clearly reflected in the loss of accuracy in the calculation of K_k by (5.4).

Generally, the CSRF is *less reliable* than the SRCF/SRIF combination. For zero initial conditions of the state error covariance matrix maximal reliability can be achieved with the CSRF. Therefore, for situations where $n >> m$, the CSRF may be preferred because of its increased *computational efficiency* and despite its *loss of accuracy*. We stress the fact that this property is only valid for the *time-invariant* case. Modifications of the CSRF exist taking into account certain time-varying effects [74], e.g. for the process noise covariance matrix Q. This however induces again an increased computational complexity.

5.2.4. Comparison of the SRCF and the SRIF

In the previous experiments the SRCF/SRIF combination performed equally well. In this section a further analysis is made to compare both implementations. Using the error model that indicates the upperbound for the round-off errors made during one SRIF recursion, given also in section 4.3.1.3:

$$\Delta_{tot} P_{k+1|k} \leq \gamma_k^2 . \Delta_{tot} P_{k|k-1} + \Delta_p \tag{5.10}$$

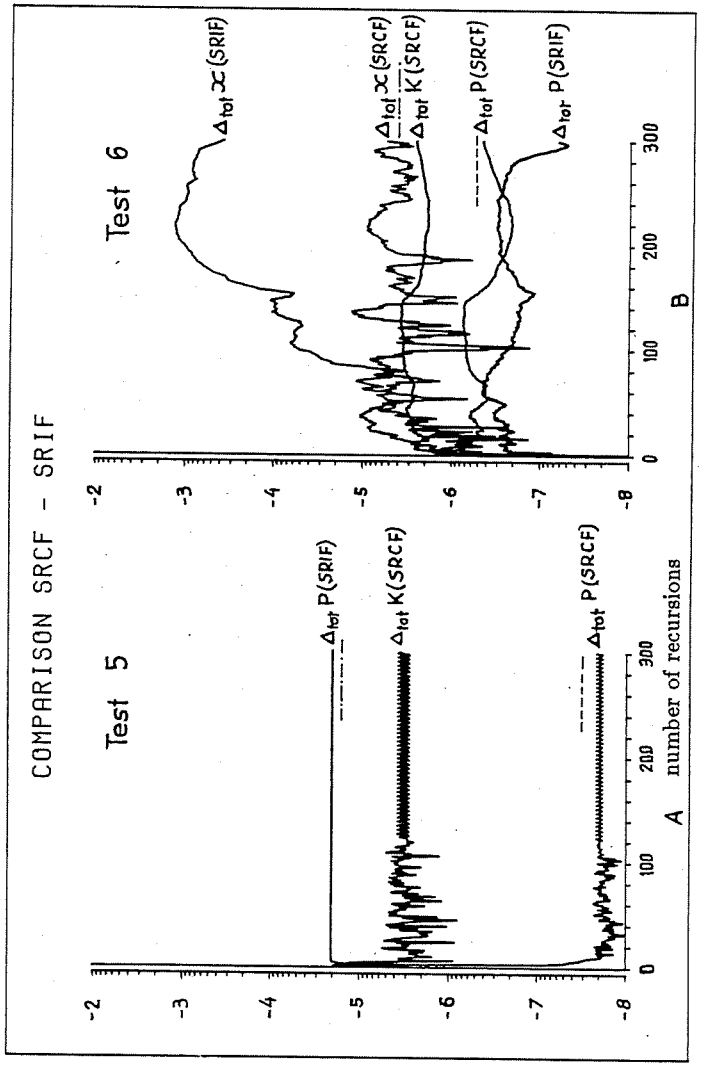


Figure 5.3: Comparison of the numerical performance of the SRCF with the SRIF.

$$\text{with:} \quad \Delta_p \leq \epsilon_{11}\{\kappa(A_k) + \kappa(R_k^{1/2}) + \frac{\tau_1}{\tau_m}[\kappa(Q_k^{1/2}) + \kappa(A_k)]\} \\ ||P_{k+1}^{-1}||.||P_{k+1}^{2}/cos\phi_3$$

$$\Delta_{tot}\hat{x}_{k+1|k+1} \leq \gamma_k^2.\Delta_{tot}\hat{x}_{k|k-1} + \Delta_x \tag{5.11}$$

$$\text{with:} \quad \Delta_x \leq \epsilon_{12}||D_k||.||u_k|| + \Delta_p[\kappa^2(T_{k+1}).||\nu_{k+1}|| + \\ \kappa(T_{k+1}).||\hat{x}_{k+1|k+1} + ||\nu_{k+1}||/cos\phi_4]$$

where $\kappa(Q_k^e)^{1/2} = \frac{\tau_1}{\tau_m}$ and ν_{k+1} is the innovation at time instant $k+1$, learns that besides $\kappa(R_k)$ and $\rho(F_k)$, other relevant factors influence the round-off error accumulation in the SRIF. The effect of these parameters is analyzed in the following tests.

1. Test 5 - Fig. 5.3a:

In this test very large condition numbers for A, Q and R (see table 5.1), are considered. As expected this indeed causes the error on P to be much higher (a factor 10^3) for the SRIF than for the SRCF. The large value of $\kappa(R)$ also causes a great loss in the accuracy of the Kalman gain calculation. The level of round-off errors on K indeed becomes a factor 10^2 larger than the round-off level of P, which is partly due to the fact that K_k is much larger and that $\kappa(R_k^e)^{1/2}$ occurs *quadratically* in the error bound for Δ_k (5.3b).

In this test we analyzed the deterioration of the error covariance matrix by the SRIF implementation by (fairly unrealistic) large condition numbers. However, our true interest should go to the accuracy of the reconstructed state. This will be considered in the next experiment.

2. Test 6 - fig. 5.3b:

For this test, the measurement error statistics were taken from real flight test measurement calibrations [41]. This results in the following forms for the process noise covariance matrix Q, respectively the measurement noise covariance matrix R:

$$Q = \begin{pmatrix} 8.10^{-6} & 0 & 0 \\ 0 & 5.10^{-5} & 0 \\ 0 & 0 & 5.10^{-8} \end{pmatrix} ; R = \begin{pmatrix} 5.10^{-2} & 0 \\ 0 & 2.10^{-1} \end{pmatrix} \tag{5.12}$$

The relevant parameters for the round-off error propagation are listed in table 5.1.

quantity	Test1	Test2	Test3
$\kappa(A)$	$1.457.10^1$	$1.457.10^1$	$1.457.10^1$
$\kappa(Q)$	1.0	1.0	1.0
$\kappa(R)$	$9.902.10^1$	$9.902.10^1$	1.0
$\kappa((R^e)^{1/2})$	7.525	8.057	1.021
c_w	1.0	0.9	1.0
$\rho(F_\infty)$	$9.803.10^{-1}$	$8.999.10^{-1}$	$9.803.10^{-1}$
$P_{0\|-1}$	$\neq 0$	$\neq 0$	$\neq 0$
$\|\|P_\infty\|\|$	$2.059.10^{-2}$	$1.973.10^{-3}$	$6.956.10^{-2}$
$\|\|K_\infty\|\|$	$4.366.10^{-1}$	$2.500.10^{-1}$	$1.287.10^{-1}$
$\|\|S_\infty\|\|$	$1.561.10^{-1}$	$4.909.10^{-2}$	$2.981.10^{-1}$
quantity	**Test 4**	**Test 5**	**Test 6**
$\kappa(A)$	$1.457.10^1$	$2.361.10^6$	$1.457.10^1$
$\kappa(Q)$	1.0	$1.389.10^6$	$9.999.10^2$
$\kappa(R)$	1.0	$1.002.10^3$	3.4
$\kappa((R^e)^{1/2})$	1.021	$1.347.10^1$	1.671
c_w	1.0	1.0	1.0
$\rho(F_\infty)$	$9.803.10^{-1}$	$9.000.10^{-1}$	$9.947.10^{-1}$
$P_{0\|-1}$	$= 0$	$\neq 0$	$\neq 0$
$\|\|P_\infty\|\|$	$6.956.10^{-2}$	$1.977.10^{-3}$	$3.357.10^{-3}$
$\|\|K_\infty\|\|$	$1.287.10^{-1}$	1.210	$1.810.10^{-2}$
$\|\|S_\infty\|\|$	$2.981.10^{-1}$	$4.493.10^{-2}$	$6.087.10^{-2}$

Table 5.1: Test conditions to evaluate the different KF implementations

In fig. 5.3b the simulated error on the state calculations, i.e. $\Delta_{tot}x_k$, for both filter implementations is plotted. Here, the error level with the SRIF is up to a factor 10^2 higher than that for the SRCF, while the two other quantities $P_{k|k-1}$, and K_k from the KF are computed with roughly equal accuracy. This is due to the high condition number of T_k in the calculation of the filtered state with the SRIF by the following equation, which corresponds to (4.8):

$$\hat{x}_{k|k} = T_k^{-1}.(\hat{s}_{k|k}) + D_{k-1}.u_{k-1} \tag{5.13}$$

5.2.5. The effect of modeling and discretization errors

According to the analysis given in section 4.3.2, the effect of modeling

errors is completely analogous to the effect of rounding errors. As a matter of fact, we came to the conclusion in the previous chapter that the effect of modeling errors could be demonstrated by working on a computer with machine precision equal to the magnitude of the modeling errors. Therefore, the experimental results concerning modeling errors will be very similar to those presented in the previous subsection.

However, to demonstrate the results related to the discretization error, given in section 4.3.3, we will perform a separate experiment. In this experiment the aircraft's flight-path is reconstructed with a SRCF for *two different discretization periods*, i.e. $\Delta t_1 = 0.05$ sec and $\Delta t_2 = 0.1$ sec. The *state reconstruction error*, i.e. the difference between the reconstructed state variable and their corresponding true (or simulated) state variable, and the *estimated standard deviation* ($\hat{\sigma}$), given by the square root of the diagonal element of the error state covariance matrix, are plotted in fig. 5.4a for Δt_1 and fig. 5.4b for Δt_2. Some important numerical values are given in table 5.2.

quantity	Δt_1	Δt_2
$\rho(F_\infty)$	$9.947.10^{-1}$	$9.895.10^{-1}$
$\|\|P_\infty\|\|$	$3.385.10^{-3}$	$5.824.10^{-3}$
$\|\|K_\infty\|\|$	$1.965.10^{-2}$	$3.926.10^{-2}$
$\|\|S_\infty\|\|$	$6.223.10^{-2}$	$8.833.10^{-2}$

Table 5.2: Different KF quantities for two different $\Delta t's$.

From these experimental results we can make the following observations:

1. The (Frobenius) norm of the error covariance matrix *increases* with larger Δt, see table 5.2. Also the *estimated standard deviations* of the errors on the reconstructed flight-path quantities *increases*, as follows from a comparison between the indicated $2-\sigma$ values in fig. 5.4a-b. This confirms clearly that the design of a KF becomes *more conservative* by increasing Δt, as is indicated by (4.76).
2. The state reconstruction errors remain *bounded* for a higher Δt, see fig. 5.4b.
3. For a larger Δt, $||K_\infty||$ increases and at the same time this causes the eigenvalues of $||F_\infty||$ to move more inside the unit circle. This is demonstrated by the smaller value of the spectral radius $\rho(F_\infty)$ for Δt_2 than for Δt_1, see table 5.2. However, the difference is very small. Therefore, the second term of (4.78) indicates that the *deterministic error* will increase due to the variation in the input

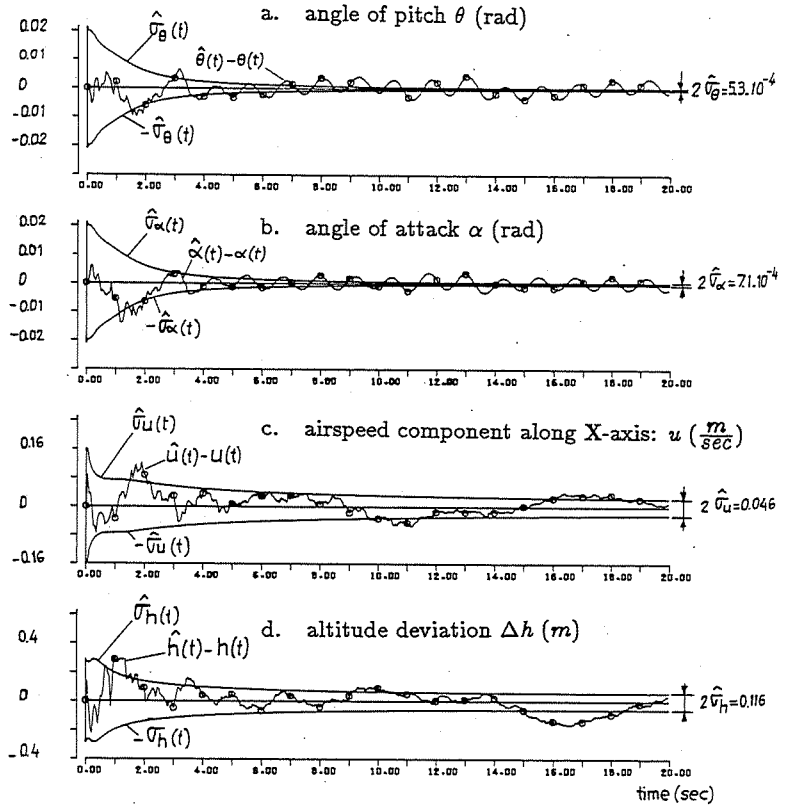


Figure 5.4a: Errors on the reconstructed flight-path quantities and estimated standard deviation bounds for $\Delta t_1 = 0.05$ sec (the magnitude of the flight-path quantities may be observed from fig. 5.8).

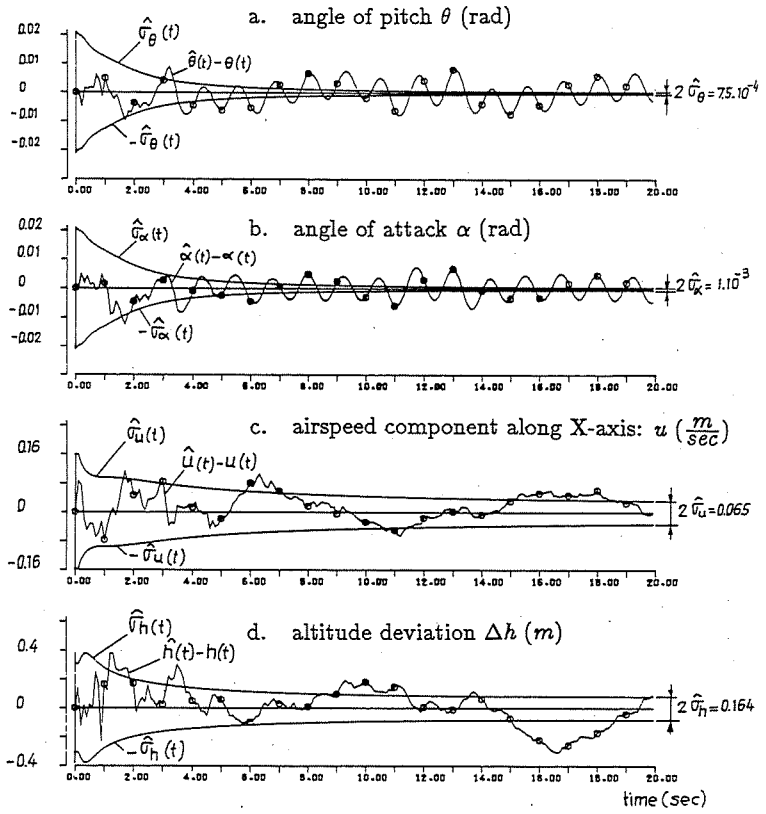


Figure 5.4b: Errors on the reconstructed flight-path quantities and estimated standard deviation bounds for $\Delta t_2 = 0.1$ sec.

signal for the larger discretization period. This can be clearly observed by comparing figures 5.4a and 5.4b. Furthermore, we see that for both Δt's the deterministic part is larger than the bounds specified by the $\hat{\sigma}$-values for all state reconstruction errors. This is because the latter only takes into account the *stochastic nature* of the reconstruction errors. Suppose that we subtract the deterministic components from the state reconstruction errors, it will become clear that these $\hat{\sigma}$-bounds accurately specify the statistics of the *stochastic inaccuracies* on the state reconstruction errors.

4. Due to the small change in $\rho(F_\infty)$ for the different Δt's, the filter estimates become indeed *slightly more sensitive* to the measurement errors. This may be observed from inspecting the random errors on the different state reconstruction errors for both Δt's. But is also indicated by the higher $\hat{\sigma}$-bounds. However, the deterministic error caused by the discretization error due to the variation in the input signal dominates this effect.

These experimental results clearly demonstrate the *practical value* of the qualitative observations made in the analysis of section 4.3.3.

5.2.6. Computational complexity

The operation counts derived in Table 4.8 for the different KF implementations demonstrate the reduction in computational complexity of the CKF, the SRCF and the SRIF. For $n >> m$ and p, they indicate that the condensed SRCF becomes the *most efficient* implementation, for the flight-path reconstruction problem (for $P_{0|-1} \neq 0$). When the actual system dimensions (n=4, m=3 and p=2) are used, the real computation times on an IBM 370 computer are listed in table 5.3, for 300 recursions. Because the dimensions are very close to each other, the savings of the SRCF are small. From this table we also observe the large difference in CPU time for the CSRF between the cases $P_{0|-1} = 0$ and $\neq 0$.
On the other hand, for the dimensions of the extended state space description (as discussed in section 5.3.2) the savings in computation time of the SRCF compared to the CKF and the SRIF becomes larger.

KF implementation	CPU (ms)	
	n = 4	n = 6
CKF	1341	2393
SRCF	1205	1862
SRIF	1244	2120
CSRF ($P_{0\|-1} \neq 0$)	1330	2276
CSRF ($P_{0\|-1} = 0$)	1188	1808

Table 5.3: Real computation time for the different KF implementations on a IBM 370 computer.

This experimental evaluation indicates that its combination of speed and accuracy makes the "condensed" SRCF *the most appealing KF implementation* to solve the basic flight-path reconstruction problem.

5.3. The adaptive flight-path reconstruction module

The reliability of the KF implementation is a necessary condition to assure real-time operation. Other very important conditions that will be analyzed in this chapter are time variations of the system model (1.18-1.19), zero bias errors on the inertial signals and efficiency. We also develop a procedure to control the convergence of the recursive KF algorithm which can be used to switch off the recursive estimation procedure when convergence has been obtained in a fixed reference flight condition (which then allows to use the computer unit for other purposes). These different items are treated below in reversed order.

5.3.1. Real-time convergence control of the SRCF

During periods where no system variations occur, e.g. during different phases in cruise, climb and descent, more efficient use of the computer unit is made if the Kalman gain updating can be stopped when "sufficient" accuracy has been obtained for the actual circumstances. Therefore, it is necessary to control the convergence by taking into account these actual system conditions. This can be achieved by formulating a *stop criterion* for the recursively computed quantities of the KF, given as:

$$\mathbf{i} \leq \mathbf{b} \tag{5.14}$$

where **i** is an indication of the convergence behaviour
b is a bound that specifies the accuracy to be achieved

This criterion can be specified for the recursions of the Kalman gain or the state error covariance matrix. Which of both leads to the most efficient and reliable procedure is investigated in the specification of **i** and **b**.

Calculation of **b**

For the actual system conditions an upperbound for the round-off errors during the SRCF recursion is given by (5.2b) for the state error covariance matrix and by (5.3b) for the Kalman gain. Among these two bounds, (5.2b) is the simplest to evaluate. The quantities that have to be computed in this equation are:

1. The Frobenius norm of $P_{k+1|k}$. This is done by $\frac{n^2}{4}$ multiplications and one square root.
2. The condition number, $\kappa(R_k^e)^{1/2}$. Since $(R_k^e)^{1/2}$ is a *triangular* matrix, reliable and efficient condition estimations are available, see e.g. [16].
3. The spectral radius $\rho(F_k)$, which can be estimated recursively by the *power method* [123].
4. $cos\phi_1$. All performed experiments indicate that this parameter is close to one. Therefore $cos\phi_1$ is taken equal to one.

The practical use of this procedure is evaluated for the tests discussed in section 5.2. Therefore, the stationary values specified by (5.2b) and (5.3b) for the SRCF, i.e.:

$$\Delta_{tot} P_\infty \leq \frac{\Delta_p}{1-\rho_\infty^2} \tag{5.15}$$

$$\Delta_{tot} K_\infty \leq \frac{c\rho_\infty}{1-\rho_\infty^2}.\Delta_p + \Delta_k \tag{5.16}$$

are plotted in the corresponding figures 5.1 to 5.3 (indicated by the dotted lines).

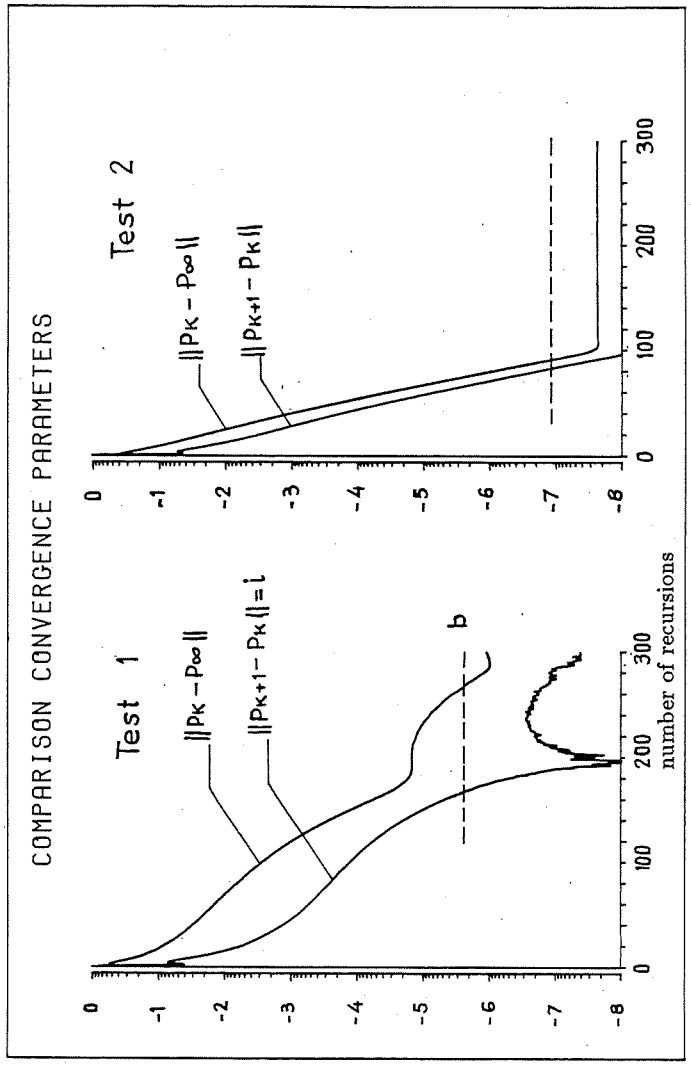


Figure 5.5: Analysis of different convergence parameters for the SRCF.

From this, it is easily seen that the bound for the round-off error on $P_{k|k-1}$ can be specified more accurately and with less work than that for K_k. Therefore, the error criterion will be formulated on the recursion for $P_{k|k-1}$.

Calculation of **i**

The indicator, **i** is the norm of the difference between the recursively computed solution $P_{k|k-1}$ and the infinite solution of the Riccati equation, i.e. P_∞. This is denoted as follows:

$$||\overline{P}_{k|k-1} - P_\infty|| \tag{5.17}$$

Since it is assumed that P_∞ is not available in real-time, this quantity is approximated by the magnitude of the deviation taken at each step k:

$$||\overline{P}_{k+1|k} - \overline{P}_{k|k-1}|| \tag{5.18}$$

which is now available at each step.
The admissibility of this approximation is evaluated for the tests indicated by numbers 1 and 2 of section 5.2. Both quantities given by equations (5.17) and (5.18) are computed. Here, P_∞ is calculated with the eigenvalue-eigenvector method described in [119]. The results are plotted in fig. 5.5. From this figure we clearly observe the similarity between the real convergence indicator and the approximated one. This indicates that for practical purposes (5.18) may be used as a "good" indicator of convergence.

In summary the efficient and reliable evaluation of both quantities **i** and **b** in (5.14) taking into account the time-varying system conditions, allows us to control the convergence of the recursive SRCF *on-line*.

5.3.2. Dealing with time-varying zero bias errors

In practical applications where inertial signals are integrated (such as in INS) or filtered by a KF it is well known that the filter estimates are only useful over a restricted period of time, depending on the accuracy of the inertial measurement system used. One of the main reasons is the presence of accelerometer zero bias errors and gyro drift, the latter of which leads to the most severe deterioration of the filter estimates. To reduce the estimation errors caused by these parameters, it is common practice in

covariance type of Kalman filtering to extend the state of the original system with the bias terms. This is in contrast with the automatic treatment of these terms in the SRIF where there is no need for extending the state vector [8]. This however only works if the zero bias terms are *observable* in the *extended state space formulation*, as will be demonstrated in the following example.

Example 5.1: For the following dynamic single input single output system:

$$x_{k+1} = 0.5x_k + (w_k + b_0) \quad \text{with } b_0 = 2.0 \tag{5.19}$$

$$y_k = x_k + v_k \tag{5.20}$$

we design a SRIF according to (3.67) and (3.68) to estimate the zero bias b_0 and the state quantity x_k. The covariance matrices of the random noise sequences w_k and v_k are not relevant for this example and therefore not indicated explicitly. From the post-array in (3.67) the estimate of the zero bias, denoted by $\hat{b}_0$, is easily computed from $\overline{w}_k$. This estimate is plotted in fig. 5.6. Since the zero bias error b_0 is *observable* in the extended state space description, given as:

$$\begin{pmatrix} b_{k+1} \\ x_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 0.5 \end{pmatrix} \begin{pmatrix} b_k \\ x_k \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} w_k \tag{5.21}$$

$$y_k = (0\ 1) \begin{pmatrix} b_k \\ x_k \end{pmatrix} + v_k \tag{5.22}$$

it is now possible to estimate b_0, as is demonstrated in fig. 5.6. This observability condition is indicated by the full rank of the observability matrix $\mathcal{O}$, defined in (1.20):

$$\mathcal{O} = \begin{pmatrix} 0 & 1 & . & . & . \\ 1 & 0.5 & . & . & . \end{pmatrix} \tag{5.23}$$

Since the zero bias errors on the inertial signals are not *all* observable from the extended state space description, the method used in this example is not applicable to the flight-path reconstruction problem. Therefore, we now turn our attention to the extended filtering problem and only treat the SRCF implementation.

Until now, a lot of effort has been put in the development of efficient computational techniques [27] to solve this extended problem. These techniques are based on the generalized partitioning estimation method of

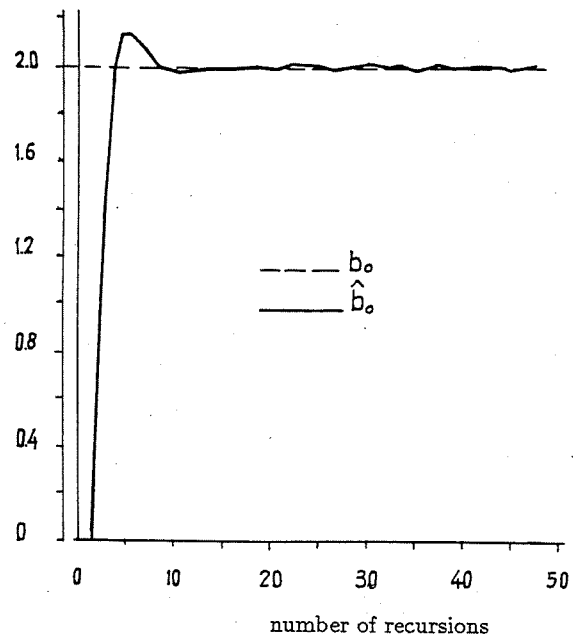


Figure 5.6: Reconstructed and real zero bias error on the input signal with the SRIF, using the non-extended state space formulation.

Lainiotis [62], to separate the estimation of the state variables from the bias terms. The influence of including bias terms in the model, on the *numerical behaviour* of the discrete Riccati equation has not been discussed yet. But simulation studies have already shown that it may lead to severe deterioration of the estimation results, especially in real-time applications.

5.3.2.1. The extended state space description

The discrete system model for the basic flight-path reconstruction problem is given by (1.18-1.19), for the time-invariant case. In this model the input signal u_k is contaminated by a zero-mean uncorrelated white noise

sequence w_k and a bias b_k. The output measurements y_k are only contaminated by a zero-mean uncorrelated white noise sequence v_k. Therefore, (1.18-1.19) becomes:

$$x_{k+1} = A_T.x_k + D_T.u_k + B_T.(w_k + b_k) \tag{5.24}$$

$$y_k = C_T.x_k + v_k \tag{5.25}$$

The index k of the system matrices in (1.18) has been dropped in this model because we now consider a fixed reference condition. Here the deterministic input distribution matrix D_T is equal to the process noise input distribution matrix B_T and we assume the following system dimensions.

$$x_k \in R^n \ , \ u_k, w_k, b_k \in R^m \ , \ y_k, v_k \in R^p$$

If the bias errors b_k are assumed to be constant, they can easily be modeled by the following state equations:

$$b_{k+1} = b_k \tag{5.26}$$

and be included in an extended state vector x^e_{k+1}. This gives rise to the following model:

$$\begin{pmatrix} b \\ x \end{pmatrix}_{k+1} = \begin{pmatrix} I_m & 0 \\ B_T & A_T \end{pmatrix} . \begin{pmatrix} b \\ x \end{pmatrix}_k + \begin{pmatrix} 0 \\ B_T \end{pmatrix} .w_k \tag{5.27}$$

$$y_k = (0 \quad C_T) . \begin{pmatrix} b \\ x \end{pmatrix}_k + v_k \tag{5.28}$$

or in compact matrix notation,

$$x^e_{k+1} = A^e.x^e_k + B^e.w_k \tag{5.29}$$

$$y_k = C^e.x^e_k + v_k \tag{5.30}$$

where I_m is the identity matrix of order m and the deterministic input signal is left out in the interest of clarity.

At this point, it should be remarked that prior to solving the filtering problem it should be verified that the extended state space model only

includes those bias errors that are *observable*. The number of observable bias errors is $nb \leq m$. From the extended state description (5.27 - 5.28), the following conclusions can be drawn directly:

1. the inclusion of the bias terms introduces a set of nb *eigenvalues* on the *unit circle*
2. the state matrices (A^e, B^e) clearly have nb *uncontrollable* modes corresponding to the bias parameters

We now investigate the implications of these two observations on the KF performance and formulate efficient and reliable implementations for the SRCF recursion.

5.3.2.2. The algorithm to make the bias terms controllable

For the above estimation problem of zero bias errors we have observed experimentally [104] that the combination of eigenvalues on the unit circle of the system model and the uncontrollability of the corresponding modes causes a serious deterioration of the performance of the KF. It makes the recursions very sensitive to numerical round-off, model errors and discretization errors and slows down the rate of convergence considerably. The method is therefore not useful anymore for real-time applications.

A general solution to these convergence problems is obtained by making the extended state space model completely controllable. This can be achieved by perturbing the zero bias errors by a white noise driving term, formulated as follows:

$$b_{k+1} = b_k + w^\star_k \tag{5.31}$$

But the inclusion of (5.31) in the extended state space formulation would increase the input system dimension m, thereby decreasing the filter efficiency.

For $nb \leq m$ it is possible to include the perturbation $w^\star_k$ into the present process noise w_k as follows:

$$b_{k+1} = b_k + \Delta X.w_k \tag{5.32}$$

and look for a procedure to calculate the submatrix ΔX of B^e in order to make the bias terms controllable. This perturbation has to be as small as possible, to obtain a *minimal deterioration* of the filter estimates, and on

the other hand large enough to make the bias terms *maximally controllable*. For the latter, we use the measure of controllability of a controllable pair $\{A, B\}$, i.e. $\mu(A, B)$, defined in [80]:

$$\mu(A,B) = min[max(||\Delta A||_2, ||\Delta B||_2)] \text{ such that the system defined by } \{A+\Delta A, B+\Delta B\} \text{ is uncontrollable}$$

which thus denotes the smallest perturbation needed to make the $\{A, B\}$ pair uncontrollable. Now $\mu(A^e, B^e + \Delta X) \leq ||\Delta X||$ for any perturbation ΔX since $\{A^e, B^e\}$ is uncontrollable. A good choice ΔX_0 is thus obtained when,

$$\mu(A^e, B^e + \Delta X_0) \approx ||\Delta X_0|| \tag{5.33}$$

since then no other perturbation ΔX of the same norm will make it as uncontrollable as ΔX_0. A recommended construction of ΔX_0 for embedding the system in a controllable one is now derived.

Step 1: Calculate a basis Y of all left eigenvectors of the system matrix A^e in the model (5.29 - 5.30), with eigenvalue 1. This can be done by a SVD (Singular Value Decomposition) of the matrix $(A^e - I)$, given as:

$$(A^e - I) = (\ U \mid Y\ ).\left(\frac{\Sigma \mid 0}{0 \mid 0}\right).V' \tag{5.34}$$

Step 2: Calculate the perturbation ΔX_0 of the input distribution matrix B^e as $Y.\Delta X_y$, where ΔX_y is any perturbation of the singular matrix $Y'B^e$, such that,

$$\begin{array}{ll} \sigma_{min}(Y'B^e + \Delta X_y) = \sigma & (a) \\ ||\Delta X_y||_2 = \sigma & (b) \end{array} \tag{5.35}$$

A matrix that satisfies this condition is easily constructed from the singular value decomposition of $Y'B^e$ [34].

We now show that for small σ, the above construction yields a fairly good solution to our problem.

Theorem 5.1: For sufficiently small σ, ΔX_0 as constructed above, satisfies the (near) optimality condition (5.33).

Proof:

Since $\{A^e, B^e\}$ has only an uncontrollable mode at $\lambda = 1$, the *positive* function of λ:

$$\sigma_{min}\{[A^e - \lambda I \mid B^e]\} \tag{5.36}$$

becomes zero only for $\lambda = 1$. Let all the other *local minima* of this *continuous* function be at least larger than $\overline{\sigma}$ and let $\sigma << \overline{\sigma}$. Then a perturbation ΔX_0 that makes this function larger than σ at $\lambda = 1$, will normally make it larger than σ for *all* values of λ since the other minima were sufficiently far from zero. In order to find a perturbation that brings the function (5.36) above the level σ for $\lambda = 1$, it is natural to consider the singular values of the matrix $[A^e - I|B^e]$ or its transformed form:

$$\left(\frac{U'}{Y'}\right).(A^e - I|B^e) = \left(\frac{U'(A^e - I) \mid U'B^e}{0 \quad\quad \mid Y'B^e}\right) \tag{5.37}$$

where $Y'B^e$ does not have full rank, indicating the nb uncontrollable modes. Assuming σ to be much smaller than the smallest singular value of $U'(A^e - I)$ (which is non singular because of step 1), then it is sufficient to make $\sigma_{min}\{Y'B^e + \Delta X_y\}$ equal to σ in order to guarantee that the smallest singular value of (5.37) is approximately equal to σ [34] and hence also,

$$\sigma_{min}[A^e - I|B^e + \Delta X_0] \approx \sigma \tag{5.38}$$

Here we thus focused on the uncontrollable mode $\lambda = 1$ only, but for sufficiently small σ the controllability of other modes remained largely unaffected.

■

Corollary 5.1: The procedure of making the system controllable corresponds to inserting more input noise to the zero bias errors. This allows the corresponding entries of the state error covariance matrix P to increase linearly. Therefore this procedure will allow "*selective*" linear forgetting [64].

The influence of this perturbation on the *convergence rate* of the recursion of the Riccati equation in a SRCF is illustrated in fig. 5.7.
Let σ_{min} be the smallest singular value of the controllability matrix of the original system $\{A, B\}$, namely,

$$[B \quad A.B \quad ... \quad A^{n-1}.B]$$

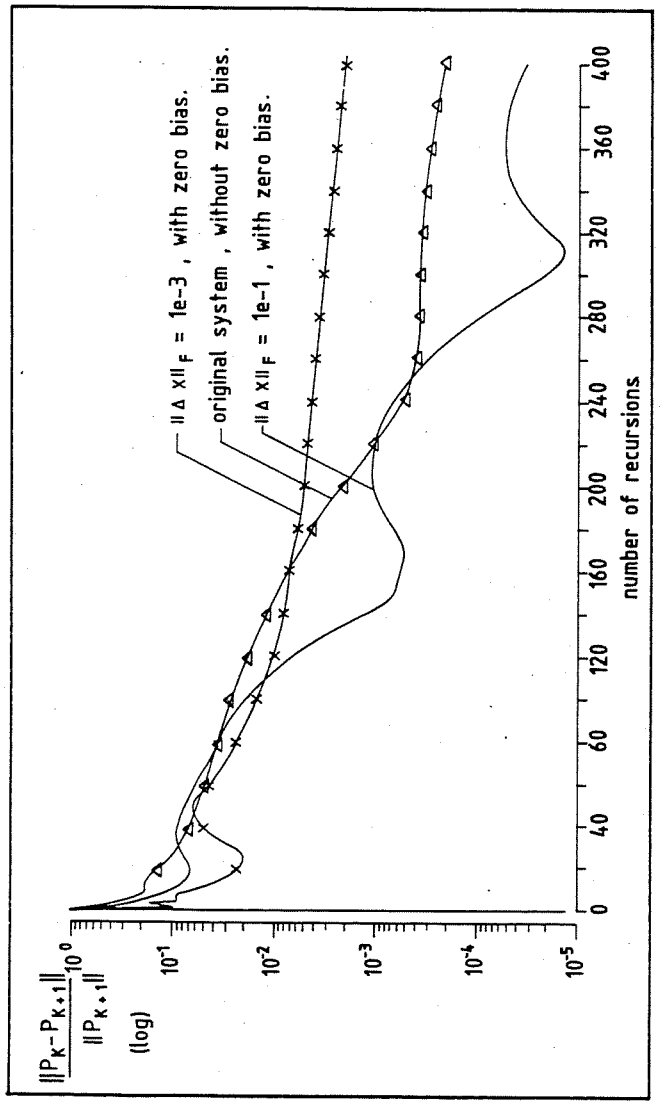


Figure 5.7: Influence on the rate of convergence by the new technique of making the bias terms controllable.

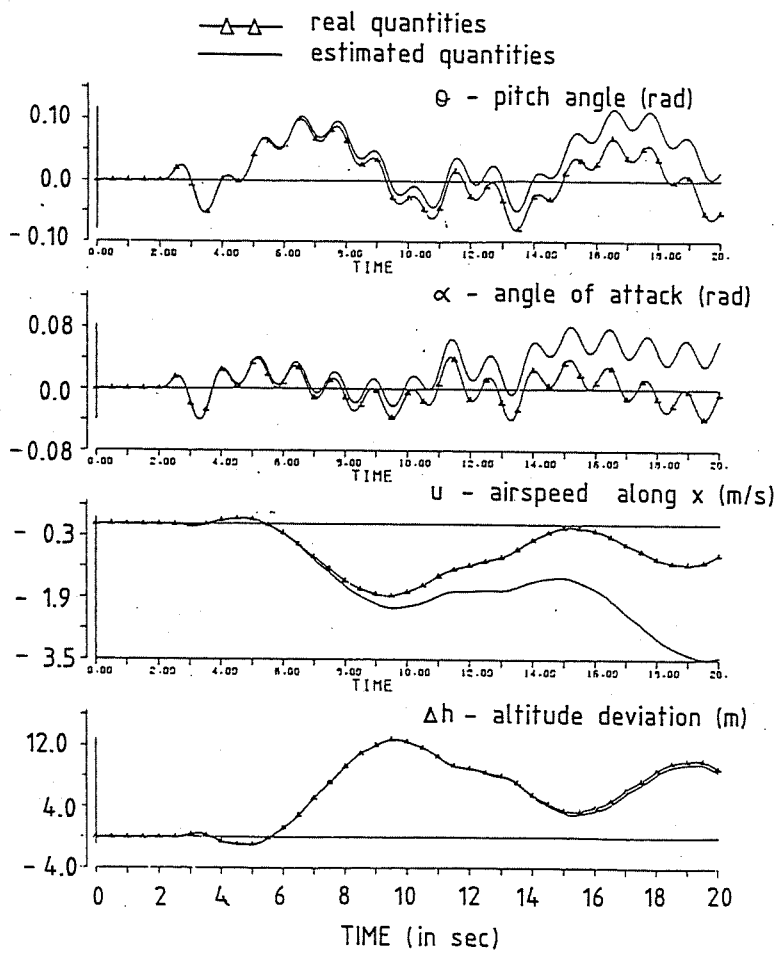


Figure 5.8: Reconstructed and real flight-path quantities under the presence of linear time varying zero bias errors for $\|\Delta X\|_F = 0$.

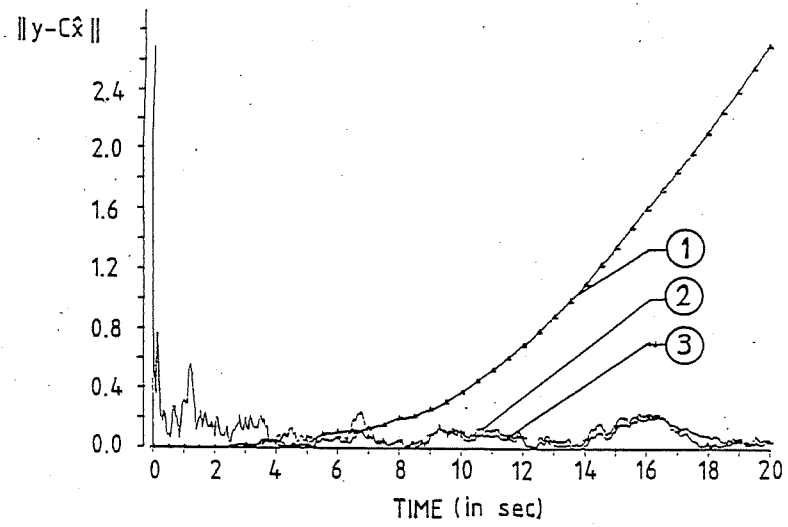
Then, for $||\Delta X||_2 < \sigma_{min}$ (which was 10^{-3} for the flight-path reconstruction problem) the rate of convergence is *slowed* down compared to the rate that occurred when only the original system $\{A, B\}$ was considered. This is displayed in figure 5.7 for $||\Delta X||_2 = 10^{-3}$. For larger perturbations, the rate of convergence is increased compared to the original rate. This is indicated in figure 5.7 for $||\Delta X||_2 = 10^{-1}$. The improvement on the convergence behaviour is one important feature of the developed procedure, however more important is that the increased input noise by the perturbation ΔX on B^e does not deteriorate the filtered estimates. This is investigated in the next subsection.

5.3.2.3. Adaptive tracking of time-varying bias errors

The additional advantages of the procedure described in the previous section in real-time applications is demonstrated in this section. We no longer assume the bias parameters to be *constant*, but they can *vary slowly* in time. If we use the same SRCF implementation as for the basic flight-path reconstruction problem in the presence of time variations on the observable zero bias errors (displayed in figure 5.11), the reconstruction errors increase *unacceptably* after a very short period. This can be observed from figure 5.8, where the real and reconstructed state quantities for the symmetric aircraft motion are displayed. The information about the real state quantities is in practical situations unkown. An alternative indicator of the accuracy performance of the KF is given by the 2-norm of the *innovation sequence*, denoted as $||C^e.\hat{x}_{k|k-1} - y_k||_2$. By comparing fig. 5.8 and fig. 5.9 we clearly observe that this measure also detects filter deterioration very well when $||\Delta X||_F = 0$.

The above filter deterioration may be overcome by the use of the procedure developed in the previous subsection. This procedure can be included in the SRCF algorithm in a very efficient manner, discussed in the next subsection. To use the procedure we have to specify the perturbation bound σ. This bound can be related to $||C^e.\hat{x}_{k|k-1} - y_k||_2$ to make the algorithm *self-adaptive*. On the other hand, defining σ as a constant, enables the SRCF to correct the state reconstruction results *very accurately* in the presence of the time varying bias errors of the previous test. Indeed, the norm of the innovation sequence, see figure 5.9 for $||\Delta X||_2 = 10^{-1}$, does not increase. Furthermore, it can be observed from figure 5.9 that this KF performance index remains in the vicinity of what would be obtained for constant but unknown bias terms, shown e.g: in fig. 5.9 for $||\Delta X||_2 = 10^{-3}$. The effect

on the reconstructed flight-path, see figure 5.10, compared to the previous test makes the developed procedure *very attractive*. From this figure it is observed that the use of accurate inertial signals, indicated by the measurement statistics used in test 6 of section 5.2, still results in very small state reconstruction errors.



(1) $||\Delta X||_F = 0$, with the zero bias errors varying linearly as given in fig. 5.11

(2) $||\Delta X||_F = 1.0\ 10^{-1}$ for the same circumstances as (1)

(3) $||\Delta X||_F = 1.0\ 10^{-3}$ for constant, unknown zero bias errors

Figure 5.9: Performance of the SRCF indicated by the 2-norm of the innovation signal

As mentioned above, the bias error on the rate gyro λ_q leads to a more significant deterioration of the filter estimates than the accelerometer bias errors. From figure 5.11, it is observed that it is exactly this bias λ_q that is followed most accurately, which is thus a desirable feature.

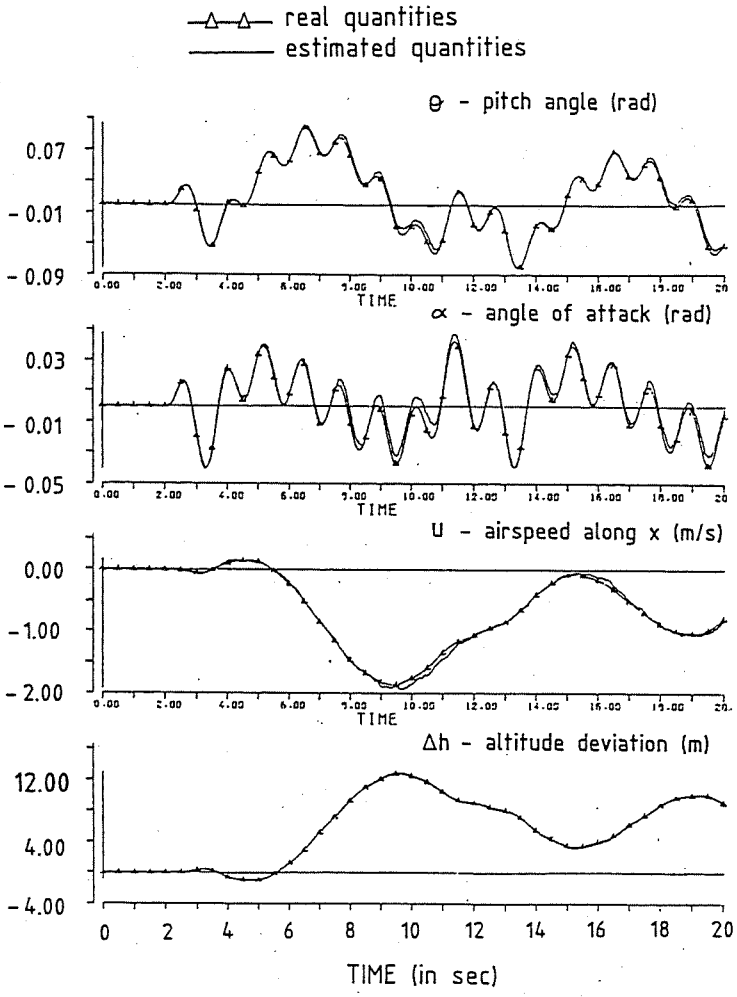


Figure 5.10: Reconstructed and real flight-path quantities under the presence of linear time varying zero bias errors for $\|\Delta X\|_F = 1.10^{-1}$.

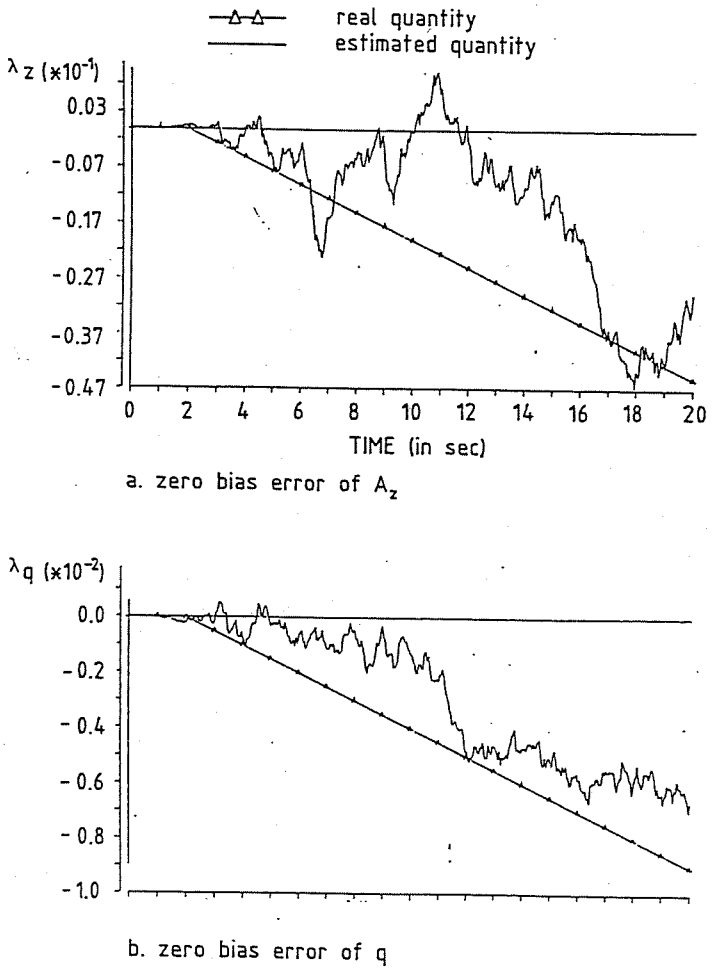


Figure 5.11: Time variations and reconstructions of the observable zero bias errors λ_z and λ_q.

5.3.2.4. Efficient inclusion of the zero-bias errors in the SRCF

The special structure of the system matrices of the extended state space description (5.27 - 5.28) where the noise input distribution matrix is updated by the procedure discussed in section 5.3.2.3., is indicated below for the dimensions of the flight-path reconstruction problem, i.e. n=4, m=3, nb=2, and p=2,:

$$A^e = \left(\begin{array}{cc|cccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \hline x & x & x & 0 & 0 & 0 \\ x & x & x & x & 0 & 0 \\ x & x & x & x & x & 0 \\ x & x & x & x & x & x \end{array}\right) \; ; \; B^e = \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{array}\right)$$

$$C^e = \left(\begin{array}{cc|cccc} 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \end{array}\right)$$

This condensed structure leads first to computational savings in the construction of the pre-array of the SRCF by the efficient calculation of the products $C^e.S_k$, $A^e.S_k$ and $(B^e + \Delta X_0).Q_k^{1/2}$. Where ΔX_0 can be put into the following special structure:

$$\Delta X_0 = \left(\begin{array}{ccc} x & 0 & 0 \\ x & x & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right)$$

Secondly, it leads to a special structure of the pre-array, which can be exploited in the triangularization as shown below:

$$\left(\begin{array}{cc|ccc|cc|ccc|c} \mathbf{x} & 0 & 0 & 0 & 0 & x_{11} & x_9 & x_7 & x_5 & x_3 & x_1 \\ x & \mathbf{x} & 0 & 0 & 0 & x_{12} & x_{10} & x_8 & x_6 & x_4 & x_2 \\ \hline 0 & 0 & \mathbf{x} & 0 & 0 & x_{13} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & x & \mathbf{x} & 0 & x_{14} & x_{14} & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & x & x & \mathbf{x} & x_{15} & x_{15} & x_{15} & 0 & 0 & 0 \\ 0 & 0 & x & x & x & \mathbf{x} & x_{16} & x_{16} & x_{16} & 0 & 0 \\ 0 & 0 & x & x & x & x & \mathbf{x} & x_{17} & x_{17} & x_{17} & 0 \\ 0 & 0 & x & x & x & x & x & \mathbf{x} & x_{18} & x_{18} & x_{18} \end{array}\right)$$

The annihilation of the elements x_1 up to x_{12} in the (1,3) and (1,4) blocks in the pre-array is done via *Givens rotations* as described in the previous chapter. The further reduction by *Householder transformations* of the elements x_{13} up to x_{18} becomes now more efficient due to the special structure of ΔX_0. Here the first nb transformations eliminating x_{13} and x_{14} operate on very small matrices. This leads to a further reduction of the computational complexity compared to the CKF.

5.4. Synthesis of the adaptive flight-path reconstruction module

The use of the inertial signals established to determine the variations in the system matrices $\{A, B, C, D\}$ given in model (5.24 - 5.25) completely by the value of the airspeed V and the flight-path angle γ in the reference condition. This becomes clear from observing the state transition matrix in (1.18). Practical experience in flight test technology with civil aircrafts has learned that for the whole flight envelope the flight-path angle remains close to zero. Therefore, V_T becomes a crucial parameter in the proposed algorithm to detect variations in the system dynamics and to prescribe forgetting patterns, i.e. to tune an exponential weighting factor c_w.

A block diagram of the complete adaptive flight path reconstruction module is given in figure 5.12. This can be divided in two parts:

1. In a "slow" supervising part one detects system parameter variations and updates the system matrices used in the extended filtering problem. Highest priority is given to the detection of system parameter variations due to variations in the reference flight condition. When no variations are indicated by this mechanism, attention is given to variations in the zero bias errors, because these variations occur at a much slower rate and dominate the reconstruction errors in silent flight phases. Therefore, the developed procedure to treat zero bias errors takes only a time-invariant system representation $\{A_T, B_T, C_T\}$ into account. On the other hand both mechanisms automatically activate the KF updating, when necessary.

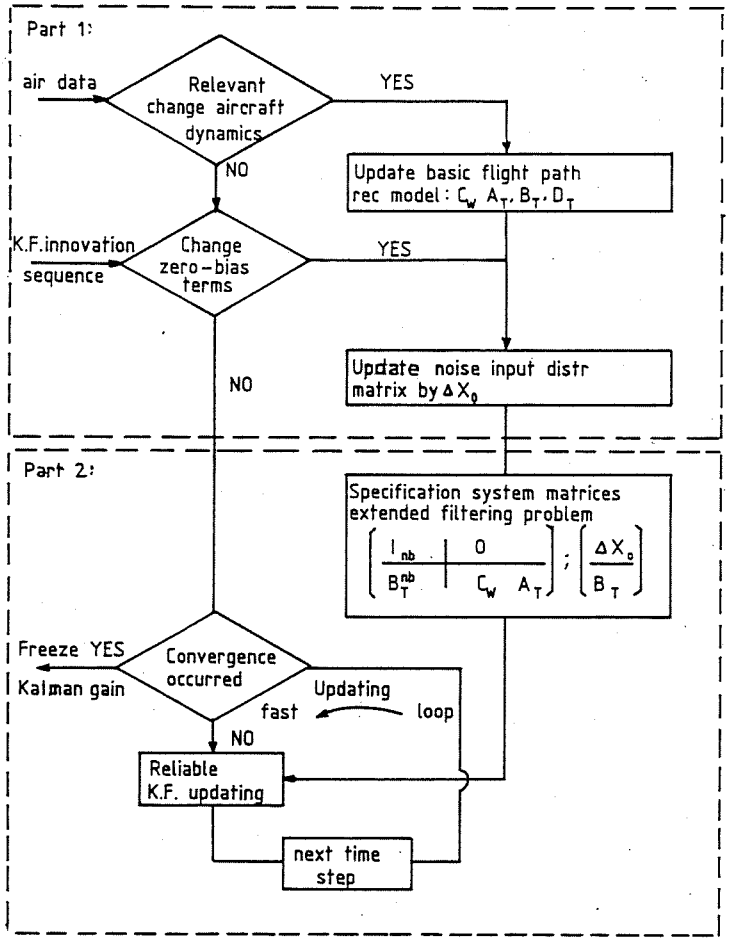


Figure 5.12: Information stream through the adaptive flight-path reconstruction algorithm.

2. In a second part the fast KF updating is performed by the reliable new SRCF implementation of section 5.3.2.4 for the extended filtering problem. This part operates at a speed determined by the data acquisition system. This causes a very heavy workload on the computational unit. Therefore, the use of very efficient SRCF implementations is of primary importance. However, to reduce the computational burden when possible an automatic convergence detection procedure has been developed that stops KF updating when sufficient accuracy is obtained.

5.5. Concluding remarks

The presented algorithm in this chapter allows to reconstruct the flight-path in an *efficient, reliable and accurate* way. The developed implementations and modifications of the Kalman filter makes the computations *robust* with respect to *round-off errors, discretization errors and modeling errors.* A separate analysis of a particular kind of modeling error, namely the zero bias errors on the inertial signals, resulted in a procedure to cope with these error sources in a very accurate way. Furthermore, the presented procedure operates *completely independently*, taking into account in a simple way the variations in the reference flight condition. This makes it a *much more attractive solution* to the flight-path reconstruction problem than the INS and RNS combination used on the present civil aircrafts, such as the $DC10$ and $B747$. This is e.g. so because for the latter combination the errors on the reconstructed flight-path may become *inacceptable large* during certain time intervals. This is demonstrated by some simple examples in this chapter.

The accurate reconstruction of the aircraft's flight-path can now be used in the following step of the two-step identification procedure to determine the so-called stability and control derivatives. This is outlined in the following chapter.

Chapter 6.

The Aerodynamic Model Identification Module

6.1. Introduction

In this chapter we analyze the second step of the two-step identification method described in chapter 1. In this step the *aerodynamic model* is determined. This model characterizes the aerodynamic forces and moments that act on the aircraft. Generally, this is done by expanding them into Taylor series with respect to the state and input variables of the aircraft. The structure of these polynomials is however not a priori fixed. Moreover, the structure may vary in time for different flight conditions. The *purpose* of aerodynamic model identification is therefore twofold:

(a) First, the structure of these polynomials has to be determined.

(b) Secondly, based on the information of the aircraft's state and input variables, the unknown coëfficients in these polynomials have to be *estimated.*

Section 6.2 addresses more concretely these two basic problems. Here, it is outlined that once the structure of these polynomials is specified, the remaining estimation problem fits in the framework of *linear regression.* As outlined in Chapter 1, for the real-time aircraft identification problem we are only interested in the linear terms of the Taylor series expansion. Therefore, the aerodynamic model structure is *a priori fixed.* In order to model the aircraft dynamics adequately over the whole flight envelope,

the unknown coëfficients in these polynomials should be *varying in time.* Section 6.3 refers to a number of practical problems that may occur in estimating time-varying parameters with Least Squares (LS) methods. This section starts with a short review of the LS estimation problem. Section 6.4 starts with the treatment of the problems mentioned in section 6.3. In this section we analyze the *reliability and efficiency* of different recursive LS (RLS) methods. For this, it is again possible to make use of the theoretical error analysis performed in chapter 4. The complete error analysis will be performed for three different RLS algorithms. The solutions to the other practical problems, mentioned in section 6.3 are presented in section 6.5. The resulting algorithms are then combined in a modular way. Section 6.6 synthesizes the overall structure of this *modular adaptive RLS algorithm* and in the final section 6.7 we present some conclusions.

6.2. The aerodynamic model identification problem

As indicated in the introduction the aerodynamic model relates the aerodynamical forces and moments that act on the aircraft to the aircraft's flight-path and to the control surface deflections. The complexity, i.e. the structure, of this model depends on the type of aircrafts under consideration, on the section of the flight envelope and response frequency ranges we want to cover with the model. These aspects will be explained in more detail in the following example.

Example 6.1: In this example we present the identification results of the moment of pitch M, indicated in fig. 1.6, of a Phantom F-4 fighter aircraft at 40.000 ft altitude [36]. The polynomial for the dimensionless moment of pitch, denoted by C_m, for angles of attack α between 20° and 40° extracted from the recorded flight test measurements is:

$$C_m = C_{m_0} + C_{m_\alpha}.\alpha + \underbrace{C_{m_{\alpha^2}}.\alpha^2 + C_{m_{\alpha^3}}.\alpha^3}_{\text{non-linear terms}} + C_{m_\beta}.\beta + C_{m_q}.q + C_{m_{\delta_e}}.\delta_e \qquad (6.1)$$

We will not discuss the procedure used to extract this polynomial from the recorded data sequences, but we only *focus* on the particular structure of this polynomial in α. From (6.1), we clearly observe the *non-linearity*

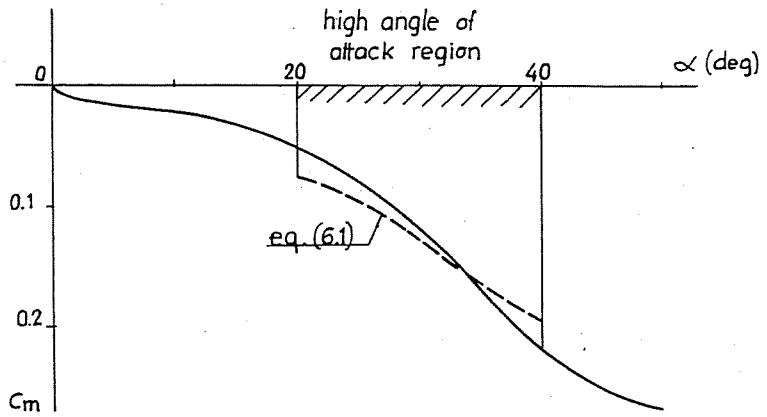


Figure 6.1: The moment of pitch C_m versus α, measured in the windtunnel and identified from flight tests.

in α. The presence of this non-linearity could also be observed from windtunnel measurements of C_m versus α. This result is given in figure 6.1. In this figure we also plotted the identified polynomial (6.1) for the numerical values of C_{m_α}, $C_{m_{\alpha^2}}$ and $C_{m_{\alpha^3}}$ taken from [36]. This is indicated by the dotted line in figure 6.1. The mismatching between the two curves in the so-called "high angle of attack region" (for $20° \leq \alpha \leq 40°$) is due to the differences which exist between the experimental conditions in real flight test and in the windtunnel test. However, this mismatching is not really important here.

What is more important is that this example learns us a number of facts *in determining the structure* of the Taylor series to describe the aerodynamic forces and moments:

1. The windtunnel results give us some guidelines about which terms should be included in the series expansions. This may be very helpful in the determination of the final polynomial structure. Until now, this is generally done by a statistical analysis technique, called "stepwise regression". This method works *iteratively*, whereby in each iteration a new term is included in the polynomial based on a statistical test, such as e.g. the F-test (see [54] for more details). This iteration continues until the aerodynamic quantity is determined by the model to some specified accuracy.

2. For the type of aircraft under consideration it is clear that the flight envelope, and more specifically the range of the angle of attack, is very large. From fig. 6.1 we learn that the angle of attack may vary from 0° to over 40°. Correspondingly, also the aircraft dynamics will change considerably for such types of aircrafts.

3. If we want to describe these dynamical properties globally over the whole flight envelope, very complex polynomials will be necessary. E.g. if we want to represent the windtunnel measurements given in fig. 6.1 by a single polynomial, it would be necessary to include "significant" terms up to α^9. If we are only interested in restricted parts of the flight envelope, much more simpler polynomials will result. Such as e.g. the third order polynomial as considered in the above example for the high-angle-of-attack region.

These observations clarify that if we are only interested in the current aircraft dynamics, i.e. in a small part of the flight region, the aerodynamic forces and moments can be adequately modelled by the *linear terms* in the series expansion. For the example 6.1, this corresponds to approximating the whole C_m versus α curve by a number of linear parts. For such a part where the linear approximation is valid, we were able in chapter 1 to derive the linear state space model given by (1.9). Therefore, the aerodynamic model structure that is of primary concern for our application is *a priori fixed*. The relationships are linear in the aircraft state and input variables and summarized by the model (1.10). The proportional constants in these relations, i.e. *the stability- and control derivatives*, will now *be time varying*. As a result, each row of the aerodynamic model (1.10) can be rewritten as:

$$y(t) = \vartheta_T' \varphi(t) \tag{6.2}$$

where $y(t)$ – represents the measured aerodynamic force or moment (these quantities can be measured very accurately, see e.g. [11])

$\varphi(t)$ – the aircraft state and input variables, which for the symmetric aircraft motion are given by $\varphi(t) = (u(t), \alpha(t), \frac{q(t)c}{V_T}, \delta_e(t))'$

ϑ_T – the unknown stability- and control derivatives which are constant for the reference condition at time T, i.e. when the linear approximation is valid

If we include the measurement errors on the measured aerodynamic forces and moments, and assume that the aircraft's state and input time histories are *known*, equation (6.2) may be cast into a *linear regression model*:

$$y(t) = \vartheta_T' \varphi(t) + e(t) \tag{6.3}$$

with $E\{e^2(t)\} = \sigma^2$ given by measurement calibrations, see e.g. in [41]. From mathematical point of view the term "regression model" is only valid when the regression variables or regressors $\varphi(t)$ in (6.3) are *exactly* known. However, these variables result from the *accurate* flight-path reconstruction technique discussed in the previous chapter or accurate in-flight measurement. Hence, the bias errors in the parameter estimates given by a LS technique can be *neglected*.

6.3. Least Squares estimation of time-varying parameters

The aerodynamic model identification problem for the particular application considered in this thesis has been formulated as a *Least Squares problem of time varying parameters*. The aspects related to this particular problem will be investigated in this section. First, we briefly review the LS method.

6.3.1. The Least Squares method

The linear regression model (6.3) can be rewritten as follows:

$$y_k = \vartheta' \varphi_k + e_k \tag{6.4}$$

where now the subscript T of ϑ has been dropped and where we assume $\varphi \in R^n$. The LS estimate of φ is given by,

$$\hat{\vartheta}_N = \arg\, min\, J_1(\vartheta) \tag{6.5}$$

with $J_1(\vartheta) = \sum_{i=1}^{N} \alpha_i (y_i - \vartheta' \varphi_i)^2$

Here $\{\alpha_i\}$ is a sequence of positive numbers. The inclusion of the coefficients α_i in the loss function $J_1(\vartheta)$ allows to give different weights to different observations. If the parameter vector ϑ is *constant*, we will assume no particular preference. Therefore all α_i's are taken equal to one. It should however be remarked that a constant α_i equal to the inverse of the variance of $e(t)$, i.e. $\alpha_i = (\frac{1}{\sigma^2})^{-1}$, results in minimum variance estimate [64]. Simple calculations show that the LS estimate (6.5) is given by,

$$\hat{\vartheta}_N = R_N^{-1} f_N \tag{6.6}$$

$$R_N = \sum_{i=1}^{N} \varphi_i \varphi_i' \alpha_i \tag{6.7}$$

$$f_N = \sum_{i=1}^{N} \varphi_i y_i \alpha_i \tag{6.8}$$

From these equations, it is clear that when all α_i's are equal to a constant α, then this *constant* does not influence the LS estimate.

For real-time operation, it is of primary importance to compute the estimates *recursively*. Therefore, the set of equations (6.6-6.8) are rearranged by simple manipulations to the following *recursive form* [64]:

$$\hat{\vartheta}_k = \hat{\vartheta}_{k-1} + R_k^{-1} \varphi_k (y_k - \hat{\vartheta}_{k-1}' \varphi_k) \tag{6.9}$$

$$R_k = R_{k-1} + \varphi_k \varphi_k' \tag{6.10}$$

The matrix R_k is called the *information matrix*, see [64]. In each recursion the new information contained in φ_k is added according to (6.10). The algorithm (6.9-6.10) is not well suited for computation as it stands, since an $n \times n$ matrix has to be inverted each time step. It is therefore more natural to introduce,

$$P_k = R_k^{-1}$$

and applying the matrix inversion lemma, given by lemma 4.1, we can rewrite (6.9-6.10) as:

$$\hat{\vartheta}_k = \hat{\vartheta}_{k-1} + K_k(y_k - \hat{\vartheta}'_{k-1}\varphi_k) \tag{6.11}$$

$$K_k = \frac{P_{k-1}\varphi_k}{1 + \varphi'_k P_{k-1}\varphi_k} \tag{6.12}$$

$$P_k = P_{k-1} - \frac{P_{k-1}\varphi_k\varphi'_k P_{k-1}}{1 + \varphi'_k P_{k-1}\varphi_k} \tag{6.13}$$

The matrix $\sigma^2.P_k$ in (6.13) tends to become the covariance matrix of the estimated parameters for large k [64]. Therefore, the estimation scheme (6.11-6.13) not only gives us an estimate $\hat{\vartheta}_k$ but it also computes a measure of accuracy of this estimate. This scheme is perhaps the most common form of the recursive LS algorithm, and we shall call it the "conventional LS" (CLS) method. The properties of the LS estimates are well known, and discussed e.g. in the book of Ljung and Söderström [64]. Let us point out here a number of observations about the presented RLS algorithm, that will be necessary for a better understanding of the subsequent parts.

Observation 6.1: (The effect of the initial conditions)
Any recursive algorithm requires some initial values to be started up. For (6.11-6.13) we need $\hat{\vartheta}_0$ and P_0. The estimates resulting from (6.11-6.13) are then:

$$\hat{\vartheta}_N = [P_0^{-1} + \sum_{i=1}^{N} \varphi_i\varphi'_i]^{-1}[P_0^{-1}\hat{\vartheta}_0 + \sum_{i=1}^{N} \varphi_i y_i] \tag{6.14}$$

By comparing (6.14) and (6.6), we see that the relative importance of the initial values *decays with time*, as the magnitude of the sums increase. This property guarantees that the recursive estimates (6.14) approach the off-line estimates given by (6.6). This property, which is of course desirable, is satisfied if *sufficiently* new information enters the estimator at each recursion. This condition plays an important role in the *analysis and performance* of the RLS method. Generally, it is indicated as the "*persistent excitation*" condition and will be specified more precisely in the subsequent sections.

Observation 6.2: (The direction of the parameter update)
The loss function $J_1(\vartheta)$ given in (6.5) is quadratic in the parameters ϑ.

Hence, in the noise-free case it can be represented by a *hyperparaboloid.* The contours of constant values of the loss function are given by,

$$\partial J(\vartheta) = 0 \tag{6.15}$$

and these are *hyperellipsoids.* Let us now focus on these contours for $n = 2$. Fig. 6.2 gives an example of the contours for that case. The tangent line in a parameter vector $\hat{\vartheta}_{k-1}$ on one of these contours is given by,

$$\frac{\partial J_1(\vartheta)}{\partial \vartheta^1}\Big|_{\vartheta^1=\hat{\vartheta}^1_{k-1}} (\vartheta^1 - \hat{\vartheta}^1_{k-1}) + \frac{\partial J_1(\vartheta)}{\partial \vartheta^2}\Big|_{\vartheta^2=\hat{\vartheta}^2_{k-1}} (\vartheta^2 - \hat{\vartheta}^2_{k-1}) = 0 \tag{6.16}$$

This equation learns us that the gradient of J in $(\hat{\vartheta}^1, \hat{\vartheta}^2)_{k-1}$ is perpendic-

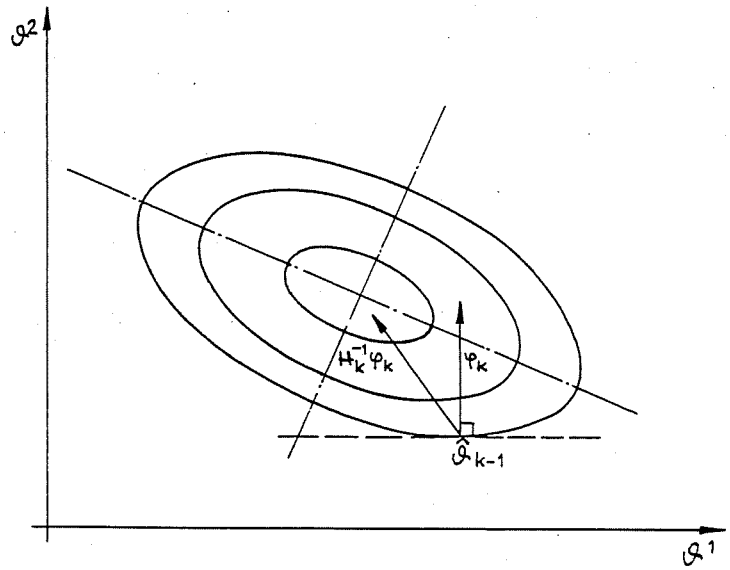


Figure 6.2: Contours of constant values $J_1(\vartheta)$ in the parameter space.

ular to this tangent line. For the quadratic loss function $J_1(\vartheta)$ in (6.5) it

can easily be shown that the gradient of $J_1(\vartheta)$ in $\hat{\vartheta}_{k-1}$ is given by,

$$\frac{\partial J_1(\vartheta)}{\partial \vartheta} \mid_{\vartheta=\hat{\vartheta}_{k-1}} = \varphi_k(y_k - \hat{\vartheta}'_{k-1}\varphi_k) \tag{6.17}$$

Which learns that the independent observations φ_k are also orthogonal to that tangent line. This is indicated in fig. 6.2. Now, if we observe (6.9), for R_k^{-1} equal to the identity matrix, the parameter step, i.e. $\hat{\vartheta}_k - \hat{\vartheta}_{k-1}$, is taken along the gradient of J_1 in $\hat{\vartheta}_{k-1}$. In this case the convergence rate is *linear*. On the other hand if R_k^{-1} is proportional to the inverse of the Hessian H_k:

$$H_k = \nabla^2 J(\vartheta) \mid_{\vartheta=\hat{\vartheta}_{k-1}} \tag{6.18}$$

the parameter step is taken towards the minimum of the hyperparaboloid, see fig. 6.2. In this case the convergence rate is *quadratic*. In general the actual direction of the parameter step will lie in between these two extremes.

Observation 6.3: (Connection with the Kalman filter)
This connection becomes clear if we write the linear regression model (6.4) as a linear state space model:

$$\vartheta_{k+1} = \vartheta_k \tag{6.19}$$

$$y_k = \varphi'_k\vartheta_k + e_k \quad \text{with} \quad E\{e_k^2\} = \sigma_k^2 \tag{6.20}$$

This model corresponds to the general state space model (3.1-3.2) used in the derivation of the KF in chapter 3, for the following system matrices:

$$A_k = I \ , \ B_k = D_k = Q_k = 0 \ , \ C_k = \varphi'_k \text{ and } R_k = \sigma_k^2 \tag{6.21}$$

For these system matrices the measurement update equations of the CKF, (4.13-4.16) represent the CLS scheme given by (6.11-6.13). On the other hand we observe that there is no need for a time update.

A very simply application of the derived RLS method is treated in the following example.

Example 6.2: In this example we want to fit a line in least squares sense through a number of noisy measurements given by:

$$y_k = \theta.k + e_k \tag{6.22}$$

Such a set of noisy measurements is displayed in fig. 6.3, indicated by the o's.

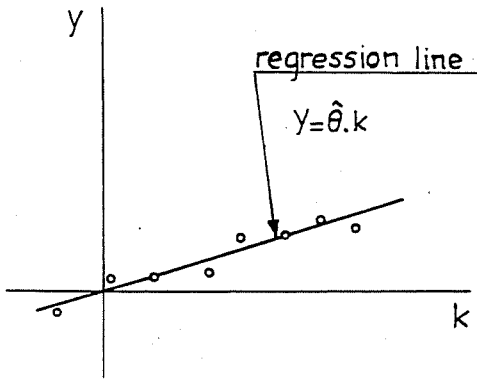


Figure 6.3: A very simple linear regression problem.

According to (6.9-6.10), the LS recursions for this problem are scalar and become:

$$\hat{\theta}_k = \hat{\theta}_{k-1} + \{\frac{r_k}{k}\}^{-1}(y_k - \hat{\theta}_{k-1}.k) \tag{6.23}$$

$$r_k = r_{k-1} + k^2 \tag{6.24}$$

This results for large k in the regression line $\hat{y}_k = \hat{\theta}.k$, displayed in fig. 6.3. The 1×1 matrix $p_k = r_k^{-1}$ becomes for this case:

$$p_k = [p_0^{-1} + \sum_{i=1}^{k} i^2]^{-1} \tag{6.25}$$

which *goes to zero* for large k. It is a desirable property that the accuracy of the estimates increases as more measurements becomes available. For this case it is due to the continuous increase of r_k in (6.24), which indicates that the independent observations φ_k always contain sufficiently new information, or are "persistently exciting". On the other hand, the gain $\{\frac{r_k}{k}\}^{-1}$ in (6.23) becomes:

$$\{\frac{r_k}{k}\}^{-1} = [\frac{1}{k}p_0^{-1} + \frac{1}{k}\sum_{i=1}^{k} i^2]^{-1} \tag{6.26}$$

which also approaches to zero for large k. So that for large k the parameter estimate $\hat{\theta}$ does not change anymore. This last phenomenon can be generalized as follows:

> If the independent observation signals φ_k remain *persistently exciting* the RLS scheme (6.8-6.9) or (6.11-6.13) will *turn itself off* after some time.

6.3.2. Difficulties with real-time RLS estimation

The *turn-off phenomenon* of the presented RLS schemes, indicated in example 6.2, makes the RLS estimator *insensitive* to new information after some time. For large k, the parameter update approximately becomes:

$$\hat{\vartheta}_k \approx \hat{\vartheta}_{k-1} \tag{6.27}$$

Therefore, when the true parameter values *are time varying*, an *unacceptable* bias will result. This difficulty can be overcome by *discounting old data* (information) from the estimator. It is a common practice to do this by weighting past data by an exponential decaying function. This corresponds to choosing the weighting coefficients $\{\alpha_i\}$ in (6.5) as in the following loss function:

$$J_2(\vartheta) = \sum_{i=1}^{N} \lambda^{N-i}(y_i - \vartheta'\varphi_i)^2 \tag{6.28}$$

with $\lambda < 1$.
The exponential weight function $\lambda^{N-i} = \lambda^t$ is displayed in figure 6.4 for different values of λ. When λ is close to 1, the weight function λ^t can be approximated as:

$$\lambda^t = e^{t.ln\lambda} = e^{t.ln(\lambda-1+1)} \approx e^{t(\lambda-1)} \tag{6.29}$$

Relation (6.29) tells that the measurements taken at $N - i \geq \frac{1}{1-\lambda}$ have a weight less than $e^{-1} \approx 36\%$ of that of the most recent data. This boundary is given by the dotted line in fig. 6.4. Therefore, the number of points that affects the estimates is roughly given by,

$$T_0 = \frac{1}{1-\lambda} \tag{6.30}$$

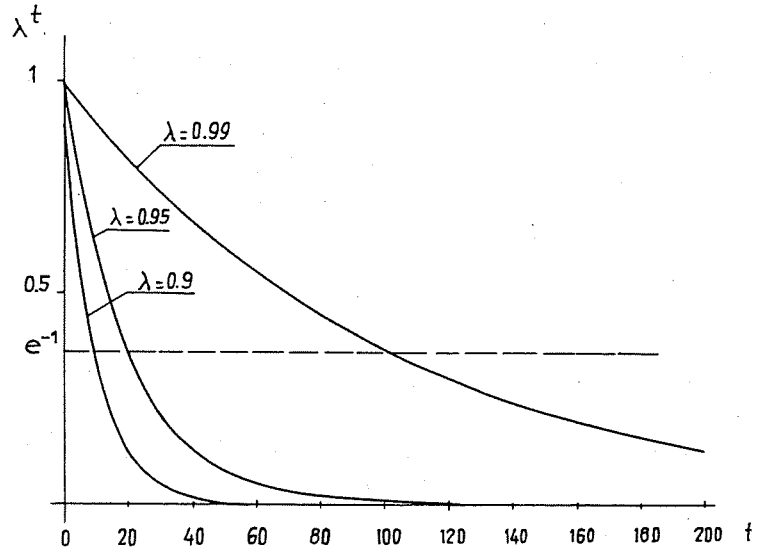


Figure 6.4: The weight function λ^t for different forgetting factors λ.

The recursive form of the LS estimate that minimizes (6.28) now becomes,

$$\hat{\vartheta}_k = \hat{\vartheta}_{k-1} + R_k^{-1}\varphi_k(y_k - \hat{\vartheta}'_{k-1}\varphi_k) \tag{6.31}$$

$$R_k = \lambda R_{k-1} + \varphi_k\varphi'_k \tag{6.32}$$

or with the application of the matrix inversion lemma:

$$\hat{\vartheta}_k = \hat{\vartheta}_{k-1} + K_k(y_k - \hat{\vartheta}'_{k-1}\varphi_k) \tag{6.33}$$

$$K_k = \frac{P_{k-1}\varphi_k}{\lambda + \varphi'_k P_{k-1}\varphi_k} \tag{6.34}$$

$$P_k = \frac{1}{\lambda}[P_{k-1} - \frac{P_{k-1}\varphi_k\varphi'_k P_{k-1}}{\lambda + \varphi'_k P_{k-1}\varphi_k}] \tag{6.35}$$

This new RLS scheme may offer a solution to the turn-off phenomenon, the problem that remains is how to choose λ. Experiments have shown, see e.g. [37], that this choice is generally a *trade-off* between fast adaptation capabilities of the estimator and long term qualities of the estimator, such as its accuracy. This trade-off is illustrated in the following example.

Example 6.3: Consider the process model:

$$y_{k+1} = \vartheta_T y_k + e_{k+1} \tag{6.36}$$

where y_k is the output signal and $\{e_k\}$ is a white noise sequence. The parameter ϑ_T is estimated according to (6.33-6.35). Three different values of the forgetting factor λ are used, namely $\lambda_1 = 0.95, \lambda_2 = 0.99$ and $\lambda_3 = 0.995$. These results are shown in fig. 6.5. At recursion $k = 100$, the parameter ϑ_T changes from -0.9 to -0.3. The trade-off in the choice of λ is obvious.

This example clarifies the existence of a number of problems related to the use of an exponential forgetting factor λ. First, the estimates become less accurate if a constant significant forgetting factor, e.g. $\lambda \leq 0.9$, is used when the unknown parameters do not vary continuously. Therefore, a significant exponential forgetting factor should only be used when a variation in the parameter values ϑ is *detected*. A second problem that may occur when a constant exponential forgetting factor less than one is used is estimator *wind-up or burst* [42]. This phenomenon can be considered as the reverse of the turn-off phenomenon discussed in example 6.2. It occurs when not sufficient information enters the estimator while using a constant forgetting factor. This problem can very well be demonstrated for the extreme condition where no information enters the estimator at all, namely $\varphi_k = 0$. For this case, P_k in (6.35) will increase exponentially, i.e. will wind-up. During such a "silent" time interval, no estimate updates are made since $K_k = 0$. However, when new information enters the estimator, K_k will become very large. This causes inacceptable large changes in the parameter estimates. This consequence is indicated as the *escape phenomenon* and may lead to *physically impossible* parameter estimates. Therefore, the algorithm should be adapted in order to avoid this kind of phenomenon.

The difficulties described above require the following adaptation of the presented recursive LS scheme:

1. For applications where the model parameters ϑ are constant over certain time intervals, it is necessary to develop a *detection mechanism* to trace the parameter changes in order to assure *accurate* parameter estimates.

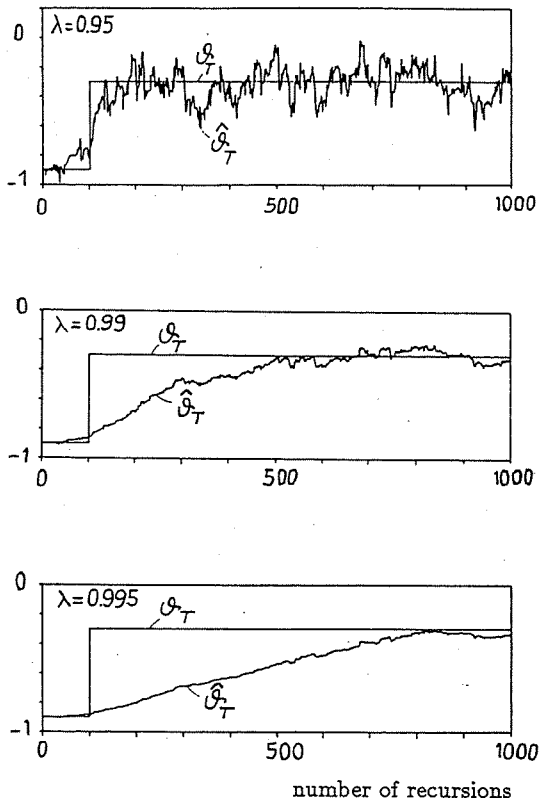


Figure 6.5: The behaviour of the parameter estimates for different forgetting factors λ.

2. Before prescribing a certain forgetting pattern, it should be *checked* firstly whether the observations φ_k contain sufficiently information, or as stated earlier , whether they are *persistently exciting*. Alternatively, use can be made of an extra disturbing signal that *assures* that this condition is always fulfilled. This approach is

e.g. used in self-tuning regulators [3]. Here, one can use the direct coupling that exist between the regulation and estimation part in the self-tuning regulator. This allows to insert an extra input signal, called a *probing signal*, that continuously *disturbs* the system in order to guarantee persistent excitation. However, when it is required to estimate the parameters in the aerodynamic model as accurate as possible, this solution is not very attractive. Therefore, we will explicitly check the persistent excitation condition.

3. A *constant small forgetting factor*, e.g. $\lambda = 0.98$, should be used in order to avoid the RLS scheme to turn itself off. This should of course only be used when the persistent excitation condition is guaranteed.

These algorithmic adaptations will be presented in section 6.5. First, we analyze the *reliability and efficiency* aspects of the different RLS schemes applicable to the aerodynamic model identification problem. These aspects can, as demonstrated in the previous chapter, certainly not be neglected for the real-time application that we are solving.

6.4. Computational aspects of different RLS methods

The computational aspects that will be considered are *reliability and efficiency*. These are analyzed for three different RLS implementations. In this analysis we make extensive use of observation 6.3. This is demonstrated in the following subsection in the derivation of these three filter implementations.

6.4.1. The different RLS implementations

6.4.1.1. The conventional recursive LS method (CLS)

This implementation is already derived in section 6.3. The recursive relationships that characterize the CLS method are given by (6.33-6.35).

6.4.1.2. The square root covariance recursive LS method (SCLS)

This is the equivalent of the SRCF implementation given by equations (4.2) and (4.3). If we use the matrices given in (6.21), the recursive relationship characterizing this implementation becomes:

$$\begin{pmatrix} \sqrt{\lambda}.\sigma & \varphi_k'.S_k \\ 0 & \frac{1}{\sqrt{\lambda}}.S_k \end{pmatrix}.U_4 = \begin{pmatrix} \sigma_k^e & 0 \\ G_k & S_{k+1} \end{pmatrix} \tag{6.37}$$

with the parameter update equation:

$$\hat{\vartheta}_{k+1} = \hat{\vartheta}_k + \frac{\lambda}{\sigma_k^e}.G_k(y_{k+1} - \hat{\vartheta}_k'\varphi_{k+1}) \tag{6.38}$$

and U_4 in (6.37) is again an arbitrary orthogonal transformation that triangularizes the pre-array. Here S_k is the Choleski factor of P_k, defined as follows:

$$P_k = S_k.S_k' \tag{6.39}$$

6.4.1.3. The square root information recursive LS method (SILS)

Similarly to the SRIF scheme given by (4.7), we can formulate for the LS problem recursions for the Choleski factor of the matrix R_k, defined in (6.7), and the parameter estimate $\hat{\vartheta}_k$ in a combined way, When the square root of R_k is defined as:

$$R_k = T_k'T_k \tag{6.40}$$

this implementation becomes:

$$U_5.\begin{pmatrix} \sqrt{\lambda}T_k & \sqrt{\lambda}\hat{\xi}_k \\ \frac{\varphi_{k+1}'}{\sigma_{k+1}} & \frac{y_{k+1}}{\sigma_{k+1}} \end{pmatrix} = \begin{pmatrix} T_{k+1} & \hat{\xi}_{k+1} \\ 0 & \varepsilon_{k+1} \end{pmatrix} \tag{6.41}$$

With U_5 operating in a similar manner as U_4. The update of the parameter estimates is given by:

$$\hat{\vartheta}_{k+1} = T_{k+1}^{-1}\hat{\xi}_{k+1} \tag{6.42}$$

Remark 6.1: Other, so-called *fast least-squares* algorithms, as e.g. described in [66], are not included in this analysis. The reason for this is that those special implementations require that the regressors φ_k possess a special *shift structure*, namely $\varphi_k = (-y_{k-1}, ..., -y_{k-n})'$. This certainly is not the case for the considered aerodynamic model identification problem.

6.4.2. Theoretical analysis of the reliability

The analysis of the effect of round-off errors on the RLS schemes, presented in section 6.4.1, can again be split up in three parts:

- the round-off errors made in a single recursion
- the propagation of a single error in the subsequent steps
- the interaction and accumulation of the errors performed in different steps

Moreover, the observation 6.3 will be exploited to use many of the results obtained in the reliability analysis of the KF given in section 4.3.

First, consider only *the propagation of a single error at time instant $k-1$ to subsequent instances without additional round-off errors.* Let the errors be denoted by δP_{k-1} and $\delta\hat{\vartheta}_{k-1}$ and defined as follows:

$$\overline{P}_{k-1} = P_{k-1} + \delta P_{k-1} \quad \text{and} \quad \overline{\hat{\vartheta}}_{k-1} = \hat{\vartheta}_{k-1} + \delta\hat{\vartheta}_{k-1}$$

The error δP_{k-1} propagates according to (4.36) for the state space model defined by (6.21). This results in:

$$\delta P_k = \frac{1}{\lambda}(I - K_k\varphi_k')\delta P_{k-1}(I - K_k\varphi_k')' + \frac{1}{\lambda}(\delta P_{k-1} - \delta P_{k-1}')\varphi_k K_k' + O(\delta^2) \tag{6.43}$$

Similarly with (4.37) we receive the following equation for the propagation of the error $\delta\hat{\vartheta}_{k-1}$:

$$\delta\hat{\vartheta}_k = (I - K_k\varphi_k')\{\delta\hat{\vartheta}_{k-1} + \frac{\delta P_{k-1}\varphi_k}{(\lambda\sigma + \varphi_k' P_{k-1}\varphi_k)}(y_k - \varphi_k'\hat{\vartheta}_{k-1})\} + O(\delta^2) \tag{6.44}$$

Equation (6.43) learns that when using an exponential forgetting factor $\lambda < 1$, the *loss of symmetry* certainly *blows up* in the error propagation. A direct conclusion that we can already draw at this point is that the symmetry of the matrix P_k must absolutely be guaranteed here under *all circumstances*. On the other hand, the matrix $(I - K_k\varphi_k')$ plays a crucial role in the propagation of the errors δP_{k-1} and $\delta\hat{\vartheta}_{k-1}$. Therefore, it will be inspected more closely. By comparing (6.31) and (6.33) we can write the gain K_k as:

$$K_k = R_k^{-1}\varphi_k \tag{6.45}$$

and using (6.32) we find that:

$$(I - K_k\varphi_k') = \lambda R_k^{-1} R_{k-1} \tag{6.46}$$

Now, consider the effect of an error induced at time instant $k_0 < k$ on the computed quantities $\overline{P}_k$ and $\overline{\hat{\vartheta}}_k$. When $\overline{P}_k$ remains symmetric, this effect becomes according to (6.43):

$$\delta P_k = \frac{1}{\lambda^{k-k_0}}\phi(k,k_0)\delta P_{k_0}\phi(k,k_0)' + O(\delta^2) \tag{6.47}$$

and according to (6.44):

$$\delta\hat{\vartheta}_k = \phi(k,k_0)\delta\hat{\vartheta}_{k_0} + \sum_{j=0}^{k-k_0-1}\phi(k,k_0+j).\mu_{k_0+1+j} + O(\delta^2) \tag{6.48}$$

with $\mu_i = \frac{\delta P_{i-1}\varphi_i}{(\lambda\sigma + \varphi_i' P_{i-1}\varphi_i)}(y_i - \varphi_i'\hat{\vartheta}_{i-1})$ and $\phi(k,k_0) = \prod_{i=k_0}^{k}(I - K_i\varphi_i')$.

Using (6.46) we can write this transition matrix $\phi(k,k_0)$ as:

$$\phi(k,k_0) = \lambda^{k-k_0} R_k^{-1} R_{k_0} \tag{6.49}$$

This transition matrix guarantees that the errors δP_{k_0} and $\delta\hat{\vartheta}_{k_0}$ are attenuated if the matrix product $\lambda^{k-k_0} R_k^{-1} R_{k_0}$ is a *contraction*. In the next theorem we will prove that this is indeed the case if the independent observations φ_k are *persistently exciting*. Therefore, we will first state a formal definition of this condition.

Definition 6.1: The independent observations φ_k are *persistently exciting* over the observation interval $k \leq i \leq k_0$ (with k larger than $k_0 + n$) if the following condition is fulfilled:

$$\alpha I \leq \sum_{i=k_0}^{k}\varphi_k\varphi_k' \leq \beta I \tag{6.50}$$

for some positive constants α and β.

Theorem 6.1: When the observations φ_k are persistently exiting over the interval $[k, k_0]$, the matrix product $\lambda^{k-k_0} R_k^{-1} R_{k_0}$, for $\lambda \leq 1$, is a *contraction.*

Proof:

The matrix R_k is defined in (6.7) as:

$$R_k = \sum_{i=0}^{k} \varphi_k \varphi_k' \lambda^{k-i}$$

Since $k_0 < k$ we have that:

$$R_k = \lambda^{k-k_0} R_{k_0} + \sum_{i=k_0}^{k} \varphi_k \varphi_k' \lambda^{k-i}$$

Using condition (6.50), the matrix difference $(R_k - \lambda^{k-k_0} R_{k_0})$ is *positive definite.* Alternatively this can be denoted as,

$$\lambda^{k-k_0} R_{k_0} < R_k$$

This proves that the product $\lambda^{k-k_0} R_k^{-1} R_{k_0}$ is a contraction for any value of $\lambda \leq 1$.

■

Corollary 6.1: Based on this theorem, we can for the first part of the analysis guarantee that as long as:

1. the observations are persistently exciting, what also assures that R_k^{-1} remains bounded (this can easily be shown).
2. use is made of a forgetting factor $\lambda < 1$
3. the matrix P_k is symmetric

the effect of a single error δP_{k_0} in (6.47) or $\delta \hat{\vartheta}_{k_0}$ in (6.48) *decays exponentially.* For the case $\lambda = 1$, the contraction of $R_k^{-1} R_{k_0}$ guarantees that the error propagation remains *stable.*

Remark 6.2: As indicated in section 4.3, the error propagation of the SRCF and the SRIF was the same as that of the CKF (with $P_{k|k-1}$ symmetric) for the case of infinite arithmetic. Therefore, it may be also stated here that the considered RLS implementations possess the *same error propagation behaviour.* This conclusion is also made in an alternative error analysis only for RLS schemes given in [65].

Secondly, we formulate *upperbounds for the errors made in each recursion by the different RLS implementations.* Here, we will make use of

theorem 4.1 of section 4.3. Since the considered RLS algorithms only process scalar measurements, the upperbounds for the P-matrix are the same for the three different RLS schemes. If we again denote this upperbound by Δ_p, we obtain:

$$\Delta_p \leq \epsilon_1 \|P_k\| \tag{6.51}$$

The upperbound for the error on the estimated parameters $\hat{\vartheta}_k$ becomes for the CLS and the SCLS:

$$\Delta_\vartheta \leq \epsilon_2 (\|\hat{\vartheta}_k\| + \|K_k\| . \|y_k\|) \tag{6.52}$$

and for the SILS:

$$\Delta_\vartheta \leq \epsilon_3 (\kappa^2(T_k)|\varepsilon_k| + \kappa(T_k)\|\hat{\vartheta}_k\| + \frac{|\varepsilon_k|}{cos\phi_1}) \tag{6.53}$$

In these bounds the ϵ_i's are again constants close to the machine precision and $cos\phi_1$ is defined by:

$$cos\phi_1 = \frac{|\varepsilon_k|}{\left\| \begin{matrix} T_k \hat{\vartheta}_k \\ \varepsilon_k \end{matrix} \right\|}$$

which is usually close to one.

The **third** part of the analysis *combines these upperbounds with the results of the propagation of a single error, to describe the total error.* According to section 4.3, the total error on the matrix P_k, denoted by $\delta_{tot} P_k$, satisfies:

$$\|\delta_{tot} P_k\| \leq \frac{1}{\lambda^{k-k_m}} \|\phi(k, k_m)\|^2 . \|\delta_{tot} P_{k_m}\| + \overline{\Delta}_p^m \tag{6.54}$$

In this equation k_m is the largest time instant, smaller than k, for which $\frac{1}{\lambda^{k-k_m}} \|\phi(k, k_m)\|^2$ is *smaller than 1.* And according to theorem 6.1, this can always be done when the observations are *persistently exciting.* On the other hand, the term $\overline{\Delta}_p^m$ represents the accumulated errors in the interval $[k_m, k]$, which according to (6.51) can be *bounded.* The total error on the parameter estimates becomes similarly,

$$\|\delta_{tot} \hat{\vartheta}_k\| \leq \|\phi(k, k_m)\| . \|\delta_{tot} \hat{\vartheta}_{k_m}\| + \overline{\Delta}_\vartheta^m \tag{6.55}$$

Both equations (6.54) and (6.55) indicate that when the observations φ_k are persistently exciting, the round-off errors produced in the three different RLS schemes will remain *bounded.* Furthermore, according to the upperbounds given in the second part of this analysis, *no preference* should be given to one of these three schemes, as long as we only consider the aspect reliability. This conclusion is of course made under the assumption that for the CLS the matrix P is forced to remain *symmetric.*

6.4.3. Comparison of the efficiency

An operation count of the different RLS schemes of section 6.4.1 strongly depends on the particular mechanization used. The mechanizations used throughout this thesis for the three different RLS algorithms are given in table 6.1-6.3. Here, we again exploit the condensed structure of the pre-array in the required triangularization of the SCLS and SILS schemes. It should however be remarked that this is also an example where condensed forms were already in common practice, see e.g. [8].

symbol	mathematical expression	number of flops
$P\varphi$	$P_{k-1}\varphi_k$	n^2
K_k	$\frac{P\varphi}{\varphi_k^t.P\varphi+1}$	$2n$
P_k	$P_{k-1}-K_k.P\varphi$	n^2
$\hat{\vartheta}_k$	$\hat{\vartheta}_{k-1}+K_k(y_k-\hat{\vartheta}'_{k-1}\varphi_k)$	$2n$

Table 6.1: Numerical mechanization of the CLS.

symbol	mathematical expression	number of flops
$S\varphi$	$S'_{k-1}\varphi_k$	$(n^2+n)/2$
	Householder triangularization	$\frac{3}{2}n^2+\frac{5}{2}n$
$\hat{\vartheta}_k$	$\hat{\vartheta}_{k-1}+K_k(y_k-\hat{\vartheta}'_{k-1}\varphi_k)$	$2n$

Table 6.2: Numerical mechanization of the SCLS.

symbol	mathematical expression	number of flops
	Householder triangularization	$\frac{3}{2}n^2+\frac{5}{2}n$
$\hat{\vartheta}_k$	$T_k^{-1}\hat{\xi}_k$	$n^2/2$

Table 6.3: Numerical mechanization of the SILS.

Table 6.4 gives a survey of the operation counts.

RLS scheme	complexity
CLS	$2n^2+4n$
SCLS	$2n^2+5n$
SILS	$2n^2+\frac{5}{2}n$

Table 6.4: Operation counts of different RLS.

From this table we observe that the operation counts for the three different implementations used are very similar. This has also been observed in the performed experiments, discussed in the next subsection.

The extensive use and analysis of the RLS method has resulted in a number of *slightly more efficient* schemes, applicable for regressor vectors with an arbitrary structure. Examples are:

- The computation of only the upper triangular part of P_k in the CLS method. This results in an algorithm with complexity $\frac{3}{2}n^2+4.5n$ [8].
- Using an alternative factorization of the matrix P_k given as $U_kD_kU_k'$, where D_k is a diagonal matrix and U_k is upper triangular with unit elements on the diagonal. Recursions for the matrices U and D are given e.g. in [8] and require only $\frac{3}{2}n^2+5.5n$ flops.

6.4.4. Experimental evaluation

In this experimental analysis, we evaluate the results of the performed theoretical analysis. This is done for the estimation problem of the *stability- and control derivatives* in the aerodynamic model. We only consider the estimation of the aerodynamical derivatives in the force equation along the X-axis. From (1.10), this equation can be written as follows:

$$A_X(t) = C_{X_u}u(t) + C_{X_\alpha}\alpha(t) + C_{X_q}\frac{q(t)C}{V_T} + C_{X_\delta}\delta_e \qquad (6.56)$$

The experiments to be presented, investigate the effect of *persistence of excitation* on the *accuracy* of the computations for the three different RLS schemes. Hereto, the aircraft was excited by a step on the elevator, which causes that the regressor vectors φ_i fail to fulfill the persistent excitation condition for $i > 160$. The way that is used to evaluate this condition is described in more detail in the next section.

As in the previous chapter, we evaluate the effect of round-off errors by performing mixed precision calculations. The accuracy is approximated by the magnitude (in norm) of the difference between the single and double precision quantities, denoted by $\overline{(.)}$ and $(.)$ respectively. For the estimated parameters, the total relative error is therefore approximated by:

$$\|\delta_{tot}\hat{\vartheta}_k\| = \frac{\|\overline{\hat{\vartheta}_k}-\hat{\vartheta}_k\|}{\|\overline{\hat{\vartheta}_k}\|} \qquad (6.57)$$

At the same time, we calculate the relative distance of the parameter estimates from their true parameter values ϑ, given as:

$$\frac{\|\overline{\hat{\vartheta}_k} - \vartheta\|}{\|\overline{\hat{\vartheta}_k}\|} \tag{6.58}$$

Fig. 6.6 presents these two quantities for the CLS for two different forgetting factors, namely $\lambda = 1$ (fig. 6.6a) and $\lambda = 0.9$ (fig. 6.6b). The results for the square root implementations for $\lambda = 1$ are given in fig. 6.7, namely for the SCLS in fig. 6.7a and for the SILS in fig. 6.7b. If we compare the quantity (6.57) in fig. 6.6a and in figures 6.7a-b, we observe clearly that $\|\delta_{tot}\hat{\vartheta}_k\|$ is *much higher* for the CLS than for the square root implementations. This is caused by the loss of symmetry of the matrix P_k. The jump in fig. 6.6a in the total error $\|\delta_{tot}\hat{\vartheta}_k\|$ corresponds to total loss of symmetry. From this instant on the computations become *completely unreliable*. When using an exponential forgetting factor $\lambda < 1$ the total error $\|\delta_{tot}\hat{\vartheta}_k\|$ *increases faster*, causing the jump to take place earlier, see fig. 6.6b. Therefore, these experiments confirm that for $\lambda < 1$ the error in the CLS will *grow faster*, and this mainly due to the *loss of symmetry*.

On the other hand, the behaviour of the error propagation for the SR implementations in fig. 6.7a-b is *very similar*. This is again completely compatible with the performed error analysis. We only observe a slow error build up, which is related to the reduced persistency of the excitation.

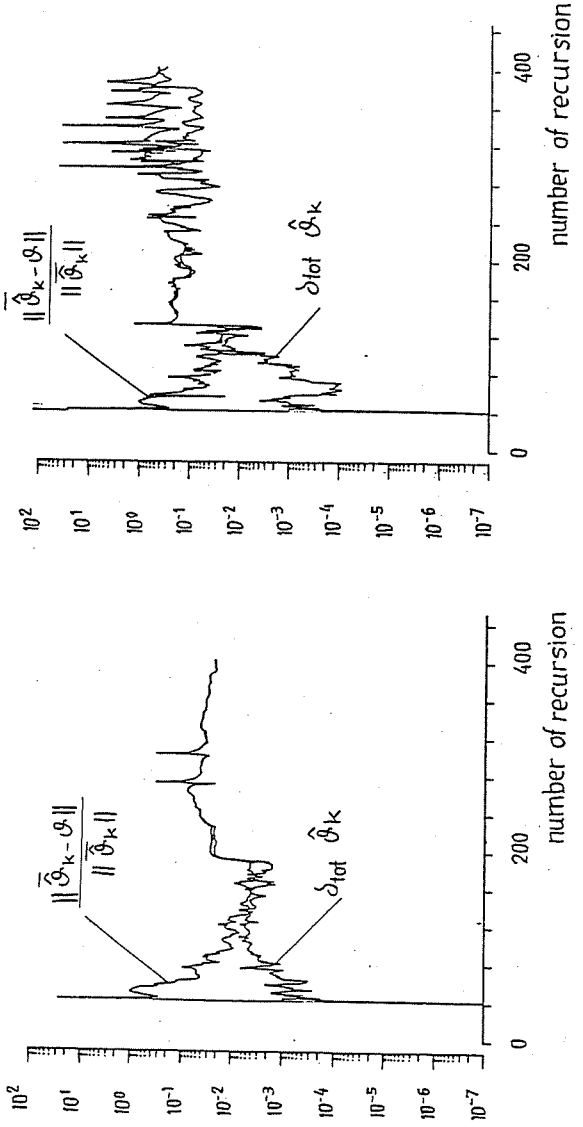


Figure 6.6a: Experimental evaluation of the numerical performance of the CLS for $\lambda = 1.0$.

Figure 6.6b: Experimental evaluation of the numerical performance of the CLS for $\lambda = 0.9$.

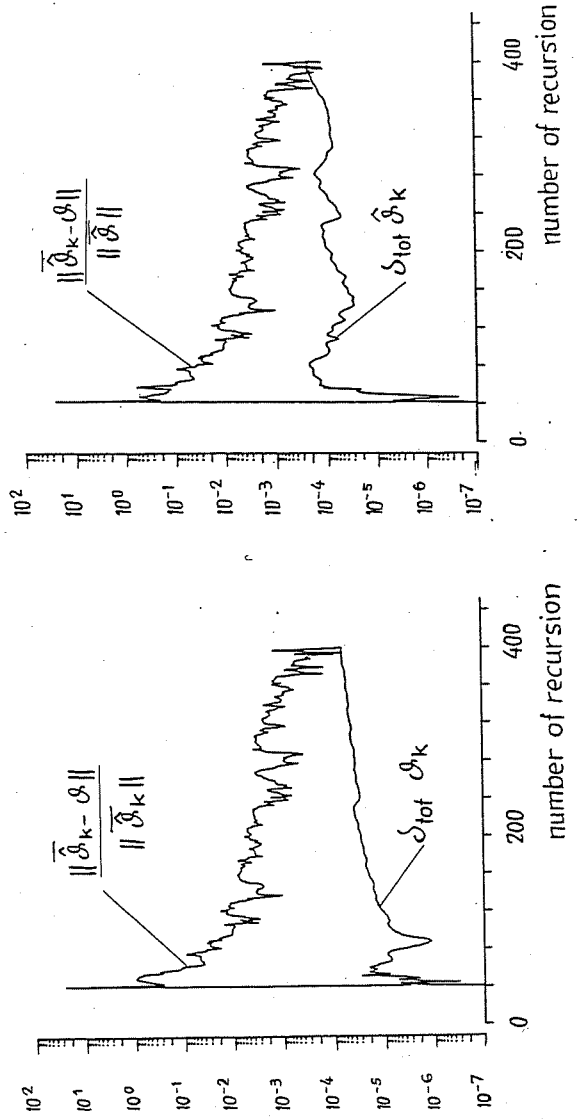


Figure 6.7a: Experimental evaluation of the numerical performance of the SCLS for $\lambda = 1.0$.

Figure 6.7b: Experimental evaluation of the numerical performance of the SILS for $\lambda = 1.0$.

6.5. Real-time aircraft model identification

In order to allow accurate real-time determination of the stability- and control derivatives, it is necessary to take into account the difficulties mentioned at the end of section 6.3. This is done in this thesis by expanding a *reliable* RLS scheme with two *external controlling mechanisms*:

- a first part that checks the persistent excitation condition of the regressor vector φ_i and
- a second part that takes into account the parameter variations.

The following subsections are devoted to the description of these two parts.

6.5.1. Checking the persistent excitation condition

6.5.1.1. The theoretical framework

This condition is defined in definition 6.1. The inequality (6.50) herein expresses nothing else than the *uniform observability* of the parameters ϑ in the time-varying state space description (6.19-6.20) [2]. Although (6.50) proves to be a very powerful analytical tool in the analysis of RLS methods, it is certainly not recommended for computational purposes. This is because the multiplication $\varphi_i\varphi_i'$ may destroy worthful information. Interesting examples of this phenomenon can e.g. be found in [123]. Therefore, we will look for an alternative procedure that operates immediately on the data φ_i.

If we arrange the set of measurements (6.4) in the time interval $[k_0, k]$ as follows:

$$\begin{pmatrix} \varphi_{k_0}' \\ \cdot \\ \cdot \\ \cdot \\ \varphi_k' \end{pmatrix} \vartheta = \begin{pmatrix} y_{k_0} \\ \cdot \\ \cdot \\ \cdot \\ y_k \end{pmatrix} - \begin{pmatrix} e_{k_0} \\ \cdot \\ \cdot \\ \cdot \\ e_k \end{pmatrix} \tag{6.59}$$

denoted as,

$$F\vartheta = Y \tag{6.60}$$

then we can easily see that the uniform observability of the parameters ϑ in the time interval $[k_0, k]$ corresponds to the *solvability* of ϑ in a *unique manner* from the set of equations (6.59). It is then required that the kernel of the matrix F in (6.59) is *empty*. However, the inaccuracies on the entries of the matrix F (due to reconstruction- or measurement errors) will always

cause this condition to be fulfilled. Therefore, to distinguish ill-conditioned from well-conditioned situations we use the more reliable *singular value decomposition* (SVD) of the matrix F [55]. This decomposition of the matrix F is given as,

$$F = U \left(\frac{\Sigma}{0} \right) V' \tag{6.61}$$

where Σ is $n \times n$ diagonal matrix containing the singular values σ_i in decreasing order on its diagonal and V' is $n \times n$ orthogonal matrix, with row vectors v_i forming and *orthonormal base of the parameter space.*

To clarify the direct link between the SVD and the persistent excitation condition we will use the *geometrical concept* of the *energy distribution* of a set of $(k - k_0)$ vectors. This concept is based on the following definition.

Definition 6.2: The magnitude of the energy of a set of $(k - k_0)$ vectors φ_i of dimension n in a direction e of an n-dimensional space is given by:

$$E_e = \sum_{i=k_0}^{k} (\varphi_i' e)^2 \tag{6.62}$$

According to this definition we can calculate and plot the energy E_e of the set of $(k - k_0)$ vectors φ_i in (6.59) in all directions e of the n-dimensional space. This leads e.g. for $n = 2$ to a picture given in fig. 6.8. The measurement errors or reconstruction errors possess a proper distribution, represented by the dotted line in fig. 6.8. This energy distribution picture clearly displays that part of the n-dimensional (parameter) space, for which the set of $(k-k_0)$ regressor vectors φ_i contain "*sufficient information*". Here the adjective "sufficient" is considered with respect to the measurement- or reconstruction errors. In that particular part of the parameter space, the regressor vectors φ_i are *persistently exciting.* Therefore, to check whether the persistent excitation condition is fulfilled, it is sufficient to inspect the *extreme values* and the corresponding *directions* of the distribution of φ_i with respect to distribution of the errors. Now it can easily be shown, see e.g. [91], that these two coincide with the *singular values* σ_i and the *right singular vectors* v_i of the matrix F respectively, see fig. 6.8.

These ideas are now incorporated in the following algorithm.

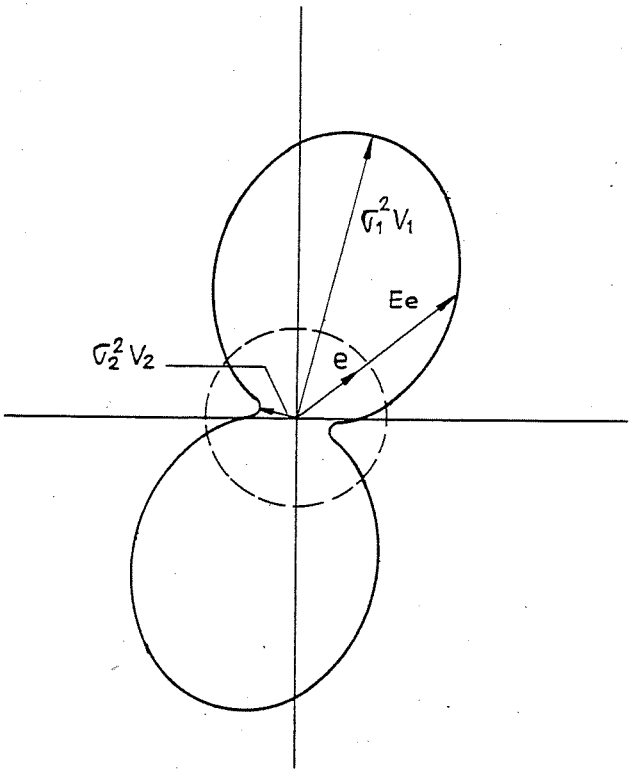


Figure 6.8: The energy distribution of a set of m-vectors for the two dimensional case.

6.5.1.2. Algorithmic design

To operate in an *adaptive way* we check the persistent excitation condition for a set of m vectors φ_i which lie inside a *window* that *moves* along with the current estimate ϑ_k. The organization is represented in fig. 6.9. The length of this window is determined by the following qualitative considerations:

1. The algorithm should be able to detect *parameter dependency* or *complete lack of information* in the signals as quickly as possible in

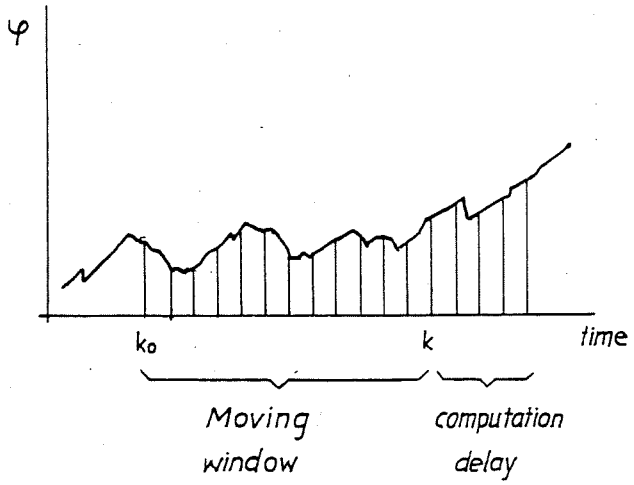


Figure 6.9: Moving window representation.

order to avoid the difficulties discussed in section 6.3. Therefore, the window length should be small. A motivation for this is clearly given by the second observation of the experiment displayed in fig. 6.10, discussed in the next subsection.

2. The window length should be not too small, since the smallest singular values of the F matrix are then very small and become very sensitive to perturbations [55]. The length should be at least larger than the number of unknown parameters.

3. When exponential forgetting schemes are used to deal with parameter variations the number of observation samples that will influence the parameter estimates are given in (6.30). Therefore, the window length should be adapted according to the exponential factor used, if necessary.

The window length is evaluated *experimentally*. A number of 20 samples has shown to amply satisfy the above conditions.

Secondly, we have to specify the *bounds* that indicate what we call *sufficient* in the light of the discussion given in the previous subsection.

According to the developed *geometrical framework*, the observations φ_i are persistently exciting if the smallest σ_i of F lies outside the distribution of the considered errors on the entries of F. This *lower bound* is specified as follows:

> When the magnitude of the errors on the elements of the regressor vector φ_i are known, it is possible to scale these elements so that they receive a common error level. This level is chosen to be *one*.

In the aerodynamic model identification problem the magnitude of the errors on the different elements of the F matrix are known, since they result from calibrated measurements or from the flight-path reconstruction technique, discussed in the previous chapter.

Finally, we note that *very efficient* SVD techniques are available to *update* the SVD of the regressor vectors in the moving window. This is due to the organization of the moving window. Namely, after each recursion one "old" regressor vector φ_{k_0} is deleted from the window and one "new" vector φ_{k+1} is added, see fig. 6.9. SVD techniques for this particular organization are e.g. discussed in [15].

6.5.1.3. Experimental evaluation

The developed procedure will also be applied to the determination of the unknown parameters in (6.56). In the experiment, we choose the elements of the regressor vector φ_i as follows:

- From recursions 1 to 40 the aircraft was not excited at all. This situation corresponds to "complete lack in persistent excitation".
- During recursions 100 and 150, the second element of the regressor vector in (6.55), i.e. $\alpha(t)$, is taken as a linear combination of the third element $\frac{q(t)c}{V_T}$ and noise.

The behaviour of the singular values is given in fig. 6.10. From this figure the following observations can be made:

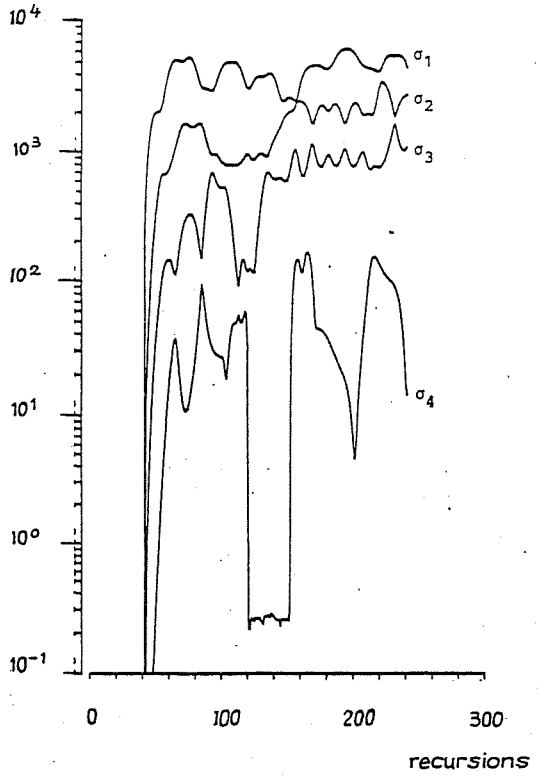


Figure 6.10: Experimental evaluation of checking the persistent excitation condition for a window length $m = 20$.

1. The time interval from recursion 1 to 40 is clearly detected. In this case, the algorithm should decide to "*freeze*" all the estimates.
2. The time interval where the second and third element of φ_i are dependent is *detected* with a delay of approximately the window length. This is certainly a reason to make the window length small. However, we considered a worst case, that does not occur during

practical applications. There it will not happen abruptly that the parameters become dependent or insufficiently exciting. It will however occur in a smoother manner. On the other hand, the instant when new information becomes available is detected immediately, see fig. 6.10.

These results confirm that the algorithm is able to tell us whether the regressor vectors φ_i in the moving window contain *new information* to update *all* parameters *or not*. If the last case occurs, as during recursions 100 and 150 of the experiment discussed above, we would like to know for which coëfficients still sufficient information is available. This is considered in the sequel.

6.5.1.4. Model reduction

When the persistent excitation condition is not satisfied, the solution ϑ of the set of equations (6.59) can be computed for a new F-matrix, given as:

$$\hat{F} = U \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_p \\ \hline & 0 & \end{pmatrix} \hat{V}' \tag{6.63}$$

where σ_1 to σ_p fulfill the condition specified by algorithm described above. The matrix $\hat{V}'$ results from deleting the last $(n-p)$ rows from the matrix V', given in (6.61). Computing the solution in this way corresponds to determining *special linear combinations* of ϑ, given by $\hat{V}'.\vartheta$. Since the parameters ϑ in the considered problem have a physical interpretation, this is not acceptable. On the other hand, we also do not use the SVD of the F matrix to compute the LSQ estimates in order to *increase* the accuracy of the estimates. If we would do this, the estimates will always be determined based on a number of n (20 in our case) measurements only. And as demonstrated in example 6.3, the use of only a *restricted* number of measurements to determine the parameter estimates (i.e. $\lambda << 1$) makes the estimates *very noisy*, see fig. 6.5. We could of course increase the number of measurements in the window, which is according to the first qualitative consideration in section 6.5.1.2 certainly *not desirable*. From this discussion we see clearly that the separation of checking the persistent excitation condition and computing the RLS estimates makes the combination *very*

flexible. Therefore, to allow the determination of the parameter estimates via the RLS method we will first reduce the model according to the following procedure.

Model reduction procedure:

Since the right singular vectors v_i of the F matrix are *normalized*, a v vector may be called "*strongly directed*" in one direction of the parameter space when the corresponding coördinate of that v vector in the particular parameter direction exceeds the value 0.5 in magnitude. According to this definition we can check the elements of the different v vectors for each recursion. Now if we denote an element of a v vector that satisfies the above condition by a +-sign and otherwise by a −-sign when not, we may obtain a "*selection map*" as e.g. represented for $n = 4$ in fig. 6.11a. This map resulted for the elements of the calculated V matrix of the previous experiment during recursions 100 and 150.

	parameter directions				
v vectors	ϑ^1	ϑ^2	ϑ^3	ϑ^4	
v_1	+	+	−	−	
v_2	−	−	−	+	"retained v vectors"
v_3	+	−	+	−	
v_4	−	+	+	−	"rejected v vectors"

Figure 6.11a: The selection map during recursions 100 and 150.

The information included in the selection maps allows us to reduce the model according to the following two rules.

Rule 1: If the $(n-p)$ "rejected" v vectors are "strongly directed" in that part of the parameter space that is not "strongly covered" by the retained v vectors, then that part of the parameter space is deleted from the model. This corresponds e.g. to the following selection map, displayed in fig. 6.11b, which advises to delete the ϑ_2 from the model.

	parameter directions				
v vectors	ϑ^1	ϑ^2	ϑ^3	ϑ^4	
v_1	+	−	−	−	
v_2	−	−	−	+	"retained v vectors"
v_3	+	−	+	−	
v_4	−	+	−	−	"rejected v vectors"

Figure 6.11b: An example of a selection map corresponding to rule 1.

Rule 2: If a part of the parameter space is "strongly covered" by the $(n-p)$ "rejected" v vectors as well as by the p retained v vectors, then that ill-conditioning is ascribed to parameter dependency between the parameters lying in this $(n-p)$ dimensional subspace. And in this case we would include only one of those parameters in the model. This situation corresponds e.g. to the selection map of fig. 6.11a, which advices only to use whether ϑ_2 or ϑ_3 in the model.

According to these rules, the selection map that resulted during recursion 100 and 150 of the previous experiment, see fig. 6.11a, clearly allows us to detect the parameter dependency that was simulated.

6.5.2. Accurate dealing with time-varying parameters

The stability- and control derivatives in the aerodynamic model (1.10) *may vary during certain time intervals.* As pointed out at the end of section 6.3.2, it is necessary to *detect* these parameter changes in order to assure *accurate* parameter estimation.

Generally speaking, such a detection mechanism implies the following. It indicates a variation in the parameters when a *significant change* occurs in a *certain measure of a quantity* of the RLS estimation scheme (6.33-6.35) or of the system under consideration. Here, both the choice of this measure and what is considered as significant is determined *heuristically.* However, we can distinguish two classes:

1. In the first class, a function of the *residuals*, $\varepsilon_k = y_k - \hat{\vartheta}'_{k-1}\varphi_k$, of the RLS method is taken as a measure. An example of such a function is the 2-norm. According to a heuristically derived relation to that measure, the forgetting factor used in the estimation scheme is prescribed. An overview of different relationships of this kind is e.g. given in [37].

2. The second class uses measurable *key parameters* of the system under consideration as a measure. These measures are then used in combination with *a priori information* about the system to determine *whether* a parameter change has occurred. According to this information again very precise forgetting patterns could be defined in order to follow these changes accurately.

Although both procedures (or a combination) can be used to trace the time variations in the aerodynamic derivatives, we will only describe a possible procedure that fits in the second class. This is because a vast

amount of *a priori information* is available about the changes in aerodynamic derivatives. This results for example from windtunnel measurements. From this experience we have learned that there are a number of *key parameters*, which are measured during flight, that indicate changes in the dynamical properties [24]. An overview of these parameters is given in the following list:

1. Environmental factors: These comprise the changes in altitude and velocity that occur over the whole envelope of the flight region. One measurable variable reflecting very well the changes in environment is the *aerodynamic pressure*,

$$Q = \frac{1}{2}\rho V_T^2$$

where ρ is the air density.

2. Weight factors: The changes in weight, moments of inertia and location of the centre of gravity results from the fuel consumption and from different conditions (passengers, cargo, etc.). This information is available at the beginning of each flight, and also influences the aerodynamic behaviour of the airplane.

3. Trust settings: The influence of the trust variations on the aerodynamic properties is for a propellor driven aircraft, such as the DHC-2 Beaver aircraft very important. A very good indication for this influence is the parameter,

$$\frac{\Delta p_t}{Q}$$

which is the dimensionless pressure increase over the propellor, measured at an arbitrary point in the propellor slipstream.

4. Geometrical factors: The changing configuration produced during manoeuvres, for example the selection of flap settings for the landing, influence of course the aircraft dynamics.

The adjustment of procedures based on this idea of course strongly depend on the type of aircraft under consideration and on the information available. Therefore, a practical design of this kind is not explicitly performed in this thesis.

6.6. Synthesis of the aerodynamic model identification module

The results of the analysis given in the previous sections can be combined in a *modular adaptive least squares algorithm*. This algorithm consists of three different modules:

- *module I*, which checks the persistent excitation condition.
- *module II*, which treats the parameter variations.
- *module III*, which performs the RLS estimation.

The information stream through these different modules is pictured in fig. 6.12.

According to the discussion given in sections 6.4.2 and 6.4.3, the first module plays a *supervising role* in this algorithm.

- It avoids the *blow up* phenomenon of the elements of the P matrix in (6.35) and hence avoids the parameter estimates *to escape* to physically inacceptable values,
- It assures that the RLS computations can be performed in a reliable way and
- It determines the structure of the regression model (6.2), i.e. about which parameters the regressor vectors contain sufficiently new information in order to update their estimates. The other parameter values are *frozen*.

As a remark for this last property, the first module can be considered as a better (a more reliable and more efficient) alternative for the so-called backward-stepwise regression method [54]. This is so because it relies on the reliable SVD concept and also because it does not operate iteratively.

The second module *detects* the variations in the stability- and control derivatives. The described procedure for this purpose relies on a vast amount of a priori information about the dynamical properties of the aircraft. This information source is available via windtunnel measurements. Via a number of key parameters it allows to determine the changes in aerodynamical derivatives. According to these measures, a forgetting pattern can precisely be *prescribed*. These two functions of the second module are displayed in fig. 6.12.

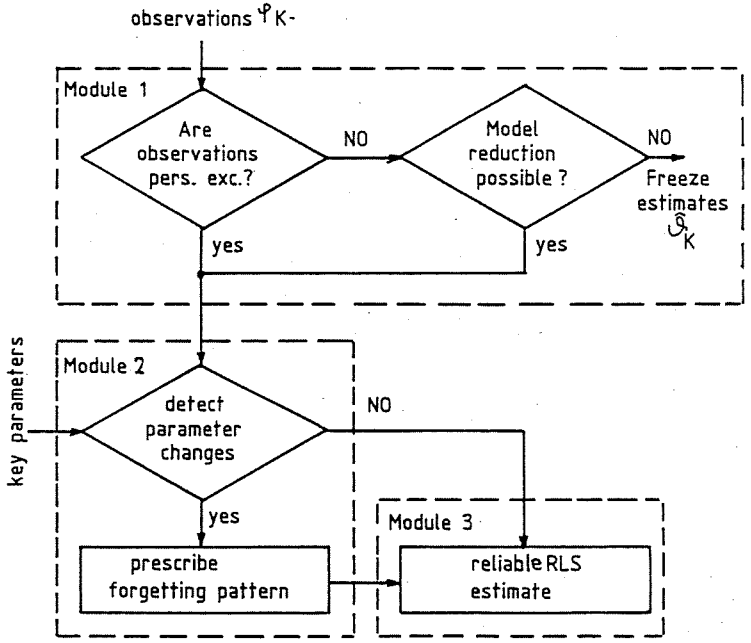


Figure 6.12: Information stream through the adaptive aerodynamic model identification algorithm.

Based on the information of the first and second module, the last module performs the RLS estimation. Therefore, we use a reliable RLS scheme. To make up this choice out of existing RLS schemes we developed analytical models for the propagation of the round-off errors for three different RLS methods. Furthermore, the combination of persistent excitation and the use of an exponential forgetting factor smaller than one guarantees *exponential convergence* of the parameter estimates [44].

6.7. Concluding remarks

The algorithm presented in this chapter allows the *real-time estimation* of the unknown stability and control derivatives in the *linear* aerodynamical model (1.10). The developed *efficient* and *reliable* procedures lead to the *accurate* estimation of these proportional constants taking into account various deteriorating influences occurring real-time.

The *real-time knowledge* of these coëfficients allow, as outlined in the introductory chapter, to improve further the operation and performance of different aircraft systems. This was particularly indicated for the Flight Management Systems. The general consequences of this new capability is still difficult to oversee. However, it possesses a number of keys that will make the flight(s) of civil aircrafts more *economical* and *safer*.

Chapter 7.

The conclusions

The research reported in this thesis is clearly stimulated by the urgent need to make flight execution, especially of civil aircrafts, *more economical and safer*. Certainly, under the impulse of the rapid developments in micro-electronics, one has tried to pursue this goal by the *automation* and *integration* of different aircraft systems using digital technology. However, experience of aircraft manufacturers demonstrated that *digital technology alone* is not sufficient to meet the severe requirements in order to reduce the aircraft operation costs while guaranteeing safety. Recently, it has become clear that a further improvement of the aircraft operation can be achieved by making use, in an intelligent way, of *system theory*. Particularly, it has been shown in this work (chapter 1) for Flight Management Systems (FMS) that a further optimization of flight requires *to identify a mathematical state space model of the aircraft that adapts automatically to the varying operational circumstances*. The *design of algorithms* to realize this idea, i.e. to make real-time aircraft model identification possible, is the *main objective* of this thesis. In the following section we review the obtained results. This evaluation allows to clarify the specific contents of the main title of this thesis. In the final section, a number of suggestions for further research related to the obtained results are made.

7.1. Review of the obtained results

The real-time and adaptive identification of a state space model of the aircraft has received much attention the last decade. Until now, several research projects on this topic have abandoned despite the use of very

powerful computers [13]. One of the main reasons was that many simplifications were needed in order to solve this non-linear identification problem. These are not necessary for the two-step identification procedure [30] considered in this work, since this procedure reformulates the *non-linear* aircraft model identification problems into *two linear* estimation problems. These problems are referred to as the flight-path reconstruction problem and the aerodynamic model identification problem respectively.

This problem separation allows to reconstruct the aircraft's flight-path via *linear filtering*. In the formulation of the system model used for this purpose, we encountered a number of *very interesting* features. First, they facilitate the *discrete* implementation of that linear filter and allow to verify *analytically* different system properties, such as controllability, observability and the eigenvalue structure of the state transition matrix, for all reference flight conditions. Secondly, the system can be exactly represented in the *Schur form* or *Observer Hessenberg form*. The interesting observation made here is that these forms fit in a more general class of system structures. Namely those forms of the state space model $\{A, B, C, D\}$, which are characterized *firstly* by a special structure of the system matrices which contain many zeros and *secondly* by the way they are obtained, i.e. *without loss of accuracy*. In numerical analysis, this class is indicated by "condensed forms". In **chapter 2**, we demonstrated the applicability of these forms in designing *reliable and efficient* algorithms for a broad class of problems in linear system theory. Throughout this thesis, this *general algorithmic design technique* is used to meet the requirements of real-time operation.

In order to solve the linear filtering problem, use could be made of two existing techniques, namely:

(a) the *Luenberger observer*

(b) the *Kalman filter*

The first technique has been investigated in **chapter 2**. Although, we came forward with a very *efficient solution* to reconstruct the flight-path based on condensed forms, the lack of knowledge where to locate the observer poles may lead to *very inaccurate* reconstruction results when the measurements contain errors, even when the latter are small. For the Kalman filter (KF), this location problem does not exist. This is because the observer dynamics are automatically specified. Furthermore, the reconstruction results become "optimal" in *minimum variance sense*.

In *chapter 3*, a unified approach was presented based on the work of C.C. Paige [81], to derive a KF implementation suited for the specific

filtering problem at hand. It has been shown that this approach covers all existing KF implementations which are valid for the general time varying case. Furthermore, from the engineering point of view, it is important to remark here that this approach allows *to derive* these implementations only by *straightforward algebraic computations.*

The analysis of the on-line applicability from now on concentrated on the investigation and adaptation of the numerical KF implementations in order to fulfill the severe requirements in real-time data processing. The three different requirements treated in **chapter 4 and 5** are now briefly discussed.

A primary requirement to assure accurate flight-path reconstruction is that the KF implementation is *numerically reliable.* In *contrast* to a very time consuming simulation study to compare the reliability of different KF implementations for the variety of system conditions that may occur real-time, use is made of a *theoretical error analysis.* This analysis is performed in **chapter 4** for four different KF implementations:

1. the conventional Kalman filter (CKF)
2. the Chandrasekhar square root filter (CSRF)
3. the square root information filter (SRIF)
4. the square root covariance filter (SRCF)

for three types of error sources, namely *round-off errors, modeling errors and discretization errors.*

From this analysis, a *number of guidelines* are formulated that enable without experimental analysis to choose the most reliable KF implementation for the given problem. To make this choice, only the system model, the measurement error statistics of the input and output signals and the initial state error covariance matrix are used. The validity of these guidelines is verified in a simulation study using the basic flight-path reconstruction problem. This simulation study, performed in **chapter 5**, also confirmed the *deficiencies* of the different KF implementations as predicted by the theoretical error analysis.

1. The CKF becomes *completely unreliable*, due to the loss of symmetry of the state error covariance matrix P, when the original state transition matrix A is *unstable* or has its eigenvalues on the unit circle. This problem can be circumvented by a simple trick, e.g. to symmetrize P after each recursion. However, the experiments

confirmed that the accuracy of this "symmetrized" CKF remains always lower than for the SRF's.

2. The loss of accuracy for the CSRF on the computation of the Kalman gain and hence on the filtered estimates is always *higher* than for other SRF's. And under certain circumstances, which are indicated in the theoretical analysis, this loss of accuracy may be very large. Furthermore, it is only applicable for the time invariant case.

3. The SRIF state error covariance computation will deteriorate for high condition numbers of the matrices A, Q and R and the state filtering will be less accurate when the condition number of P is *large.*

On the other hand, this simulation study showed the *excellent numerical behaviour of the SRCF* for the conditions encountered in the flight-path reconstruction problem.

Secondly, this SRCF implementation is analyzed more deeply to take into account several conditions that occur during real-time operation and will deteriorate the filter estimates. In **chapter 5**, we considered (a) variations in the aircraft system dynamics due to a change in reference condition and (b) the presence of zero bias errors on the inertial measurements. Therefore, the algorithm is extended by two modules, which take these variations into account and are *self-tuning.* In the first module, the air data information about the aircraft speed V_T and the flight-path angle γ_T is used to determine variations in the aircraft system dynamics and to follow them. When no variations are detected by this module the second module takes over and *tracks time-varying zero bias errors.* These two modules *speed up the rate of convergence* of the KF recursions while assuring *minimal loss of accuracy* of the reconstructed flight-path. This is done by appropriately tuning the exponential weighting factor c_w and also by calculating the *minimal perturbation* of the input distribution matrix of the extended state space description, in which the observable zero bias errors are included as state variables (in order to make the zero bias terms *maximally controllable*).

A third important requirement of the filter implementation is *efficiency.* The special "condensed" structure of the extended state description resulted in a very time consuming SRCF implementations, which on a IBM 370 machine was 40% faster than the CKF for the flight-path reconstruction problem. On the other hand, to reduce the workload of the computing unit during "silent" operational circumstances or to increase efficient use of it,

a very simple procedure is developed, based on the error models, described in chapter 4, to control the convergence of the recursive SRCF scheme.

The *accurate knowledge* of the aircraft's flight-path, guaranteed by the modified SRCF implementation, allows to determine the aerodynamic model, i.e. to estimate the stability and control derivatives, by the simple *least squares method.*

In **chapter 6**, we investigated and adapted different recursive least squares (RLS) methods in order to allow the real-time determination of the stability and control derivatives.

A *first part* of this analysis concentrated on the *reliability* and *efficiency* of different RLS procedures. For the analysis of the reliability a similar *theoretical* error propagation study is made as for the different KF implementations. This study highlighted that the conventional RLS method again produces *completely unreliable* estimates due to the loss of symmetry of the error covariance matrix of the parameter estimates. Furthermore, it has been proved that this inacceptable deficiency becomes worser when use is made of *exponential forgetting.* On the other hand, the error propagation models derived for the square root RLS (SRLS) schemes, demonstrated that when the *regressor vectors are persistently exciting no preference* should be given to any particular SRLS scheme, as long as we only consider the aspect reliability. The conclusions made in this theoretical analysis were *confirmed* experimentally for the aerodynamic model identification problem.

The *second part* of the analysis investigated the difficulties which may occur during the real-time operation of the RLS method. In order to handle these difficulties, the reliable RLS scheme has been expanded by two extra modules.

A first module was designed *to check the persistent excitation condition* of the regressor vectors *in an explicit way.* The developed procedure, which relies on the *reliable* singular value decomposition concept, *supervises in a self-adaptive way*, the operation of the RLS scheme in order to *guarantee* that the stability and control derivatives could be *accurately* estimated under possible *ill-conditioned circumstances*, such as parameter dependency.

The second module *detects* time variations in the stability and control derivatives and *prescribes* appropriate forgetting patterns. For this purpose we have only formulated a number of *guidelines* which may be helpful in the actual design of the module.

An experimental evaluation with the aerodynamic model identification problem demonstrated the excellent capabilities of the second module to

control the whole RLS scheme.

7.2. Main conclusion

The new algorithms and new implementations of existing algorithms, based on "condensed forms" (**chapters 2,4 and 5**) certainly form a **new class of algorithms in linear system theory**. The unification of the algorithmic reliability and efficiency that characterizes this class, is a very crucial property when using these algorithms real-time.

The theoretical error analysis of **chapter 4**, allows to decide *in advance*, prior to any experimental analysis, which KF implementation is best suited for the problem under consideration. This makes very fast decisions possible in order to guarantee reliability, and therefore is of extreme interest in the analysis of KF algorithms that have to operate real-time. A result that is very hard to obtain with the very time consuming classical method of simulation used so far. Furthermore, this analysis can in a very straightforward way be applied to investigate the reliability of RLS methods.

The theoretical results, summarized above, directly or indirectly *benefited* from methods used in *numerical analysis.* Therefore, the title "A new class of algorithms in linear system theory" can be interpreted more globally as being a **new contribution** to the (recent) interdisciplinary approach between linear system theory and numerical analysis.

In this context, the new class of algorithms is applied to the *realization* of real-time aircraft model identification. More precisely, the theoretical foundations of **chapters 2 and 4** have enabled the *reliable estimation* of:

1. *The aircraft's flight-path*
2. *The stability and control derivatives*, which in their turn determine in a straightforward way:
   a. the *aircraft handling qualities*
   b. the *aircraft performances*
   c. a *linear state space model* of the aircraft with the same input and output signals as available to the pilot to control the aircraft, namely the control surface deflections and the barometric signals respectively.

Furthermore, and perhaps more importantly in the contribution towards a more economical and safe flight, these practical results have been obtained in a *particular manner*, characterized by the following three attributes of the developed computational schemes:

1. they are *self-tuning* and guarantee accurate processing of the algorithms while taking into account several *time varying* effects, which would normally inacceptably deteriorate the estimation results.
2. they only make use of measurements supplied by a *standard instrumentation* system on the present and (certainly) on the near future civil aircraft's
3. they operate completely *independently*, without making use of additional units that perform a similar task to correct for possible estimation errors. Such as is e.g. the case for the present Inertial Navigation Systems where flight-path reconstruction errors are periodically corrected by Radio Navigation Systems.

Therefore, this summary makes it clear that the presented computational schemes possess many perspectives in the near future, what can be the subject for further research. This is partially elucidated in the next section.

7.3. Suggestions for further research

The results presented in this thesis certainly do not claim to have solved the real-time aircraft model identification problem. This is mainly because no practical evaluation is performed on a flight simulator or in real flight tests. However, the very promising results with the linear computer simulation of the DHC-2 Beaver aircraft have shown that through the developed techniques, the two-step identification method can strongly be considered for practical evaluation. This is then also one of the main areas to continue research. More concretely, we can split up this research in the following three parts

1. The first part concentrates on the refinement of the developed algorithms. The modules that require a Singular Value Decomposition (SVD), can be made much more efficient by the use of recursive SVD updating techniques [15].
   Secondly, the present form of the two-step procedure does not take into account *atmospheric turbulence*. The inclusion of this effect

   can be done by adding a module to the flight-path reconstruction algorithm that identifies the error statistics of the noise on the inertial measurements. This can be done with a technique described in [124], or based on the use of the more reliable SVD-concept directly operating on the inertial measurements. Furthermore, the loss of accuracy of the reconstructed flight-path can make it preferable to use the Total Least Squares method [34] for obtaining more accurate aerodynamic model estimates.
2. In the second part, a complete test program can be carried out to evaluate the performance of the two-step procedure as a *real-time aircraft identification method.* This can be performed by iron bird tests, followed by real flight tests. These tests may highlight possible deficiencies of the proposed algorithms, which will then require modifications of the original approach. This part will answer the question, whether this method can be used to reduce the very expensive flight-test analysis time to certify aircraft prototypes or aircraft modifications.
3. A third part can evaluate the "spinoff" of the on-line availability of the estimated aerodynamic model parameters of the aircraft while taking into account time variations in the acting aircraft operating circumstances. First, in the design of aircraft control systems. The two-step procedure can be easily extended by a control computing module, based on optimal control theory [50]. This will make *adaptive aircraft control* an achievable goal. Secondly, the impact of the estimated aircraft parameters on *Flight Management Systems (FMS)* can be evaluated more deeply than in [102].

A *second main research area* is the continuation of the theoretical analysis of algorithms in linear system theory. **Condensed forms** can be used to derive new algorithms which combine efficiency and reliability, for other problems in linear system theory. Here two further extensions of these ideas are obvious:

1. System theoretical problems for *generalized state space systems*, such as e.g. the generalized KF algorithm as presented by C.C. Paige in [81]. This general representation has the advantage to allow for singular covariance matrices Q_k, R_k, $P_{k|k-1}$ and even for singular information matrices $P_{k|k}^{-1}$, thereby avoiding any use of *generalized inverses*. The main disadvantage however is that it makes the algorithm very *inefficient*. Maybe this can be overcome

by condensed forms.

2. The problem of computing the *smoothed* or *predicted* filter estimates.

Another interesting topic in the continuation of this research is the extension of the theoretical error analysis to other specific designed KF implementations, such as e.g. the UDU' factorization implementation [8], which is only applicable for sequential processing [1] of the measurements.

Appendix A.

Linearization of the aircraft's equations of motion

A.1. Outline of the linearization procedure

The set of *non-linear* differential equations (1.5-1.7) in chapter 1 represent the aircraft's motion over its whole flight envelope. In this appendix we shall compute an approximate system model to (1.5-1.7) which is only valid within a restricted domain of the aircraft's flight envelope. Mathematically, the latter is represented by the defined state space. We will assume that this approximation is given by a set of *linear* differential equations. Therefore, equations (1.5-1.7) will be linearized. The resulted set of equations then describe the *perturbed* state deviations from a *nominal trajectory*, which we will see has to be an equilibrium point of equations (1.5-1.7).

Recalling the equations of motion (1.5-1.7) in compact matrix notation:

$$\dot{x} = f(x, u, t) \tag{A.1}$$

In combination with the definition of the initial conditions and the output equations,

$$x_{t_T} = x_T \tag{A.2}$$

and

$$y = h(x, u, t) \tag{A.3}$$

this completely defines the system description.
We furthermore assume that the considered aircraft dynamics are *station-*

ary, what is reflected by the fact that f and h in (A.1) and (A.3) do not change significantly from their initial values at t_T. Thus we may set,

$$\begin{cases} \dot{x} &= f(x, u) \\ y &= h(x, u) \end{cases} \tag{A.4}$$

The assumption of small perturbations may be reflected by taking x and u to be always close to their reference values x_T, u_T and these values are furthermore an equilibrium point of (A.4), given as:

$$f(x_T, u_T) = 0 \tag{A.5}$$

This defines our "nominal trajectory".
Now, if x and u are "close" to x_T and u_T they may be written as $x = x_T + \delta x$, $u = u_T + \delta u$ and these substituted in (A.4). The fact that δx and δu are small is now used to motivate an expansion of (A.4) about x_T and u_T and to suggest that only terms in the first power of the small quantities δx and δu need to be retained. We thus have a vector equation and need the expansion of a vector-valued function of a vector variable,

$$\frac{d}{dt}(x_T + \delta x) = f(x_T + \delta x, u_T + \delta u) \tag{A.6}$$

It is simply to verify that (A.6) can be written as: [1]

$$\delta\dot{x} = f(x_T, u_T) + f_x(x_T, u_T)\delta x + f_u(x_T, u_T)\delta u + \ldots \tag{A.7}$$

where f_x and f_u are the Jacobi matrices of the vector function f with respect to the state vector x and input u respectively. The rows of these Jacobi matrices are composed of *gradients* of the different scalar components of f, denoted by f_i, to x and u. The gradient for example of the first component of f, i.e. f_1, with respect to the vector x is given by:

$$f_{1x} = \left(\frac{\partial f_1}{\partial x_1}, \ldots, \frac{\partial f_1}{\partial x_n}\right) \tag{A.8}$$

With this notation, the Jacobi matrices in (A.7) are given by,

$$f_x = \begin{pmatrix} f_{1x} \\ \cdot \\ \cdot \\ \cdot \\ f_{nx} \end{pmatrix} \quad ; \quad f_u = \begin{pmatrix} f_{1u} \\ \cdot \\ \cdot \\ \cdot \\ f_{nu} \end{pmatrix} \tag{A.9}$$

[1] Note that $\frac{d(x_T)}{dt} = 0$ because our "reference trajectory" x_T is a constant here.

Now, to return to (A.7), we note that by (A.5) we chose x_T, u_T to be an equilibrium point, so the first term on the right hand side of (A.7) is zero, and since the higher powers of the "small" signals δx and δu are neglected, we are led to the approximation:

$$\delta\dot{x} \approx f_x(x_T, u_T)\delta x + f_u(x_T, u_T)\delta u \qquad (A.10)$$

Similarly, for the output equations in (A.4) we have:

$$y = h_x(x_T, u_T)\delta x + h_u(x_T, u_T)\delta u \qquad (A.11)$$

But now the notation is overly clumsy and we drop the δ-part of δx and δu and define the constant matrices:

$$A = f_x(x_T, u_T) \quad B = f_u(x_T, u_T)$$
$$C_x = h_x(x_T, u_T) \quad C_u = h_u(x_T, u_T)$$

to obtain the form we used in chapter 1, namely:

$$\begin{cases} \dot{x} &= Ax + Bu \\ y &= C_x x + C_u u \end{cases} \qquad (A.12)$$

A.2. Linear state equations of the symmetric aircraft motion

The procedure of section A.1 is applied to the equations of motion (1.5-1.7) in the Body fixed reference frame. This particular reference frame is defined in the following definition.

Definition A.1: The aircraft's body fixed reference frame F_B, pictured in fig. 1.6, is a right-handed system $O_B X_B Y_B Z_B$ of orthogonal axes. The origin O_B of F_B lies in the aircraft's center of gravity (c.g). The $X_B O_B Z_B$-plane coincides with the aircraft's plane of symmetry. The positive Y_B axis points to the right. The positive Z_B axis points downwards.

The nominal trajectory will be defined by a stationary, rectilinear, symmetrical flight. The state, input vector and aerodynamic forces are thus given by:

state:

$$\begin{cases} u &= u_T &+ \delta u \\ w &= w_T &+ \delta w \\ \theta &= \theta_T &+ \delta\theta \\ q &= &\delta q \\ \Delta h &= \Delta h_T &+ \delta\Delta h \end{cases}$$

input:

$$\delta_e = \delta_{e_T} + \delta\delta_e$$

aerodynamic forces and moment:

$$\begin{cases} X &= X_T &+ \delta X \\ Z &= Z_T &+ \delta Z \\ M &= &\delta M \end{cases}$$

where these quantities are defined in table 1.1.
Substitution in (A.10), with the vector function f of (1.5-1.7), leads to,

$$\delta\dot{u} = \frac{\delta X}{m} - g.cos.\theta_T.\delta\theta - \delta q.w_T \qquad (A.13)$$

$$\delta\dot{w} = \frac{\delta Z}{m} - g.sin.\theta_T.\delta\theta - \delta q.u_T \qquad (A.14)$$

$$\delta\dot{\theta} = \delta q \qquad (A.15)$$

$$\delta\dot{q} = \frac{\delta M}{mk_y^2} \qquad (A.16)$$

$$\delta\dot{\Delta h} = u_T.cos\theta_T.\delta\theta + sin\theta_T.\delta u + w_T.sin\theta_T.\delta\theta - cos\theta_T.\delta w \qquad (A.17)$$

Because the deviations $\delta X, \delta Z$ and δM of the aerodynamic forces and moments are functions of the state and input variables, they can be expressed as:

$$\begin{cases} \delta X &= \sum_j \frac{\partial X}{\partial x_j}.\delta x_j &+ \sum_k \frac{\partial X}{\partial u_k}.\delta u_k \\ \delta Z &= \sum_j \frac{\partial Z}{\partial x_j}.\delta x_j &+ \sum_k \frac{\partial Z}{\partial u_k}.\delta u_k \\ \delta M &= \sum_j \frac{\partial M}{\partial x_j}.\delta x_j &+ \sum_k \frac{\partial M}{\partial u_k}.\delta u_k \end{cases}$$

We shall assume that X, Z and M (only) depend on u, w, q and δ_e, so that the above set of equations can be written as:

$$\delta X = X_u.\delta u + X_w.\delta w + X_q.\delta q + X_{\delta_e}.\delta\delta_e \qquad (A.18)$$

$$\delta Z = Z_u.\delta u + Z_w.\delta w + Z_q.\delta q + Z_{\delta_e}.\delta\delta_e \qquad (A.19)$$

$$\delta M = M_u.\delta u + M_w.\delta w + M_q.\delta q + M_{\delta_e}.\delta\delta_e \qquad (A.20)$$

Equations (A.18-A.20) are generally expressed in dimensionless form, using the relations:

$$\delta X = \delta C_X \frac{1}{2}\rho_T V_T^2 S \qquad (A.21)$$

$$\delta X = \delta C_Z \frac{1}{2}\rho_T V_T^2 S \qquad (A.22)$$

$$\delta X = \delta C_m \frac{1}{2}\rho_T V_T^2 S c \qquad (A.23)$$

with ρ the density of the air and S the wing surface of the aircraft.
Using equations (A.21-23), equations (A.18-A.20) become:

$$\delta C_X = \frac{X_u}{\frac{1}{2}\rho_T V_T S}\frac{\delta u}{V_T} + \frac{X_w}{\frac{1}{2}\rho_T V_T S}\frac{\delta w}{V_T} + \frac{X_q}{\frac{1}{2}\rho_T V_T S c}\frac{\delta qc}{V_T} + \frac{X_{\delta_e}}{\frac{1}{2}\rho_T V_T^2 S}\delta\delta_e \qquad (A.24)$$

which may also be written as,

$$\delta C_X = C_{X_u}\frac{\delta u}{V_T} + C_{X_\alpha}\frac{\delta w}{V_T} + C_{X_q}\frac{\delta qc}{V_T} + C_{X_\delta}\delta\delta_e \qquad (A.25)$$

Similarly,

$$\delta C_Z = C_{Z_u}\frac{\delta u}{V_T} + C_{Z_\alpha}\frac{\delta w}{V_T} + C_{Z_q}\frac{\delta qc}{V_T} + C_{Z_\delta}\delta\delta_e \qquad (A.26)$$

$$\delta C_m = C_{m_u}\frac{\delta u}{V_T} + C_{m_\alpha}\frac{\delta w}{V_T} + C_{m_q}\frac{\delta qc}{V_T} + C_{m_\delta}\delta\delta_e \qquad (A.27)$$

Equations (A.24-A.26) are called the aerodynamical model equations, summarized in matrix notation by equations (1.10). If we now additionally denote,

$$\mu_c = \frac{m}{\rho S c}$$

the linear set of differential equations becomes, (where we again neglected the δ notation):

$$\dot{x} = Ax + Bu \qquad (A.28)$$

with

$$x = (\frac{u}{V_T},\ \frac{w}{V_T},\ \theta,\ \frac{qc}{V_T},\ \Delta h)' \quad and \quad u = (\delta_e)$$

and

$$A = \frac{V_T}{c}\begin{pmatrix} \frac{C_{X_u}}{2\mu_c} & \frac{C_{X_\alpha}}{2\mu_c} & -\frac{c}{V_T^2}g\cos\theta_T & \frac{C_{X_q}}{2\mu_c} & 0 \\ \frac{C_{Z_u}}{2\mu_c} & \frac{C_{Z_\alpha}}{2\mu_c} & -\frac{c}{V_T^2}g\sin\theta_T & 1+\frac{C_{X_q}}{2\mu_c} & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \frac{C_{m_u}}{2\mu_c k_y^2} & \frac{C_{m_\alpha}}{2\mu_c k_y^2} & 0 & \frac{C_{m_q}}{2\mu_c k_y^2} & 0 \\ c.\sin\theta_T & -c.\cos\theta_T & c.(\cos\theta_T\ \ \sin\theta_T).\begin{pmatrix}\frac{u}{V_T}\\ \frac{w}{V_T}\end{pmatrix} & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} \frac{C_{X_\delta}}{2\mu_c} \\ \frac{C_{Z_\delta}}{2\mu_c} \\ 0 \\ \frac{C_{m_\delta}}{2\mu_c k_y^2} \\ 0 \end{pmatrix}$$

For a number of applications, the use of the air-trajectory frame is interesting. This is defined as follows.

Definition A.2: The aircraft's air-trajectory frame of reference F_S is also an aircraft-fixed right-hand system $O_S X_S Y_S Z_S$ of orthogonal axes. The origin O_S of F_S lies in the aircraft's center of gravity (c.g.). The X_S axis lies in the $X_B O_B Z_B$-plane of the aircraft's body frame of reference F_B and points forward. In undisturbed steady symmetric flight the X_S axis of F_S coincides with the airspeed of the c.g. The positive Y_S axis points to the right. The positive Z_S axis points downwards.

Using this frame of reference, the values of the state in a stationary point

become:

$$\begin{cases} u_T &= V_T \\ w_T &= 0 \\ \theta_T &= \gamma_T \\ \alpha_T &= 0 \\ q_T &= 0 \\ \Delta h_T &= 0 \end{cases} \qquad (A.29)$$

And the linear state equations become:

$$\dot{x} = Ax + Bu \qquad (A.30)$$

with

$$x = (\frac{u}{V_T},\ \alpha,\ \theta,\ \frac{qc}{V_T},\ \Delta h)' \quad and \quad u = (\delta_e)$$

and

$$A = \frac{V_T}{c} \begin{pmatrix} \frac{C_{X_u}}{2\mu_c} & \frac{C_{X_\alpha}}{2\mu_c} & -\frac{c}{V_T^2} g cos\gamma_T & \frac{C_{X_q}}{2\mu_c} & 0 \\ \frac{C_{Z_u}}{2\mu_c} & \frac{C_{Z_\alpha}}{2\mu_c} & -\frac{c}{V_T^2} g sin\gamma_T & 1+\frac{C_{X_q}}{2\mu_c} & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \frac{C_{m_u}}{2\mu_c k_y^2} & \frac{C_{m_\alpha}}{2\mu_c k_y^2} & 0 & \frac{C_{m_q}}{2\mu_c k_y^2} & 0 \\ c.sin\gamma_T & -c.cos\gamma_T & c.cos\gamma_T & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} \frac{C_{X_\delta}}{2\mu_c} \\ \frac{C_{Z_\delta}}{2\mu_c} \\ 0 \\ \frac{C_{m_\delta}}{2\mu_c k_y^2} \\ 0 \end{pmatrix}$$

A.3. Linear observation model

Analogue to above outlined operation, the observation model in (A.12) for the following output signals:

1. the airspeed V,
2. the altitude variation Δh,
3. the specific aerodynamic forces A_X and A_Z [2],
4. the rate of pitch q

becomes,

$$\begin{pmatrix} u \\ \Delta h \\ A_X \\ A_Z \\ q \end{pmatrix} = \begin{pmatrix} V_T & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \frac{V_T^2}{2\mu_c c} C_{X_u} & \frac{V_T^2}{2\mu_c c} C_{X_\alpha} & 0 & \frac{V_T^2}{2\mu_c c} C_{X_q} & 0 \\ \frac{V_T^2}{2\mu_c c} C_{Z_u} & \frac{V_T^2}{2\mu_c c} C_{Z_\alpha} & 0 & \frac{V_T^2}{2\mu_c c} C_{Z_q} & 0 \\ 0 & 0 & 0 & \frac{V_T}{c} & 0 \end{pmatrix} \begin{pmatrix} \frac{u}{V_T} \\ \alpha \\ \theta \\ \frac{qc}{V_T} \\ \Delta h \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \frac{V_T^2}{2\mu_c c} C_{X_\delta} \\ \frac{V_T^2}{2\mu_c c} C_{Z_\delta} \\ 0 \end{pmatrix} \delta_e \qquad (A.31)$$

[2] defined as $\frac{X}{m}$ and $\frac{Z}{m}$ respectively.

Appendix B.

The Skew Householder transformation

The Householder transformation matrix H in (2.3) of chapter 2, can be written completely equivalently as:

$$H = (I - 2\frac{tt'}{t't}) \qquad (B.1)$$

If we denote $\frac{t}{\|t\|_2}$ by u, than we obtain:

$$H = (I - 2uu') \text{ with } \|u\|_2 = 1 \qquad (B.2)$$

This transformation can be constructed now to reflect a vector towards a chosen k-th basis vector of the natural orthogonal space with the same 2-norm, denoted by $\|.\|_2$. This is discussed in chapter 2, section 2.3.

However for the skew Householder transformation use is made of the Σ-norm, which is defined as follows.

Definition B.1: The Σ-norm of a vector x is the product:

$$x'\Sigma x \qquad (B.3)$$

for some particular signature matrix Σ ($\Sigma^2 = I$) and is denoted as $\|x\|_\Sigma$.

Now to reflect a vector to a chosen k-th basis vector of the natural orthogonal space with the same Σ-norm, the transformation matrix H_Σ is chosen as follows:

$$H_\Sigma = I - 2uu'\Sigma \text{ with } \|u\|_\Sigma = 1 \qquad (B.4)$$

That this is the correct solution will be verified in the sequel. First, we will verify whether the defined transformation H_Σ is Σ-orthogonal. Therefore, the product $H_\Sigma \Sigma H'_\Sigma$ has to be Σ. Using (B.4) this product becomes:

$$\begin{aligned} H_\Sigma \Sigma H'_\Sigma &= (I - 2uu'\Sigma)\Sigma(I - 2\Sigma uu') \\ &= \Sigma - 2uu' - 2uu' + 4uu'\Sigma uu' \end{aligned}$$

Since $\|u\|_\Sigma = 1$, the proposed transformation matrix is indeed Σ-orthogonal. Furthermore, it can easily be verified in a similar way that $H_\Sigma^2 = I$, as is the case for the classical Householder transformation in (B.1).

Secondly, we will now construct H_Σ based on a vector x, transforming it to the first basis vector e_1. The construction of the transformation H_Σ goes as follows. Define a vector w, accordingly:

$$w = x + sign(x_1)\|x\|_\Sigma e_1 \qquad (B.5)$$

with $sign(x_1)$ defined in chapter 2. In the subsequent derivation, the $sign(x_1)$ will assumed to be 1. The Σ-norm of w is:

$$\|w\|_\Sigma^2 = 2(\|x\|_\Sigma^2 + \|x\|_\Sigma x_1) \neq 0 \qquad (B.6)$$

Now u in (B.4) is choosen as $\frac{w}{\|w\|_\Sigma}$. This choice is verified by applying H_Σ to the vector x, where we make use of (B.5):

$$\begin{aligned} H_\Sigma x &= (I - 2uu'\Sigma)x \\ &= x - 2\frac{ww'\Sigma x}{\|w\|_\Sigma^2} \\ &= x - \frac{(x + \|x\|_\Sigma e_1)\|w\|_\Sigma^2}{\|w\|_\Sigma^2} \\ &= -\|x\|_\Sigma e_1 \qquad (B.7) \end{aligned}$$

what was desired.

Appendix C.

Error propagation in some basic linear algebra problems

Here we briefly recall the propagation of rounding errors in some basic problems in linear algebra. The norm used is the 2-norm.

Let the matrix-vector pair (A, b) be known with relative precision δ_a and δ_b, respectively :

$$\delta_a = \|\delta A\|/\|A\| \ , \ \delta_b = \|\delta b\|/\|b\|$$

then we have the following lemma (assuming A to be invertible).

Lemma C.1 [123]

The errors on the products $A.b$ and $A^{-1}.b$ can be bounded by

$$\|\overline{(A.b)} - (A.b)\| \leq (\delta_a + \delta_b).\|A\|.\|b\| + O(\delta^2)$$

$$\|\overline{(A^{-1}.b)} - (A^{-1}.b)\| \leq \delta_a.\kappa(A).\|A^{-1}.b\| + \delta_b.\|A^{-1}\|.\|b\| + O(\delta^2)$$

When the errors δ_a and δ_b are the backward errors of the above problem solved on a computer with machine precision ϵ, then the above bounds are reasonably well approximated by :

$$\|\overline{(A.b)} - (A.b)\| \leq \epsilon_1.\|A\|.\|b\| \approx \epsilon_2.\|A.b\|$$

$$\|\overline{(A^{-1}.b)} - (A^{-1}.b)\| \leq \epsilon_3.\kappa(A).\|A^{-1}.b\|$$

where all ϵ_i are of the order of ϵ.

■

The above approximation implies that no serious cancellations occur in the product $A.b$, which in general is a reasonable assumption.

Let A now be a m×n matrix of rank $m < n$ and transform the compound matrix $[\ A \mid b\ ]$ by a unitary transformation Q as follows

$$Q.[\ A \mid b\ ] = \begin{pmatrix} A_1 & \mid & b_1 \\ 0 & \mid & b_2 \end{pmatrix}$$

where A_1 is now invertible. Then we have the following lemma (where $.^+$ denotes the generalized inverse of a matrix).

Lemma C.2 [88]

The errors on the least squares solution $A^+.b$ and the residual b_2 can be bounded by

$$\|\overline{(A^+.b)} - (A^+.b)\| \leq \delta_a.\{\kappa(A).\|A^+.b\| + \kappa(A).\|A^+\|.\|b_2\|\} + \delta_b.\|A^+\|.\|b\| + O(\delta^2)$$

$$\|\overline{(b_2)} - (b_2)\| \leq \delta_a.\kappa(A).\|b\| + \delta_b.\|b\| + O(\delta^2)$$

When the errors δ_a and δ_b are the backward errors of the above problems solved on a computer with machine precision ϵ, then they are both of the order of ϵ and the above bounds are reasonably well approximated by :

$$\|\overline{(A^+.b)} - (A^+.b)\| \leq \epsilon_4.\{\kappa(A).\|A^+.b\| + \kappa(A).\|A^+\|.\|b_2\| + \|A^+\|.\|b_2\|/cos\phi\}$$

$$\|\overline{(b_2)} - (b_2)\| \leq \epsilon_5.\{1 + \kappa(A)\}.\|b_2\|/cos\phi$$

where all ϵ_i are of the order of ϵ and $cos\phi = \|b_2\|/\|b\|$.

■

We terminate with perturbation bounds on the QR-factorization of a matrix A.

Lemma C.3 [93]

Let $A = Q.R$, where A has full column rank n, $Q'.Q = I_n$ and R is upper triangular. Then for a small perturbation $\overline{A}$ of A there exist perturbations $\overline{Q}$ and $\overline{R}$ of the factors, such that

$$\overline{A} = \overline{Q}.\overline{R}$$

and

$$(A - \overline{A}) = Q.(R - \overline{R}) + \Delta$$

with

$$\|\Delta\| \leq [\delta_a.\kappa(A)]^2.\|A\|$$

When the error δ_a is the backward error of the above decomposition solved on a computer with machine precision ϵ, then it is of the order of ϵ and the above bound becomes

$$\|\Delta\| \leq \epsilon_6^2.\kappa(A)^2.\|A\|$$

A similar result can also be found for a "skew decomposition", i.e. where $Q'.\Sigma.Q = \Sigma$, for some signature matrix Σ.

■

Although all these bounds are written for the 2-norm, they also hold for several other norms, up to a constant which is close to 1 and can therefore be absorbed in the ϵ_i. This is e.g. important when deriving the above bounds for a matrix B instead of a vector b. This is done by using the bounds for each column b_i of the matrix B and combining these bounds into a bound involving the norm of B, for which in this case the Frobenius norm is a natural choice [92]. These mixed bounds (as far as norms are concerned) can then again be formulated in terms of one norm only, by again adapting the ϵ_i appropriately.

Appendix D.

Samenvatting

D.1. Inleiding

Het onderzoek uitgevoerd in dit doctoraat is duidelijk gestimuleerd door de nood die de dag van vandaag bestaat om de vluchtuitvoering, en dan vooral deze van civiele verkeersvliegtuigen, meer economisch en meer veilig te maken. Door de snelle ontwikkeling in de micro-electronica heeft men getracht dit doel na te streven door de automatizatie en integratie van verschillende vliegtuigsystemen door gebruik te maken van de *digitale technologie.* De ervaring van verschillende vliegtuigconstructeurs heeft nochtans aangetoond dat digitale technologie *alleen* niet voldoende is om de uitbatingskosten te verminderen en gelijktijdig de vliegveiligheid te verzekeren. De laatste tijd is duidelijk geworden dat een *verdere verbetering* van de vluchtuitvoering maar bereikt kan worden door, op een intellegente manier, gebruik te maken van *systeemtheorie.* Meer bepaald wordt aangetoond in het eerste hoofdstuk dat een verdergaande optimalisering van de vlucht mogelijk is indien men beschikt over een mathematisch *toestandsmodel* van het vliegtuig dat zich automatisch aanpast aan de tijdsvarierende vluchtcondities (zoals atmosferische veranderingen, slijtage van de motoren, enz.). Het *ontwikkelen van algoritmen* om dit idee te realizeren, d.w.z. om "online" vliegtuigmodel identificatie mogelijk te maken, is het *hoofddoel* van dit werk. In de navolgende secties geven we een kort overzicht van de verkregen onderzoeksresultaten.

D.2. Het (her-) formuleren van het vliegtuig identificatie probleem

Het zelfinstellend identificeren van een toestandsmodel van het vliegtuig heeft veel aandacht gekregen het laatste decennium. Tot nog toe zijn er reeds verschillende onderzoeksprojecten geweest om dit probleem op te lossen. Een voorbeeld hiervan is een onderzoeksprogramma van de NASA [13] dat ondanks het gebruik van zeer krachtige computers vroegtijdig en zonder succes beeindigd werd. Een van de hoofdoorzaken van dit falen is zeker het doorvoeren van vele vereenvoudigingen om het *niet lineaire* identificatieprobleem op te lossen. Deze vereenvoudigingen zijn niet noodzakelijk voor de *tweetrapsidentificatie methode* [30] onderzocht in dit werk. De methode herformuleert het niet lineaire identificatie probleem in *twee lineaire schattingsproblemen.* Deze problemen worden respectievelijk aangeduid als het *vliegbaanreconstructie probleem* en het *aerodynamisch model identificatie probleem.* Door deze probleemsplitsing kan de vliegbaan gereconstrueerd worden door zgn. *lineair te filteren.* De kennis van de vliegbaan op haar beurt maakt van het aerodynamisch model, dat niets anders is dan een mathematische beschrijving van de aerodynamische krachten en momenten die op het vliegtuig werken, een *lineair regressie model.* Hierdoor kunnen de onbekende coefficienten, namelijk de zgn. stabiliteits en besturings afgeleiden, bepaald worden m.b.v. de kleinste kwadraten methode. De eenvoudige vorm van beide bovenstaande deelproblemen maken de tweetraps identificatie methode zeer aantrekkelijk om toepasbaar te maken in "real-time".

D.3. Concrete doelstellingen van het doctoraat

Het real-time toepasbaar maken van de tweetrapsidentificatie methode vereist onderzoek in de volgende drie hoofdgebieden:

- de digitale technologie
- de instrumentatie technologie
- het ontwerp van rekenmethoden

Het onderzoek in deze thesis concentreert zich op het derde aspect. Meer nauwkeurig omschreven, zullen rekenmethoden ontworpen worden voor het

oplossen van beide deelproblemen van de tweetraps identificatie methode. Daar het noodzakelijk is dat deze algoritmen gegevens verwerken tijdens de vluchtuitvoering, d.w.z. in "real-time", dienen deze te voldoen aan de volgende vaak *tegenstrijdige* eisen:

1. Algebraische efficientie: Voor de huidige generatie vluchtwaardige computers, blijft dit algoritmisch aspect zeker zeer belangrijk. Om dit met een voorbeeld toe te lichten nemen we de ROLM1602B vluchtwaardige computer van het NLR (het Nederlands instituur voor Lucht- en Ruimtevaart onderzoek). Het uitvoeren van een optelling in *enkele precisie* m.b.v. "floating-point" routines vraagt $300\mu s$. Wanneer we ditzelfde in *dubbele precisie* willen uitvoeren is echter $5\times$ zoveel tijd noodzakelijk. De ontwikkelingen in de microelectronica zullen deze eis minder streng maken.

2. Algoritmische betrouwbaarheid en nauwkeurigheid: Deze aspecten hangen nauw samen. Bij de "real-time" verwerking van metingen dienen nauwkeurige voorzorgen genomen te worden i.v.m. de aanwezigheid van meetfouten. De grootte van deze fouten hangt natuurlijk direct af van de precisie van het gebruikte meetsysteem. Niettemin, kan de manipulatie van de gegevens door de gekozen of ontworpen algoritmische implementatie het gewenste resultaat zeer onnauwkeurig maken. Dit effect dient binnen zekere grenzen gehouden te worden of beter nog, dient het geminimaliseerd te worden. Deze aspecten zullen belangrijk blijven, onafhankelijk van de ontwikkelingen in de digitale technologie of de instrumentatie technologie. Hierdoor is het noodzakelijk om:

   - *betrouwbare programmatuur* te ontwikkelen en te gebruiken. Zo kan de voortplanting van afrondingsfouten en andere foutenbronnen (zoals discretizatie fouten, model fouten, enz.) in de berekeningen *geminimaliseerd* worden. De nood aan dit aspect wordt nog geaccentueerd wanneer de voorkeur gegeven wordt aan enkele precisie berekeningen om de efficientie te vergroten.
   - de verschillende aanwezige meetfouten in het gebruikte meetsysteem nauwkeurig te modelleren en te behandelen. Een gelijkaardige benadering is bovendien ook noodzakelijk om de tijdsvariaties, die zich voordoen tijdens de vluchtuitvoering, in de aerodynamische eigenschappen van het vliegtuig *automatisch* te behandelen.

Deze algoritmische eisen zullen onderzocht worden voornamelijk voor het lineair filter en het kleinste kwadraten schattings deelprobleem van de tweetraps identificatie methode. De resultaten van dit onderzoek zullen getest worden op basis van digitaal gesimuleerde vliegproefgegevens, waarbij gebruik gemaakt wordt van een model van een DHC-2 Beaver vliegtuig. Dit laatste is het laboratorium vliegtuig van de Technische Hogeschool te Delft.

D.4. Analyse van het vliegbaanreconstructie probleem

In het afleiden van het systeem model dat aan de basis ligt van het lineaire filter hebben we een aantal belangrijke observaties gemaakt. Ten eerste, laten deze observaties toe het filter probleem meteen *discreet* te stellen en verder laten ze toe verschillende systeem eigenschappen voor alle mogelijke referentie vliegcondities *analytisch* te verifieren. De kennis van deze eigenschappen zoals observeerbaarheid, controleerbaarheid en de eigenwaarde structuur van de systeem transitie matrix zijn zeer belangrijk in de beoordeling van de prestaties van het lineaire filter. Ten tweede, is op een eenvoudige manier aangetoond dat het systeem model voorgesteld kan worden in de "Schur" vorm of de "Observer-Hessenberg" vorm. Een interessante observatie die hier gemaakt werd is dat deze vormen passen in een meer algemene klasse van systeem structuren uit de numerieke analyse. Namelijk deze vormen van toestandsmodellen $\{A, B, C, D\}$ die gekarakteriseerd worden door:

a. een speciale structuur van de verschillende matrices die veel *nullen* bevat

b. de manier waarop deze speciale structuur bekomen is, namelijk zonder verlies van nauwkeurigheid.

In numerieke analyse wordt deze klasse van matrix structuren aangeduid als "gecondenseerde vormen". Het doordacht gebruik van deze vormen kan leiden tot algoritmen of implementaties van algoritmen die de eisen van betrouwbaarheid en efficientie *combineren*. In hoofdstuk 2 zijn deze vormen dieper geanalyseerd in het ontwerp van algoritmen voor een brede klasse van systeem theoretische problemen. Verder in deze thesis zullen zij bovendien gebruikt worden om de algoritmen in het vliegtuig model identificatie probleem gelijktijdig efficient en betrouwbaar te maken.

D.4.1. Het gebruik van gecondenseerde vormen

Kanonieke structuren van de systeemmatrices $\{A, B, C, D\}$ worden reeds geruime tijd aangewend om systeem theoretische problemen efficient en/of elegant op te lossen, zie bijvoorbeeld in [67] en [18]. Niettemin, bleken de meeste van deze voorgestelde technieken enkel van toepassing te zijn op academische voorbeelden. De hoofdoorzaak was dat de theoretische verwachtingen niet uitkwamen voor praktische problemen. En dit kon vaak toegeschreven worden aan de gevoeligheid van de ontworpen rekenmethoden voor afrondingsfouten en/of meetfouten. In de numerieke analyse heeft aandacht aan dit aspect uitgewezen dat gebruik van orthogonale gelijkvormigheids transformaties dergelijke moeilijkheden oplost, die bovendien ook leiden tot "gecondenseerde" structuren die elegante oplossingen toelaten. In [123] is dit bijvoorbeeld mooi aangetoond door J. Wilkinson voor het eigenwaarde bepalings probleem.

Gecondenseerde vormen worden echter reeds lang gebruikt in de numerieke analyse. Dit heeft aanleiding gegeven tot een aantal *conceptuele basis technieken*, waardoor het gebruik van gecondenseerde vormen beschouwd kan worden als een methodiek om algoritmen te ontwerpen.

In hoofdstuk 2 is het nut van deze methodologie onderzocht voor problemen uit de systeem theorie. Dit is gerealiseerd door ten eerste een overzicht te geven van reeds bestaande algoritmen, afgeleid op basis van gecondenseerde vormen, en ten tweede door aan dit overzicht verschillende uitbreidingen toe te voegen. Kort samengevat werden efficiente en betrouwbare algoritmen afgeleid, respectievelijk voorgesteld, voor de volgende problemen uit de systeem theorie:

- Lineaire toestands terugkoppeling
- Oplossing van de Lyapunov vergelijking voor het discrete of continue tijdsprobleem
- Het bepalen van de overdrachtsfunctie, polen en nulpunten van een lineair systeem gegeven door middel van een toestandsbeschrijving
- Het bepalen van de frequentie responsie en systeem responsie

Het toestands reconstructie probleem m.b.v. een *Luenberger waarnemer* is uitgebreider behandeld, omdat deze techniek gebruikt kan worden voor het ontwerp van het lineair filter voor de reconstructie van de vliegbaan. Ondanks de formulering van een zeer efficient algoritme voor dit waarnemer probleem, op basis van gecondenseerde vormen, leidde de *heuristische keuze* van de observer dynamica tot *zeer onnauwkeurige* reconstructie

resultaten wanneer de metingen fouten bevatten. Zelfs wanneer deze laatste zeer klein waren.

Een zeer interessant toepassingsgebied van de gepresenteerde "algoritme ontwerp" techniek zijn de systeem theoretische problemen voor de zgn. gegeneraliseerde toestandsbeschrijving. Deze laatste kan als volgt voorgesteld worden:

$$\lambda E x = A x + B u \qquad (D.1)$$

$$y = C x + D u \qquad (D.2)$$

waarbij λ de differentie operator voorsteld in het continue geval en de verschuivings operator in het discrete geval. De matrix E in (D.1) is echter niet noodzakelijk inverteerbaar. Deze vorm van toestands beschrijving kan resulteren uit het lineairizatie van de algemene niet lineaire beschrijving [87], of rechtstreeks uit de formulering van de systeem beschrijving [120]. Door het gebruik van gecondenseerde vormen is het mogelijk bestaande algoritmen voor de klassieke toestandsbeschrijving (waarbij de E matrix in (D.1) de eenheidsmatrix is) toe te passen op de gegeneralizeerde beschrijving (D.1-D.2). Dit is aangetoond in hoofdstuk 2 voor het veralgemeende gereduceerd Luenberger waarnemer probleem.

D.4.2. Het gebruik van het Kalman filter

De heuristische keuze van de waarnemer dynamica zoals bij de Luenberger waarnemer stelt zich niet voor het zgn. *Kalman filter* (KF). Deze tweede techniek om het lineaire filter te ontwerpen voor de reconstructie van de vliegbaan bepaalt bovendien een *optimale* reconstructie, in minimum variantie zin, van de vliegbaan. In hoofdstuk 3 hebben we een duidelijke *klassificatie* aangebracht in de grote verscheidenheid van bestaande KF algoritmen. Dit steunt op een nieuwe theoretische afleiding van het KF [81]. Deze afleiding is langs de ene kant zeer instructief omdat het enkel steunt op het eenvoudige kleinste kwadraten concept en eenvoudige algebra. Langs de andere kant kan ze gebruikt kan worden als een "algemeen kookboek recept om een maaltijk te bereiden" (een lineair KF te ontwerpen) dat het "best past bij zijn eigen smaak op een bepaald ogenblik" (meest geschikt is vanuit het standpunt van efficientie en betrouwbaarheid voor een bepaald gesteld filter probleem). Deze klassificatie omvat alle bestaande KF implementaties toepasbaar voor het algemene discrete tijdsvariabele geval. Dit is

eenvoudig aangetoond door de afleiding van twee extreme filter implementaties, namelijk het "square root" covariantie filter en het "square root" informatie filter. Van deze filters kan het origineel KF, dat een recursie formuleert voor de volledige toestandscovariantie matrix of zijn inverse, veel eenvoudiger afgeleid worden, dit enkel door uit te vermenigvuldigen. De omgekeerde werkwijze die tot dusver meestal gebruikt is geworden in de literatuur is niet zo recht toe recht aan toepasbaar.

Van nu af concentreert de analyse van de "on-line" toepasbaarheid zich op het onderzoek en aanpassing van de numerieke KF implementaties om aan de hoger vermelde eisen (zie sectie D.3) tegemoet te komen. Meer specifiek zullen voor het vliegbaan reconstructie probleem drie verschillende eisen behandeld worden, samengevat in hoofdstukken 4 en 5. Dit heeft geleid tot een volledig *zelfinstellend* algoritme om dit probleem op te lossen.

D.4.3. Het zelfinstellend vliegbaan reconstructie algoritme

Een eerste onmisbare eis om een nauwkeurige vliegbaan reconstructie te garanderen is de *numerieke betrouwbaarheid* van de KF implementatie. In tegenstelling tot een *zeer uitgebreide simulatie studie* om de betrouwbaarheid van verschillende KF implementaties te vergelijken voor de grote verscheidenheid van systeem condities die zich kunnen stellen tijdens de werkelijke vlucht, is gekozen voor een *theoretische studie*. De analyse is uitgevoerd in hoofdstuk 4 voor vier verschillende KF implementaties:

- het conventionele KF (CKF)
- het Chandrasekhar "square root" filter (CSRF)
- het "square root" informatie filter (SRIF)
- het "square root" covariantie filter (SRCF)

Deze analyse heeft duidelijk gemaakt welke systeem parameters en op welke manier de voortplanting van fouten in de verschillende filters beinvloeden. Voor het CKF en het SRCF zijn deze:

- het conditiegetal van de covariantie matrix van het innovatie signaal in het KF
- de spectrale norm (of straal) van de filter toestands transitie matrix

Voor het CSRF, wordt de keuze van:

- het conditiegetal van de covariantie matrix van het innovatie signaal in het KF
- de start waarde van de toestands covariantie matrix

kritisch, en voor het SRIF zijn de relevante parameters in de numerieke betrouwbaarheid:

- het conditiegetal van de covariantie matrix van het innovatie signaal in het KF
- de spectrale norm (of straal) van de filter toestands transitie matrix
- het conditiegetal van de systeem ruis covariantie matrix
- het conditiegetal van de toestands transitie matrix
- het conditiegetal van de Choleski factor van de toestands covariantie matrix

Dit inzicht leidde tot het formuleren van een aantal *vuistregels* die toelaten om *zonder* experimentele analyse de meest betrouwbare KF implementatie te kiezen voor een gegeven probleem. Om deze keuze te maken dienen we enkel te beschikken over een toestandsmodel van het systeem, de meetfout-, systeemfout covariantiematrix en de beginwaarde van de toestandscovariantie matrix. Het basis vliegtuig reconstructie probleem is aangewend om de geldigheid van deze richtlijnen te staven. In deze simulatie studie, werden bovendien de tekortkomingen (afgeleid in de theoretisch studie) van de verschillende KF implementaties bevestigd.

1. Het CKF wordt *volledig onbetrouwbaar*. Dit is te wijten aan het verlies van symmetrie van de toestands covariantie matrix, wanneer de transitie matrix van het originele systeem model *onstabiel* is, of eigenwaarden heeft op de eenheidscirkel in het complexe vlak. Dit probleem kan voorkomen worden door het gebruik van eenvoudige trukjes, zoals bijvoorbeeld het symmetrisch maken van de toestands covariantie matrix na (of voor) elke nieuwe recursie. Dit kan bekomen worden door een eenvoudige uitbreiding van het conventionele algoritme zonder een gevoelige toename van de rekentijd. Niettemin, hebben de experimenten aangetoond dat ondanks het weer competitief worden van het CKF hierdoor, de afrondingsfouten toch altijd hoger bleven voor het CKF in vergelijking tot de beschouwde "square root" filters (SRF).
2. Het verlies van nauwkeurigheid in de berekening van de Kalman filter versterkings matrix en dus ook in de berekening van de gereconstrueerde toestand bij het CSRF is *steeds* hoger dan de andere SRF.

Onder welbepaalde omstandigheden aangegeven in de theoretische analyse wordt dit verlies aan nauwkeurigheid zelfs zeer groot. Verder is deze filter implementatie alleen toepasbaar voor het tijdsinvariabele geval. Daar het filter voor bepaalde omstandigheden zeer efficient kan werken wordt het aangeraden voor tijdsinvariante filter problemen waar de efficiente van zeer groot belang is en het verlies van nauwkeurigheid geaccepteerd kan worden.

3. In het SRIF kan de berekening van de toestands covariantie matrix zeer slecht worden indien de conditie getallen van de toestands transitie matrix, de meetruis covariantie matrix en systeemruis covariantie matrix zeer hoog worden. Voor praktische toepassingen is het mogelijk het conditie getal van beide laatste covariantie matrices te verbeteren door het schalen van de uitgangs-, respectievelijk ingangsgrootheden. Doch, het veranderen van het conditie getal van de toestands transitie matrix is veel moeilijker te bereiken, zodat dit probleem de berekeningen m.b.v. het SRIF altijd nadelig zal beinvloeden in vergelijking tot het SRCF. De berekening van de gereconstrueerde toestand m.b.v. het SRIF zal ook aan nauwkeurigheid verliezen wanneer het conditiegetal van de toestands covariantie matrix groot is.

Langs de andere kant, heeft de simulatie studie het *excellente numerieke gedrag* aangetoond van het SRCF voor de condities die zich stellen in het vliegbaan reconstructie probleem.

Ten tweede is het SRCF verder geanalyseerd om het toe te laten rekening te houden met verschillende condities die zich kunnen stellen gedurende de vlucht uitvoering en die de berekeningen nadelig zouden beinvloeden. In hoofdstuk 5, zijn de volgende problemen beschouwd:

a. de variaties in de dynamische eigenschappen van het vliegtuig door verandering in de referentie conditie

b. de aanwezigheid van nulpuntsfouten op de gebruikte traagheidssignalen

Om te voorkomen dat de vliegbaan reconstructie fouten ontoelaatbaar groot zouden worden is het originele SRCF uitgebreid met twee modules, die deze veranderingen in acht nemen op een volledig *zelfinstellende* manier. In de eerste module zijn de atmosferische gegevens over de vliegsnelheid t.o.v. de omringende lucht en de hoek van de vliegbaan gebruikt om de variaties in de dynamische vliegtuig eigenschappen te bepalen en deze te volgen. Wanneer

deze module echter geen variaties kon waarnemen, neemt de tweede module de controle van het SRCF over en gaat na of er een verandering is van de nulpuntsfouten op de traagheidssignalen. Deze twee modules verhogen de convergentie snelheid van de KF recursies, terwijl zij een *minimaal verlies van nauwkeurigheid* van de gereconstrueerde vliegbaan verzekeren. Dit is bewerkstelligd door een passende en automatische keuze van de exponentiele gewichtsfactor en door de automatische berekening van de *minimale verstoring* van de ingangs verdelings matrix van de uitgebreide toestands beschrijving, waarin de observeerbare nulpuntsfouten zijn opgenomen.

Een derde belangrijke eis van de filter implementatie is de *efficientie*. De speciale structuur van de uitgebreide toestands beschrijving maakte een eenvoudige transformatie mogelijk naar een gecondenseerde "Schur" vorm. Hierdoor is een zeer tijdsbesparende (maar zeker niet triviale) SRCF filter implementatie afgeleid. In de simulatie experimenten met het vliegbaan reconstructie probleem op de IBM 370 rekenmachine van het universitair rekencentrum van de KUL was deze implementatie 40 % sneller dan het CKF. Langs de andere kant is een eenvoudige procedure ontwikkeld, gebruik makend van de resultaten van de fouten analyse, om de convergentie van het recursieve SRCF te controleren. Hierdoor kan de werkbelasting van de computer verminderd worden gedurende "rustige" fasen in de vlucht of kan er meer efficient gebruik van gemaakt worden.

D.5. Het zelfinstellend aerodynamisch model identificatie algoritme

De nauwkeurige kennis van de vliegbaan, die gegarandeerd wordt door het aangepaste SRCF algoritme, laat toe het aerodynamisch model te bepalen, d.w.z. het schatten van de stabiliteits en besturings afgeleiden, door de eenvoudige kleinste kwadraten methode (KM).

In hoofdstuk 6 hebben we verschillende recursieve KM (RKM) onderzocht en aangepast om de bepaling van deze grootheden in "real-time" toe te laten.

Een eerste deel van deze analyse concentreerde zich op de *betrouwbaarheid* en *efficientie* van de verschillende RKM procedures. Voor de analyse van de betrouwbaarheid is een gelijkaardige studie gemaakt als de studie van de fouten voortplanting in de verschillende KF implementaties. Deze studie toonde aan dat het conventionele RKM *volledig onbetrouwbare*

schattingen produceert. Dit was weer te wijten aan het verlies van symmetrie van de fouten covariantie matrix van de geschatte parameters. Verder is bewezen in dit hoofdstuk dat dit onacceptabel gebrek van het conventioneel RKM verergerd wordt door het exponentieel vergeten van oude meetgegevens, die noodzakelijk zijn om tijdsvariaties in de stabiliteits en besturings afgeleiden te kunnen volgen. Langs de andere kant, hebben de fouten modellen ontwikkeld voor de "square-root" RKM (SRKM) schema's, aangetoond dat er geen voorkeur bestaat voor een bepaald SRKM schema, indien de *regressor vectors "persistently exciting" blijven.* Dit wel te verstaan als we enkel belang hechten aan de betrouwbaarheid van de berekeningen. De conclusies van deze theoretische foutenstudie werden daarnaast weer bevestigd door een experimentele studie met het aerodynamisch model identificatie probleem.

Het tweede deel van de analyse onderzocht de moeilijkheden die kunnen ontstaan wanneer de RKM tijdens de vlucht dient te werken. Om deze moeilijkheden het hoofd te bieden, werd een betrouwbaar RKM uitgebreid met twee extra modules:

- module I, die de "persistent excitation" conditie test
- module II, die de parameter variaties behandelt

De eerste module controleert het gehele schattings proces. Dit wordt duidelijk uit de eigenschappen die deze eerste module bezit:

- Het voorkomt dat de elementen van de fouten covariantie matrix van de geschatten parameters ontoelaatbaar groot worden, waardoor het wegslingeren naar fysisch niet accepteerbare schattingen onmogelijk wordt.
- Het verzekerd dat de RKM berekeningen uitgevoerd kunnen worden op een betrouwbare manier.
- En het bepaalt de structuur van het regressie model, d.w.z. het bepaalt die parameters waarover de regressor vectors voldoende informatie bevatten om een nieuwe parameter schatting toe te laten. De andere parameter schattingen worden tijdelijk "bevroren".

In verband met de laatste eigenschap kan opgemerkt worden dat deze module van de totale RKM procedure een beter alternatief maakt, d.w.z. meer efficient en meer betrouwbaar, dan de tot dusver gebruikte "backward-stepwise" regressie methode.

Een tweede module stelt de variaties vast in de stabiliteits en besturings afgeleiden. De beschreven procedure voor dit doel steunt op de aanwezige

kennis van de dynamische eigenschappen van het vliegtuig. Deze informatie kan zeker bekomen worden uit de aanwezige windtunnel metingen. Via een aantal sleutel parameters wordt het dan mogelijk om veranderingen in de aerodynamische eigenschappen nauwkeurig vast te stellen. Volgens deze vaststellingen, kan dan een vergeetpatroon bepaald worden om oude gegevens te vergeten.

D.6. Hoofdconclusie van het onderzoek

De nieuwe algoritmen en nieuwe implementaties van bestaande algoritmen, waarbij gebruik gemaakt is van gecondenseerde vormen, (zie hoofdstukken 2, 4 en 5) vormen zeker een **nieuwe klasse van algoritmen in de lineaire systeem theorie**. De vereniging van de algoritmische betrouwbaarheid en efficientie dat deze klasse van algoritmen kenmerkt, is een cruciale eigenschap wanneer we deze algoritmen zouden gebruiken in "real-time".

De theoretische fouten analyse uitgevoerd in hoofdstuk 4, laat toe om zonder enige experimentele analyse het KF te bepalen, dat het best geschikt is voor een bepaalde probleemstelling. Dit maakt zeer snelle beslissingen mogelijk om de betrouwbaarheid te garanderen, en daardoor is het uitermate interessant voor de analyse van KF algoritmen die moeten werken in "real-time". Dit is een resultaat dat zeer moeilijk verkregen kan worden door de zeer tijdrovende methode van simulatie die tot dus ver altijd voorgesteld werd om dit probleem te behandelen. Verder is deze analyse op een zeer eenvoudige manier toegepast om de betrouwbaarheid van RKM te onderzoeken.

De theoretische resultaten, hierboven samengevat, hebben op een directe of indirecte manier voordeel gehaald uit methoden gebruikt in de numerieke analyse. Daardoor kan de titel van dit doctoraat "Een nieuwe klasse van algoritmen in de lineaire systeem theorie" breder geinterpreteerd worden, als een **nieuwe bijdrage** tot de (recente) interdisciplinaire benadering tussen lineaire systeem theorie en numerieke analyse.

In dit licht is de nieuwe klasse van algoritmen toegepast in de realizatie van "real-time" vliegtuig model identificatie. Meer precies, hebben de theoretische grondslagen van hoofdstukken 2 en 4 de betrouwbare schatting mogelijk gemaakt van:

1. de *vliegbaan* van het vliegtuig

2. de stabiliteits en besturings afgeleiden, die op hun beurt:
    a. de *manoeuvreerbaarheid* van het vliegtuig,
    b. de *prestaties* van het vliegtuig,
    b. een *lineair toestandsmodel* van het vliegtuig, met gelijke in- en uitgangssignalen als diegene beschikbaar voor de piloot, namelijk de roerhoeken en de barometrische signalen.

    bepalen.

Misschien belangrijker in de bijdrage om de vlucht meer economisch en veiliger te maken, werden deze praktische resultaten bekomen op een specifieke manier, die gekenmerkt wordt door de volgende attributen van de ontwikkelde rekenschema's:

1. zij zijn *zelfinstellend* en garanderen de nauwkeurige werking van de algoritmen terwijl rekening gehouden wordt met verschillende tijdsvarierende effecten, die normaal aanleiding geven tot ontoelaatbare fouten in de schattingsresultaten.
2. zij maken enkel gebruik van signalen die gemeten worden door een *standaard instrumentatie* systeem op de huidige generatie verkeersvliegtuigen. De nauwkeurigheid waarmee deze signalen gemeten worden zal enkel toenemen voor de toekomstige generatie verkeersvliegtuigen.
3. zij werken volledig *onafhankelijk*, dit is zonder gebruik te maken van andere systemen die een gelijkaardige taak uitvoeren ter correctie van mogelijk schattingsfouten. Dit is bijvoorbeeld noodzakelijk voor de huidige gebruikte traagheidsnavigatie systemen waar de fouten op de vliegbaan periodiek verminderd worden door Radio navigatie systemen.

Deze samenvatting maakt dus duidelijk dat de voorgestelde rekenschema's vele perspectieven inhouden voor de nabije toekomst. Wat natuurlijk aanleiding geeft tot verder onderzoek. Dit zal meer concreet behandeld worden in de navolgende sectie.

D.7. Suggesties voor verder onderzoek

De resultaten voorgesteld in dit werk pretenderen zeker niet het "real-time" vliegtuig reconstructie te hebben opgelost. Dit is hoofdzakelijk omdat geen praktische evaluatie is uitgevoerd op een vliegtuigsimulator of in

werkelijke vliegproeven. Niettemin tonen de veel belovende resultaten in de lineaire computer simulatie met het DHC-2 Beaver vliegtuig aan dat de ontwikkelde algoritmen nu pas beschouwd kunnen worden in een diepere practische evaluatie. Dit vormt dan ook een van de hoofdgebieden van verder onderzoek. Meer concreet, kunnen we dit verder onderzoek opsplitsen in de volgende drie delen:

1. Het eerste deel, zal zich moeten toeleggen op een verdere verfijning van de ontwikkelde algoritmen. De modules waar een singuliere waarde ontbinding noodzakelijk is kunnen meer efficient gemaakt worden door het gebruik van recursieve singuliere waarde ontbindingstechnieken [15].
   Ten tweede, houdt de huidige vorm van de tweetraps identificatie methode geen rekening met atmosferische turbulentie. Met dit effect kan rekening gehouden worden door het vliegbaanreconstructie algoritme uit te breiden met een module die de fouten statistiek bepaalt van de systeemruis. Dit kan bijvoorbeeld gedaan worden met een techniek beschreven in [124], ofwel kan een singuliere waarde ontbinding uitgevoerd worden rechtstreeks op de traagheidssignalen. Het geintroduceerde verlies aan nauwkeurigheid van de gereconstrueerde vliegbaan kan het gebruik van de totale KM [34] wenselijk maken om nauwkeurige schattingen van de aerodynamische model parameters te garanderen.
2. In een tweede deel, kan dan een volledig test programma uitgevoerd worden, om de prestatie van de tweetrapsmethode als een "real-time" vliegtuig identificatie methode te evalueren. Dit kan eerst uitgevoerd op een vliegtuig simulator, gevolgd door werkelijk vliegproeven. Deze testen kunnen mogelijke tekortkomingen van de voorgestelde algoritmen aan het licht brengen. Dit zal dan aanleiding geven tot veranderingen in het ontwerp. Het doel van deze evaluatie moet de vraag beantwoorden, of de tweetrapsmethode in deze vorm het mogelijk maakt de zeer kostbare tijd van vliegproeven nodig voor vliegtuig prototypes en vliegtuigveranderingen te certifieren te verminderen.
3. Een derde deel, kan de "spinoff" evalueren van de "on-line" kennis van aerodynamische model parameters die rekening houden met tijdsvariaties in vlucht condities. Ten eerste, kan dit gebeuren in het ontwerp van vliegtuig besturings systemen. Daardoor zou de tweetrapsidentificatie module uitgebreid dienen te worden door een

regel module, bijvoorbeeld gebaseerd op de optimale regeltheorie [50]. Dit zal adaptieve regeling van verkeersvliegtuigen mogelijk kunnen maken. Ten tweede, kan het gevolg voor de optimalisering van de vlucht geevalueerd worden.

Een tweede hoofd onderzoeksgebied is het verder zetten van de theoretische analyse van algoritmen in lineaire systeemtheorie. Gecondenseerde vormen, kunnen zoals aangegeven in hoofdstuk 2, aangewend worden om nieuwe algoritmen af te leiden die betrouwbaarheid en efficientie verenigen voor andere problemen in de lineaire systeemtheorie. Twee verdere uitbreidingen in dit licht kunnen hierbij zeker aan de orde komen:

1. Systeem theoretische problemen voor gegeneralizeerde systeembeschrijvingen, zoals het gegeneralizeerde KF algoritme, zoals voorgesteld door C.C. Paige in [81]. Deze veralgemeende voorstelling laat singuliere covariantie matrices toe, zonder gebruik te maken van de gegeneralizeerde inverse. Het grote nadeel van de methode is echter dat de berekeningen zeer inefficient verlopen. Het gebruik van gecondenseerde vormen kan dit euvel misschien verhelpen.
2. Het KF probleem om filter voorspellingen te maken.

Een ander interessant onderzoeksgebied hierin is het verder zetten van de fouten analyse voor andere KF implementaties die geformuleerd zijn op basis van bepaalde specifieke systeem eigenschappen, zoals bijvoorbeeld de UDU' factorizatie implementatie [8], die enkel van toepassing is voor het sequentieel verwerken van de uitgangsmetingen [1].

REFERENCES

[1] B.D.O. Anderson , J.B. Moore, "Optimal filtering", *Prentice Hall*, Information and System Sciences Series, Englewood Cliffs, New Jersey, 1979.

[2] B.D.O. Anderson , J.B. Moore, "Detectability and stabilizability of time varying discrete time linear systems," *SIAM J. Control and optimization*, Vol. 19, No. 1, pp. 20-32, 1981

[3] K.J. Åstrom, U. Borisson, L. Ljung and B. Wittenmark, "Theory and applications of self-tuning regulators," *Automatica, 13*, pp. 457-476, 1977

[4] A. Barraud, "A numerical algorithm to solve A'XA - X = Q," *IEEE Trans. Aut. Contr.*, Vol. AC-22, pp.883-885, 1977

[5] S. Barnett, "Matrices in Control Theory," *Van Nostrand Reinhold Company*, London, 1971

[6] R. Bartels, G. Stewart, "Solution of the matrix equation AX + XB = C," *Comm. ACM*, Vol. 15, pp. 820-826, 1972

[7] R.E. Bellman, "Matrix Analysis", 2nd ed., *McGraw-Hill*, New York, 1968

[8] G.J. Bierman, "Factorization methods for discrete sequential estimation", *Academic Press*, New York, 1976

[9] G.J. Bierman , C.L. Thornton, "Numerical comparison of Kalman filter algorithms: orbit determination case study", *Automatica*, Vol.13, pp.23-35, 1977

[10] D. Boley, "Computing the controllability-observability decomposition of linear time-invariant systems. A numerical approach,", *Ph. D. Thesis*, Dept. Comp. Sci., Stanford Univ., 1981

[11] J.H. Breeman, K. Van Woerkom, "Aspects of flight test instrumentation" *Agard Lecture series No. 104 on parameter identification,* Oct. 1979

[12] A.E. Bryson, Y.C. Ho, "Applied Optimal Control," Revised printing, *J. Wiley*, New York, 1975

[13] A.E. Bryson, "Mini-issue on NASA's advanced control law program for the DFBW aircraft," *IEEE Trans. Aut. Contr.*, Vol. AC-22,pp.752-805, Oct. 1977

[14] A.E. Bryson, "Kalman filter divergence and aircraft motion estimators," *Int. J. Guidance and Control*, Vol. 1, No. 1, pp.71-79, 1978

[15] J.R. Bunch and P. Nielsen, "Updating the singular value decomposition," *Numer. Math. 31*, Springer Verslag, pp. 111-129, 1978

[16] Cline, C. Moler, G. Steward, J. Wilkinson, "An estimate for the condition number of a matrix" *SIAM J.of Numerical Analysis*, 16, pp. 368-375, 1979

[17] J.F. Creedon, "Flight Management Systems I: What are they and why are they being developed?" *AIAA Paper 83-2235*, 1983

[18] T.R. Crossley and B. Porter, "State observations and their application," *Ch. 13 of Modern approaches to control system design*, edt. by N. Munro, England 1979

[19] J.J. Dongarra, C. Moler, J. Buch and G. Stewart, "LINPACK User's guide," *SIAM*, Philadelphia, 1979

[20] D. Duncan and S. Horn, "Linear dynamic recursive estimation from the viewpoint of regression analysis," *J. Amer. Stat. Assoc. 67*, pp. 815-821, 1972

[21] A. Emami-Naeini and P. Van Dooren, "Computation of zeroes of linear multivariable systems," *Automatica*, Vol. 18, pp. 415-430, 1982

[22] H. Erzberger , J.D. Chapel, "Concepts and algorithms for terminal-area traffic management," *Proceedings of ACC 1984*, pp. 166-173, San-Diego, June 1984

[23] P. Eyckhoff, "System Identification," *J. Wiley*, London, 1974

[24] B. Etkin, "Dynamics of atmospheric flight," *J. Wiley*,New York 1975

[25] G. Francis, "The QR transformation. A unitary analogue to the LR transformation - I and II," *Computer Journ.*, Vol. 4, pp. 265-271 and 332-345, 1961 and 1962

[26] G. Franklin and J. Powell, "Digital Control of Dynamic Systems," *Addison-Wesley publishing company*, California 1980

[27] B. Friedlander, "Treatment of bias in recursive filtering," *IEEE Trans Aut.Contr.*, Vol. AC-14, pp.359-367, August 1969

[28] B. Friedlander, T. Kailath and L. Ljung, "Scattering theory and linear least-squares estimation. Part II: Discrete time problems," *Journal of the Franklin Inst.*, vol. 301, no. 1 & 2, pp. 71-82, 1976

[29] F.R. Gantmacher, "The Theory of Matrices", *Chelsea Publishing Company*, New York, 1960

[30] O.H. Gerlach, "Analysis of a method for determination of performance and stability derivatives in non-stationary, symmetric flights" Delft University of Technology, *VTH-report no-117*, 1964

[31] G. Golub, S. Nasch and C. Van Loan, "A Hessenberg-Schur method for the problem AX + XB = C," *IEEE Trans. Aut. Contr.*, Vol. AC-24, pp. 909-913, 1979

[32] G. Golub and W. Kahan, "Calculating the singular values and pseudo-inverse of a matrix," *SIAM Journ. Num. Anal.*, Vol. 2, pp. 205-224, 1965

[33] G. Golub, "Numerical methods for solving linear least squares problems," *Numer. Math.*, Vol. 7, pp. 206-216, 1965

[34] G.H. Golub, C.F. Van Loan, "Matrix Computations", *The John Hopkins University Press*, Baltimore, 1983

[35] N.K. Gupta and R.K. Mehra, "Computational aspects of the Maximum Likelihood estimation and reduction in sensitivity function calculation," *IEEE Trans. Aut. Contr.*, AC-19, No. 6, 1974

[36] W.E. Hall, N.K. Gupta and J.S. Tyler, "Model structure determination and parameter identification for non-linear aerodynamic flight regimes," *AGARD proceedings No. 172 on methods for Aircraft state and parameter identification*, Nov. 1974

[37] T. Hägglund, "New estimation techniques for adaptive control," *PhD-dissertation*, Lund Institute of Technology, Sweden 1983

[38] S. Hammerling, "Numerical solution of the stable non-negative definite Lyapunov equation," *IMA Journ. Num. Anal.*, Vol. 2, pp.303-323, 1982

[39] S. Hammerling, "Newton's method for solving the algebraic Riccati equation," *NPL Report DITC 12/82*, National Physical Laboratory, 1983

[40] B.L. Ho and R.E. Kalman, "Effective construction of linear state-variable models form input/output functions," *Regelungstechnik 14*, pp. 545-548, 1966

[41] R.J. Hosman, "Advanced flight test instrumentation: design and calibration," *Agard Lecture series No. 104 on Methods for Aircraft state and parameter identification*, pp. 6.1-6.17, Nov. 1974

[42] D.J. Hughes and O.L. Jacobs, "Turn-off, escape and probing in non-linear stochastic control," *IFAC Symp. on Stochastic control*, pp. 343-352, 1974

[43] A.H. Jazwinski, "Stochastic processes and filtering theory", *Academic Press*, New York, 1970

[44] R.M. Johnstone and B.D.O. Anderson, "Experimental convergence of recursive least squares with exponential forgetting factor," *System and Control letters*, No. 2, 1982

[45] H.L. Jonckers, "Application of the Kalman filter to flight-path reconstruction from flight test data including estimation of instrumental bias error corrections," Delft University of Technology, *VTH-report no-162*, March 1976

[46] L.S. Jong, "Towards a formal definition of numerical stability," *Numer. Math.*, Vol. 28, pp. 211-220, 1977

[47] A.W. Jonas, L.E. Garner, "Leadless chip carriers: The packaging technique for the 1980's," *IEEE Trans. Syst. Cir.*, Vol. SC-17, pp. 713-722, 1981

[48] T. Kailath, "An innovation approach to least-squares estimation (four parts)," *IEEE Trans. Aut. Contr.*, Vol. AC-16, No. 6, pp. 646-660, 1968

[49] T. Kailath, "Some new algorithms for recursive estimation in constant linear systems", *IEEE Trans. Inform. Theory*, Vol. IT-19, pp. 750-760, Nov. 1973

[50] T. Kailath, "Linear Systems," *Prentice-Hall*, Englewood Cliffs, New Jersey, 1980

[51] R.E. Kalman, "A new approach to Linear filtering and prediction problems," *Trans. ASME* (J. Basic Eng.), Vol.82D, pp.34-45, Mar. 1960

[52] P.G. Kaminski et al, "Discrete square root filtering. A survey of current techniques", *IEEE Trans.Aut.Contr.*, Vol.AC-16, pp.727-736, Dec. 1971

[53] L. Kaufman, "The LZ-algorithm to solve the generalized eigenvalue problem," *SIAM Journ. Num. Anal.*, Vol. 11, pp. 997-1024, 1974

[54] M.G. Kendall and A. Stuart, "The advanced theory of statistics," Vol I and II, 2^{nd} *Edition Charles Griffin and Comp.*, Ltd, London 1967

[55] V.C. Klema and A.J. Laub, "The singular value decomposition: its computation and some applications," *IEEE Trans. Aut. Contr.*, AC-25, No. 2, 1980

[56] J. Kautsky, N. Nichols and P. Van Dooren, "Robust pole assignment in linear state feedback," *Int. Journ. Contr.*, Vol. 41, pp. 1129-1156, 1985

[57] S. Kourouklis and C.C. Paige, "A constrained least squares approach to the general Gauss-Markov linear model," No. 375, pp. 620-625, 1981

[58] M. Konstantinov, P. Petkov and N. Christor, "Synthesis of linear systems with desired equivalent form," *Journ. CACM*, Vol. 6, pp. 27-35, 1980

[59] V. Kučera, "Singular Systems: Steady stare LQG Control," *1985 Benelux meeting on systems and control*, Jan. 1985

[60] H. Kwakernaak, "Asymptotic Root Loci of Multivariable linear optimal Regulators," *IEEE Trans. Aut. Contr.*, AC-21, pp. 378-382, 1976

[61] A. Laub, "Efficient multivariable frequency response computations," *IEEE Trans. Aut. Contr.*, Vol. AC-26, pp. 407-408, 1981

[62] D.G. Lainiotis, "Partitioned estimation algorithms II: Linear estimation" *J. Inf.Sci. vol.7*, pp.317-340, 1974

[63] E.J. Lerner, "The automated cockpit: The digital revolution catches up with commercial airline cockpits and stirs old questions over how many crew members are needed," *IEEE Spectrum*, pp. 57-62, Febr. 1983

[64] L. Ljung , T. Söderström, "Theory and practice of recursive identification," *MIT Press*, 1983

[65] S. Ljung and L. Ljung, "Error propagation properties of recursive least squares adaptation algorithms," *Automatica*, Vol. 21, No. 2, pp. 157-167, 1985.

[66] L. Ljung, M. Morf and D. Falconer, "Fast calculation of gain matrices for recursive estimation schemes," *Int. J. Control 27*, No. 1, 1978

[67] D. Luenberger, "Observing the state of a linear system," *IEEE Trans. Mil. Electr.*, Vol. 8, pp. 74-80, 1964

[68] D. Luenberger, "Observers for multivariable systems," *IEEE Trans. Aut. Contr.*, Vol. AC-11, pp. 190-197, 1966

[69] R.E. Maine , K.W. Iliff, "Formulation and implementation of a practical algorithm for parameter estimation with process and measurement noise," *SIAM J.Appl.Math, Vol.41,No.3*, pp.558-579, Dec. 1981

[70] A. Mc Farlane, ed., "Frequency-Response Methods in Automatic Control," *IEEE Reprint book*, IEEE press, New York 1979. See also *IEEE Trans. Aut. Contr.*, AC-24, pp. 250-265, 1979

[71] A. Miele, "Lagrange multipliers and quasi-steady flight mechanics," *Journal of Aero/Space Sciences*, pp. 592-598, Sept. 1959

[72] J.A. Mulder et al, "Analysis of aircraft performance, stability and control measurements," *Agard Lecture series No. 104 on parameter identification*, pp. 5.1-5.87, Oct. 1979

[73] G. Minimis and C. Paige, "An algorithm for pole assignment of time invariant multi-input linear system," *Proceedings 21st IEEE conf. Dec. and Contr.*, pp.62-67, 1982

[74] M.Morf , T. Kailath, "Square-root algorithms for least squares estimation", *IEEE Trans.Aut.Contr.*, Vol. AC-20, pp.487-497, Aug. 1975

[75] C. Moler and G. Stewart, "An algorithm for the generalized matrix eigenvalue problem," *SIAM Journ. Num. Anal.*, Vol. 10, pp. 241-256, 1973

[76] C. Moler and C. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix," *SIAM Rev.*, Vol. 20, pp. 801-836, 1978

[77] J.A. Mulder, J.C. Den Hollander and H. Brinkhorst, "Aerodynamic model identification from dynamic flight test data and windtunnel experiments," *Agard conf. no. 339 on ground flight tests techn. and correlation*, Oct. 1982

[78] C.C. Paige, "Computer solution and perturbation analysis of generalized linear least squares problems," *Math. Compt.*, Vol.33, No. 145, pp. 171-183, 1978

[79] C.C. Paige, "Fast numerically stable computations for generalized linear least squares problems," *SIAM Journ. Numer. Anal.*, Vol. 16, No. 1, pp. 165-171, 1979

[80] C.C. Paige, "Properties of numerical algorithms related to computing controllability," *IEEE Trans. Aut. Contr.*, Vol. AC-26, pp. 130-138, 1981

[81] C.C. Paige, "Covariance matrix representation in linear filtering", in the *Special Issue of Contemporary Mathematics on Linear Algebra and its Role in Systems Theory*, AMS, 1985.

[82] C.C. Paige , M.A. Saunders, "Least squares estimation of discrete linear dynamic systems using orthogonal transformations", *SIAM J.Num.Anal.*, Vol.14, pp.180-193, April 1977

[83] R. Patel, "Computation of minimal order state-space realizations and observability indices using orthogonal transformations," *Int. Journ. Contr.*, Vol. 33, pp. 227-246, 1981

[84] B. Porter and A. Garis, "Computer-aided robustness analysis of linear multivariable set-point tracking systems incorporating PI controllers," *IASTED symposium on Aut. Contr. and Ident.*, pp. 131-136, Copenhagen, June, 1983

[85] J.E. Potter, R.G. Stern, "Statistical filtering of space navigation measurements", *Proceedings 1963 AIAA Guidance and Control Conference*, 1963

[86] C.F. Price, "An analysis of the Divergence Problem in the Kalman Filter," *IEEE Trans. Aut. Contr.*, Vol. AC-16, no. 6, pp. 699-702, 1968

[87] H. Rosenbrock, "State-space and multivariable theory," Wiley, New York 1970

[88] A. van der Sluis, "Stability of the solution of linear least squares problems", *Numerische Mathematik*, Vol.23, pp.241-254, 1975

[89] L. Silverman, "Representation and realization of time-variable linear systems," *Technical Report 94*, New York, Columbia Univ., Dept. Elec. Eng, 1966

[90] R.E. Spradlin, "Flight Management Systems II: Where are we today and what have we learned?," *AIAA paper 83-2236*, 1983

[91] J. Staar and J. Vandewalle, "Singular value decomposition: A reliable tool in the algorithmic analysis of linear systems," *Journal A*, Vol. 23, No. 2, pp. 69-74, 1982

[92] G.W. Stewart, "Introduction to matrix computations", *Academic Press*, New York, 1973

[93] G.W. Stewart, "Perturbation bounds for the QR factorization of a matrix", *SIAM J.Num.Anal.*, Vol.14, pp.509-518, June 1977

[94] G.W. Stewart, "On the perturbation of pseudo-inverses, projections and linear least squares problems", *SIAM Review*, Vol.19, pp. 634-662, Oct. 1977

[95] R.J. Tibor, J.C. Hall, "Flight Management Systems III: Where are we going and will it be worth it?," *AIAA paper 83-2237*, 1983

[96] M.H. Verhaegen, "The application of the Maximum Likelihood Method to Aerodynamic model identification and flight-path reconstruction from simulated flight-test data," *Eng. thesis*, Delft University of Technology, Aug. 1982

[97] P. Van Dooren, M.H. Verhaegen, "The use of condensed forms in linear system theory," *IEE Workshop on Robust Num. softw. in Contr. Syst. design*, London, Nov. 1983

[98] P. Van Dooren, M.H. Verhaegen, "On the use of unitary state-space transformations", in the *Special Issue of Contemporary Mathematics on Linear Algebra and its Role in Systems Theory*, AMS, 1985.

[99] M.H. Verhaegen, "Adaptive flight-path reconstruction and aerodynamic model identification," submitted to the *Journ. of Aircrafts*

[100] M.H. Verhaegen, P. Van Dooren , "A theoretical error analysis and new implementations of different Kalman filter algorithms," to appear in *IEEE Trans. Aut. Contr.*

[101] M.H. Verhaegen , J. Vandewalle, "An analysis of two methods to analyze dynamic flight test manoeuvres," *IASTED symposium on Automatic control and Identification*, pp.13.17-13.22, Copenhagen, June 1983, also in *IASTED J. Mod. and simulation*

[102] M.H. Verhaegen , J. Vandewalle, "The contribution of aircraft identification techniques in the development of Flight Management

Systems," *Journal A*, VOL. 26, No. 2, pp. 57-66, 1985

[103] M.H. Verhaegen et al, "A robust and efficient least squares estimation algorithm," *IFAC conference on Identification and System parameter estimation*, pp. 681-687, London, June 1985

[104] M.H. Verhaegen , P. Van Dooren, "An efficient implementation of square root filtering: Error analysis, complexity and simulation on flight-path reconstruction," *INRIA conference on analysis and optimization of systems*, Nice, June 1984, Springer-Verslag 62-63, pp. 250-267

[105] M.H. Verhaegen, J. Vandewalle and P. Van Dooren, "A Robust and Efficient technique for dealing with time varying instrumental bias in linear filtering," *IASTED Symposium on Modelling, Identification and Control*, Innsbrück, Febr. 1984, also in *IASTED J. Contr. and Comp.*, Vol. 12, no. 3, 1984

[106] M. Verhaegen, "On the need of reliable software in aircraft control systems," *T-dag on Aerospace and Spinoff*, pp. 151-159, Oostende, 27 Sept. 1984

[107] P. Van Dooren, A. Emami-Naeini and L. Silverman, "Stable extraction of the Kronecker structure of pencils," *Proceedings 17th IEEE Conf. Dec. and Contr.*, pp. 512-524, 1979

[108] A. Varga, "A numerically stable algorithm for standard controllability form determination," *Electronic Letters*, Vol. 17, pp. 74-75, 1981

[109] P. Van Dooren, "The generalized eigenstructure approach for solving Riccati equations," *IEEE Trans. Aut. Contr.*, Vol-26, pp. 111-129, 1981

[110] P. Van Dooren, "Numerical Linear Algebra: An increasing interest in linear system theory," *Proceedings ECCTD 81*, The Hague, The Netherlands, pp. 538-544, Aug. 1981

[111] P. Van Dooren, M. Van Begin, "MATLAB-SC users guide," *Technical Note No. 166*, PRLB Brussels, June 1985

[112] A. Varga, "A Schur method for pole assignment," *IEEE Trans. Aut. Contr.*, Vol. AC-26, pp. 517-519, 1981

[113] A. Varga and V. Sima, "Numerically stable algorithm for transfer function matrix evaluation," *Int. Journ. Contr.*, Vol. 33, pp.

1123-1133, 1981

[114] C. Van Loan, "Computing integrals involving the matrix exponential," *IEEE Trans. Aut. Contr.*, Vol. AC-23, pp. 395-404, 1978

[115] A. Varga, V. Sima and C. Varga, "On numerical simulation of linear continuous control systems," *Proceedings Simulation'83 Symposium*, Praha, 1983

[116] P. Van Dooren, "Reduced order observers: a new algorithm and proof," *Systems and Control Letters*, Vol. 4, pp. 243-251, 1984

[117] G. Verghese, P. Van Dooren and T. Kailath, "Properties of the system matrix of a generalized state-space system," *Int. J. Control*, 1979

[118] G. Verghese, "Infinite-Frequency behavior in Generalized Dynamical Systems," *Ph.D. dissertation*, Dept. of Electrical Eng., Stanford University, Stanford, Calif., Aug. 1978

[119] P. Van Dooren, "A generalized eigenvalue approach for solving Riccati equations," *SIAM Scisc. Vol.2*,no.2, June 1981

[120] J. Vandewalle, H. De Man and J. Rabbaey, "Time, Frequency and z-domain modified nodal analysis of switched-Capacitors Networks," *IEEE Trans. on Circ. Syst.*, CAS-28, No. 3, pp. 186-195, 1981

[121] M. Van Ostaeyen, "Geometrische correctie van scanner beelden uit de luchtfotografie," *Ph.D. dissertation*, Dept. Comp. Sc., Catholic University Leuven, Leuven, Dec. 1985

[122] J. Wilkinson and C. Reinsch, "Handbook for automatic computation. II: Linear algebra," *Springer Press*, New York, 1971

[123] J.H. Wilkinson, "The algebraic eigenvalue problem", *Clarendon Press*, Oxford, 1965

[124] R.K. Mehra, "On the identification of variances and adaptive Kalman filtering," *IEEE Trans. Aut. Contr.*, AC-15, Apr. 1970