

# Multiple-Model Based Diagnosis for Adaptive Fault-Tolerant Control

Redouane Hallouzi



# **MULTIPLE-MODEL BASED DIAGNOSIS FOR ADAPTIVE FAULT-TOLERANT CONTROL**

PROEFSCHRIFT

ter verkrijging van de graad van doctor  
aan de Technische Universiteit Delft,  
op gezag van de Rector Magnificus Prof. dr. ir. J.T. Fokkema,  
voorzitter van het College voor Promoties,  
in het openbaar te verdedigen op  
donderdag 17 april 2008 om 15:00 uur

door

**Redouane HALLOUZI**

elektrotechnisch ingenieur  
geboren te Beni Touzine, Marokko

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. ir. M. Verhaegen

Samenstelling promotiecommissie:

|                               |                                           |
|-------------------------------|-------------------------------------------|
| Rector Magnificus,            | voorzitter                                |
| Prof. dr. ir. M. Verhaegen,   | Technische Universiteit Delft, promotor   |
| Prof. dr. R. Babuška,         | Technische Universiteit Delft             |
| Prof. dr. ir. J. Hellendoorn, | Technische Universiteit Delft             |
| Prof. dr. ir. J. A. Mulder,   | Technische Universiteit Delft             |
| Prof. dr. M. Kinnaert,        | Université Libre de Bruxelles             |
| Prof. dr. J. Stoustrup,       | Aalborg University                        |
| Dr. ir. S. Kanev,             | Energy Research Centre of the Netherlands |
| Prof. dr. ir. B. De Schutter, | Technische Universiteit Delft, reservelid |



This dissertation has been completed in partial fulfillment of the requirements for the graduate study of the Dutch Institute of Systems and Control (DISC).



The work presented in this thesis has been supported by the Dutch Technology Foundation (STW) under project number 04506.

ISBN: 978-90-9022978-2

Copyright © 2008 by R. Hallouzi.

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission from the copyright owner.

Printed in The Netherlands

# Preface

**T**his thesis is the result of 4 years of Ph.D. research, during which I had the pleasure to interact with various people. These people all contributed to this thesis in their own ways. First of all, I want to express my sincere gratitude to my supervisor Michel Verhaegen. Michel has been responsible for guiding and supporting me in many ways during my Ph.D. research. Using his extensive knowledge and insight he has been able to provide me with very useful suggestions when I needed them most. At the same time he has always left me the freedom to pursue my own research interests.

I enjoyed working with all my colleagues at the Delft Center for Systems and Control. I would particularly like to mention Jelmer Braaksma, Sjoerd Dietz, Rudy Negenborn, Justin Rice, Eric Trottemant, and Jan-Willem van Wingerden, with whom I spent enjoyable times both inside and outside of the office. I really appreciated the time with my roommate Diederick Joosten who was always willing to discuss just about anything that popped into our minds. I also want to thank Olaf Gietelink for the good company he has been as a carpool buddy and for exchanging thoughts on fault diagnosis issues.

I am grateful to the members of my Ph.D. committee for providing me with constructive remarks, which have helped me to improve this thesis. I would like to thank the participants of GARTEUR AG-16 for the interesting discussions during project meetings throughout Europe and for the development of the aircraft benchmark model. I am grateful to the SIMONA crew for making it possible to implement my controller in their simulator. I also would like to thank the members of the STW user committee for their input during project meetings and for hosting some of these meetings.

Last, but certainly not least, I would like to thank my parents and my family for their support.

*Redouane Hallouzi,  
Amsterdam, March 2008.*



# Contents

|                                                                |           |
|----------------------------------------------------------------|-----------|
| <b>Preface</b>                                                 | <b>v</b>  |
| <b>1 Introduction</b>                                          | <b>1</b>  |
| 1.1 Need for Diagnosis and Fault-Tolerant Control . . . . .    | 1         |
| 1.2 Model-Based Diagnosis . . . . .                            | 4         |
| 1.3 Fault-Tolerant Control . . . . .                           | 11        |
| 1.3.1 Control Allocation . . . . .                             | 12        |
| 1.3.2 Gain-Scheduling . . . . .                                | 13        |
| 1.3.3 Model Predictive Control . . . . .                       | 13        |
| 1.4 Research Objectives . . . . .                              | 15        |
| 1.5 Boeing 747 Benchmark Model . . . . .                       | 16        |
| 1.6 Contributions . . . . .                                    | 17        |
| 1.7 Organization of the Thesis . . . . .                       | 18        |
| <b>I Fault Diagnosis</b>                                       | <b>21</b> |
| <b>2 Multiple-Model Estimation: a Convex Model Formulation</b> | <b>23</b> |
| 2.1 Introduction . . . . .                                     | 23        |
| 2.2 Problem Formulation . . . . .                              | 26        |
| 2.2.1 Hybrid Model Structure . . . . .                         | 27        |
| 2.2.2 Convex Model Structure . . . . .                         | 29        |
| 2.3 Multiple-Model Estimation Algorithms . . . . .             | 29        |
| 2.3.1 Interacting Multiple-Model Filter . . . . .              | 30        |
| 2.3.2 Dual Convex Model Filter . . . . .                       | 31        |
| 2.3.3 Extended Kalman Filter . . . . .                         | 37        |
| 2.3.4 Properties of Multiple-Model Filters . . . . .           | 38        |
| 2.4 Experimental Results . . . . .                             | 38        |

|          |                                                              |           |
|----------|--------------------------------------------------------------|-----------|
| 2.4.1    | Aircraft Fault Detection and Isolation . . . . .             | 40        |
| 2.4.2    | Tracking of a Maneuvering Target . . . . .                   | 45        |
| 2.5      | Conclusions . . . . .                                        | 49        |
| <b>3</b> | <b>Automatic Model Set Design</b>                            | <b>51</b> |
| 3.1      | Introduction . . . . .                                       | 51        |
| 3.2      | Problem Formulation . . . . .                                | 53        |
| 3.3      | Orthogonal Decomposition (OD) Based Method . . . . .         | 53        |
| 3.3.1    | Motivation for Using Orthogonal Decompositions . . . . .     | 54        |
| 3.3.2    | Derivation of the OD-based Model Set Design Method . . . . . | 57        |
| 3.4      | Convex Polytope Based Method . . . . .                       | 60        |
| 3.5      | Limit Values Based Method . . . . .                          | 63        |
| 3.6      | Simulation Example . . . . .                                 | 64        |
| 3.6.1    | Model Set Design For a Mass-Spring-Damper System . . . . .   | 66        |
| 3.6.2    | Simulation Results . . . . .                                 | 68        |
| 3.6.3    | Differences Between the Model Set Design Methods . . . . .   | 73        |
| 3.6.4    | Comparison with the “Minimum-Mismatch” Method . . . . .      | 75        |
| 3.7      | Conclusions . . . . .                                        | 77        |
| <b>4</b> | <b>Model Sets for Aircraft Fault Diagnosis</b>               | <b>79</b> |
| 4.1      | Introduction . . . . .                                       | 79        |
| 4.2      | Lock-in-Place Faults . . . . .                               | 81        |
| 4.3      | Model Set Design . . . . .                                   | 82        |
| 4.3.1    | Description of the Linearized Models . . . . .               | 83        |
| 4.3.2    | Incorporating Different Trim Offsets . . . . .               | 84        |
| 4.3.3    | Model Set I . . . . .                                        | 84        |
| 4.3.4    | Model Set II . . . . .                                       | 86        |
| 4.4      | Simulation Results . . . . .                                 | 87        |
| 4.4.1    | Linear Simulation . . . . .                                  | 88        |
| 4.4.2    | Nonlinear Simulation . . . . .                               | 89        |
| 4.5      | Conclusions . . . . .                                        | 94        |



|           |                                                                        |            |
|-----------|------------------------------------------------------------------------|------------|
| <b>II</b> | <b>Fault-Tolerant Control and its Integration with Diagnosis</b>       | <b>95</b>  |
| <b>5</b>  | <b>Persistency of Excitation for Subspace Predictive Control (SPC)</b> | <b>97</b>  |
| 5.1       | Introduction . . . . .                                                 | 97         |
| 5.2       | Subspace Predictive Control . . . . .                                  | 99         |
| 5.2.1     | Subspace Predictor . . . . .                                           | 100        |
| 5.2.2     | Subspace Predictor Integrated with a Predictive Control Law            | 103        |
| 5.3       | Persistency of Excitation . . . . .                                    | 106        |
| 5.3.1     | Computing the SVD Using Inverse Iterations . . . . .                   | 107        |
| 5.3.2     | Excitation Condition Integrated with the SPC Cost Function             | 108        |
| 5.4       | Simulation Results . . . . .                                           | 109        |
| 5.4.1     | SPC as an Adaptive Controller . . . . .                                | 109        |
| 5.4.2     | Persistency of Excitation for SPC . . . . .                            | 111        |
| 5.5       | Conclusions . . . . .                                                  | 112        |
| <b>6</b>  | <b>Subspace Predictive Control Applied to Fault-Tolerant Control</b>   | <b>115</b> |
| 6.1       | Introduction . . . . .                                                 | 115        |
| 6.2       | Architecture of the Fault-Tolerant Control System . . . . .            | 117        |
| 6.2.1     | Control Loops . . . . .                                                | 118        |
| 6.2.2     | Fault Isolation . . . . .                                              | 118        |
| 6.3       | Closed-Loop Subspace Predictive Control . . . . .                      | 119        |
| 6.3.1     | Closed-Loop Subspace Predictor (CLSP) . . . . .                        | 119        |
| 6.3.2     | CLSP Integrated with a Predictive Control Law . . . . .                | 123        |
| 6.4       | SPC (Re-)configuration . . . . .                                       | 125        |
| 6.5       | Simulation Results . . . . .                                           | 128        |
| 6.5.1     | Trajectory Following for the Nominal Case . . . . .                    | 129        |
| 6.5.2     | Trajectory Following for Elevator Lock-in-Place . . . . .              | 129        |
| 6.5.3     | Trajectory Following for Rudder Runaway . . . . .                      | 131        |
| 6.5.4     | Trajectory Following for “Bijlmerramp” Condition . . . . .             | 133        |
| 6.5.5     | Discussion of the Simulation Results . . . . .                         | 134        |
| 6.6       | Real-Time Implementation . . . . .                                     | 136        |
| 6.7       | Conclusions . . . . .                                                  | 138        |
| <b>7</b>  | <b>Conclusions and Recommendations</b>                                 | <b>139</b> |
| 7.1       | Conclusions . . . . .                                                  | 139        |
| 7.2       | Recommendations for Further Research . . . . .                         | 143        |

|          |                                                                  |            |
|----------|------------------------------------------------------------------|------------|
| <b>A</b> | <b>Flight Parameters and Controls of the Boeing 747 Aircraft</b> | <b>145</b> |
| A.1      | Flight Parameters . . . . .                                      | 145        |
| A.2      | Controls of the Boeing 747 . . . . .                             | 146        |
| <b>B</b> | <b>Discretization of a Multiple-Model System</b>                 | <b>149</b> |
| <b>C</b> | <b>Recursive Update of Matrix <math>R</math></b>                 | <b>151</b> |
|          | <b>Bibliography</b>                                              | <b>155</b> |
|          | <b>List of Abbreviations</b>                                     | <b>167</b> |
|          | <b>Summary</b>                                                   | <b>169</b> |
|          | <b>Samenvatting</b>                                              | <b>173</b> |
|          | <b>Curriculum Vitae</b>                                          | <b>177</b> |

**F**ault diagnosis combined with fault-tolerant control is a key enabling technology for increasing safety and reliability of control systems. Therefore, considerable research effort has been and still is being made to develop fault diagnosis and fault-tolerant control methods that can readily be applied to complex real-life systems. This thesis makes a contribution to these ongoing developments. In this chapter the position of the work performed in this thesis in relation to existing research is explained. To this end, an overview of existing methods is presented. On the basis of an analysis of advantages and disadvantages of existing methods, the chosen path for fault diagnosis and fault-tolerant control in this thesis is motivated. For fault diagnosis the multiple-model framework is adopted and for fault-tolerant control a data-driven adaptive control method is used. Additionally, this chapter presents an overview of the contributions of this thesis. Finally, this chapter outlines the structure of the thesis and the relations between the subsequent chapters.

---

## 1.1 Need for Diagnosis and Fault-Tolerant Control

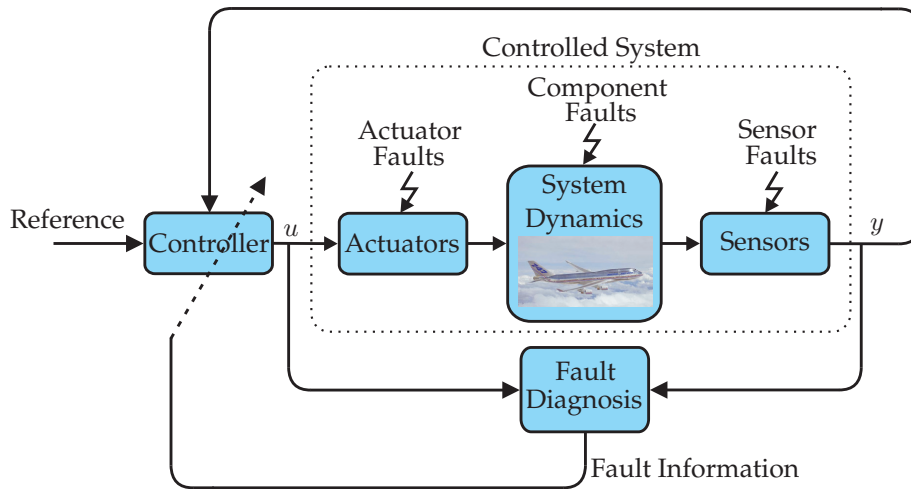
Our present-day society is strongly dependent on the availability and correct functioning of control systems. Control systems appear in many products that are used in everyday life, but mostly remain unnoticed by their users. For example, control systems can be found in household appliances ranging from washing machines to coffee makers. But they can also be found in cars, ships, and aircraft. Under normal conditions, control systems perform the tasks they are designed for and therefore their users are unaware of them. However, when a fault occurs that prevents correct functioning of the system, this indeed gets noticed by the user. If, for example, the heating element in a washing machine does not function properly anymore, this can result in laundry not being entirely clean. Although this lack of reliability of the washing machine can be annoying for its user, the consequences are not catastrophic. However, a fault that occurs in a more safety-critical system,

such as an aircraft, can result in a catastrophe involving injury or even the death of many people. An example of the catastrophic consequences of a fault is the disaster related to EL AL flight 1862 that crashed into a building in Amsterdam in 1992. This crash happened after both engines had separated (Smaili and Mulder 2000) from one side of the aircraft. This crash, known in The Netherlands as the “Bijlmerramp” (NRC 2007), caused the death of 43 people. From this example it is apparent that measures should be taken to ensure safe operation of control systems even in the case of faults.

Before continuing, it is important to establish what events can be classified as a fault. A generally accepted definition of a fault (Blanke et al. 2006) is that it is an unpermitted deviation of at least one characteristic property or parameter of a system from its acceptable/usual/standard condition. The determination of a fault at a certain time is referred to as fault detection. Only detecting that a fault has occurred usually does not provide enough information to accommodate the fault. For this purpose, more information, such as the kind of fault and the location of the fault, is required. The determination of a fault, its kind, and its location is generally referred to as fault detection and isolation (FDI). If, in addition to FDI also the size of the fault is determined, this is referred to as fault diagnosis (FD).

Once a fault has been diagnosed, it has to be accommodated. This can be done in several ways depending on the type of application. In non-safety-critical systems (such as washing machines) it can suffice to notify the user of the system that a fault has occurred. In this way the user knows that something is wrong with the system and that he therefore has to repair the system. If the fault is repaired in a timely fashion this can even prevent more faults from happening as a result of the first fault. In safety-critical systems a fault has to be accommodated in a more immediate way to prevent that the fault will eventually lead to a disaster. A method to do so, is to design a controller that can adapt or reconfigure itself based on the FD information such that the system can still operate safely. A control system with this property can therefore be named a fault-tolerant control (FTC) system. In Figure 1.1 the architecture of an FTC system is shown. In this figure it can be seen that the control system consists of three different parts, all of which are susceptible to faults. These three parts are the actuators, the sensors, and the components of the system. Components can be anything that is part of the physical structure of a system, for example the wing of an aircraft. Component faults can therefore also be referred to as structural faults. Moreover, it can be seen in Figure 1.1 that there is a controller that issues a control command  $u$ , such that the controlled system tracks the desired reference signals. For this purpose, the controller has availability of the reference signals and the measurement outputs of the system  $y$ . It can also be seen that faults are diagnosed based on the signals  $u$  and  $y$ . The fault information from the FD system is subsequently used to adapt or reconfigure the controller.

An important requirement for an FTC system is that the control system has to be redundant. This means that if a fault occurs, the controller has sufficient control authority left to proceed with control of the system, possibly with degraded performance. For example, in most aircraft both the rudder surfaces and aileron



**Figure 1.1:** Architecture of an FTC system.

surfaces can be used for lateral control. Although the aileron surfaces are the ones that are primarily used, if a fault occurs in the aileron surfaces the rudder surfaces can (partly) take over and vice versa. A series of three examples will be given that both forms a strong motivation for FTC systems and illustrates how redundancy can be used to prevent disasters. All of these examples are related to the rudder problems on earlier versions of the Boeing 737. Several incidents occurred with this type of aircraft that could possibly be accounted to a failing rudder surface. Three of these incidents are described in the following:

1. On March 3, 1991, United Airlines flight 585 crashed in Colorado Springs, USA. 25 people were killed during this disaster. This particular incident was carefully investigated by the National Transportation Safety Board, which concluded that this accident was probably caused by an uncontrolled “movement of the rudder surface to its blowdown limit” (NTSB 2001). The blowdown limit of a control surface is defined as the maximum amount of surface travel available for an aircraft at a specific flight condition. This phenomenon in which a surface deflects to its maximum or minimum limit is also known as “runaway” or “hard over”.
2. On September 8, 1994, US Air flight 427 crashed in Pittsburgh, USA. Again it was concluded by the National Transportation Safety Board that a probable cause of this accident was the uncontrolled movement of the rudder surface to its blowdown limit (NTSB 1999).
3. On April 11, 1994, problems occurred with the rudder on a Boeing 737 property of Continental Airlines over the Gulf of Honduras (Seattle Times 2007). After the pilot had been flying normally, he felt the aircraft suddenly twist and roll violently to the right. He immediately disengaged the autopilot

and turned the control wheel sharply to the left. These actions caused the ailerons to counteract the rolling/yawing movement caused by the faulty rudder. After that, the pilot succeeded in safely landing the aircraft.

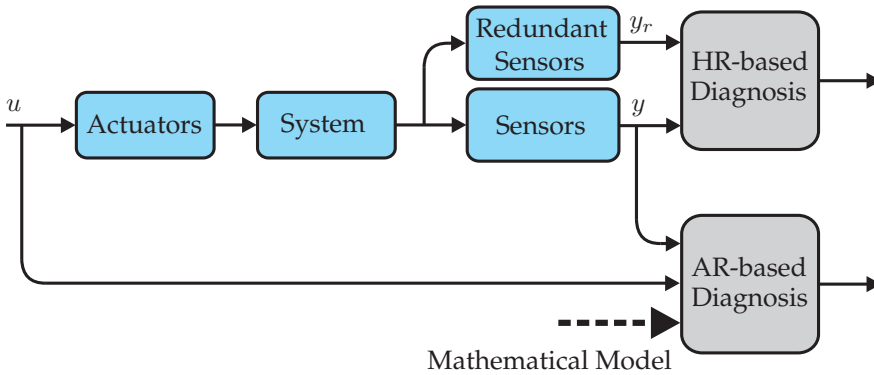
From the last incident the conclusion can be drawn that even in case of a potentially catastrophic fault such as rudder runaway, which most probably also caused the first two fatal incidents, there is enough redundancy available in the aircraft to safely land the aircraft. A requirement for this, however, is that appropriate actions should be taken immediately after occurrence of the fault. The pilot in the last incident managed to do so successfully. However, even experienced pilots are not always able to take the necessary actions when faults occur, such as for example the pilots of EL AL flight 1862, which caused the “Bijlmerramp”. Therefore, FTC systems that can quickly take appropriate actions in case of faults can indeed increase safety in safety-critical systems such as aircraft.

## 1.2 Model-Based Diagnosis

A generally applicable approach to perform diagnosis is based on the concept of *hardware redundancy*. The main idea behind this concept is to use multiple sensors, actuators, and system components for the same purpose. If a fault occurs, it can then be diagnosed by determining which of the redundant system parts exhibits different behavior from the others. For example, in the aircraft industry, using hardware redundancy is a proven concept to diagnose sensor faults (Oostrom et al. 2002). Vital sensors are tripled or even quadrupled and faults in these sensors are diagnosed by using voting schemes. Drawbacks of using hardware redundancy are that it adds to hardware costs, maintenance, and weight. Furthermore, extra hardware also adds extra weight and requires extra space which is not desirable in many applications. Note that hardware redundancy should not be confused with the previously described redundancy of the control system. The main difference is that for hardware redundancy the system is redundantly equipped with components having exactly the same function. This is opposed to redundancy of the control system, for which the components have different primary functions.

In order to perform diagnosis some kind of redundancy is always required. If using hardware redundancy is not a viable option, this redundancy must be obtained otherwise. Another way to do this is, is to use the relations that exist between the measured variables of the different system parts. This concept is known as *analytical redundancy* and exploits the redundant analytical relations between the different measured variables. A simple example that illustrates analytical redundancy is a washing machine with a faulty heating element. If the heating element is driven by a certain amount of current (measured by a sensor), then the water is expected to have a certain temperature. If there is too much discrepancy between the measured temperature and the expected water temperature, it can be concluded that a fault has occurred. In this simple case it cannot be

determined whether the heating element, the temperature sensor, or the current sensor is faulty. For this purpose more relations (if present) between the different system parts should be exploited. The analytical relations between the different system parts can be captured in a mathematical model. This explains why diagnosis methods that use the analytical redundancy concept are referred to as model-based diagnosis methods. In Figure 1.2 the hardware and analytical redundancy concepts are illustrated for the case of sensor diagnosis. It can be seen that the scheme based on hardware redundancy (HR) uses a set of redundant sensors, which provide the same measurements  $y_r$  as the original set of sensors. Instead of using redundant sensors the scheme based on analytical redundancy (AR) uses a mathematical model.



**Figure 1.2:** Comparison between sensor diagnosis schemes based on hardware redundancy (HR) and analytical redundancy (AR).

In accordance with the current trend of research into diagnosis methods, model-based diagnosis is considered in this thesis. This subject started to be studied in the early 1970s just after the establishment of observer theory. One of the first reported methods for model-based diagnosis is a fault detection filter (Beard 1971; Jones 1973). Since then, this subject has received significant interest in literature resulting in many different diagnosis methods. Several books and papers have appeared that give an overview of this field (Chen and Patton 1999; Simani et al. 2003; Venkatasubramanian et al. 2003; Kinnaert 2003; Blanke et al. 2006; Isermann 2006). The bulk of model-based diagnosis methods can be classified into one of three classes (Frank 1990). An important aspect that distinguishes the different classes is the type of faults that can be detected. The two main fault types are multiplicative faults and additive faults. Multiplicative faults are characterized by a product of the fault signal with the system variables. Additive faults are characterized by a summation of the fault signal and the system variable. The three classes for model-based diagnosis are each described in the following.

### Observer-based methods

The main idea behind observer-based methods is to estimate the outputs of the system from the measurements or subsets of the measurements through use of observers. Subsequently, the estimation residual (also known as the innovation), can be computed as the difference between the estimated output and the measured output. This residual can be used for the purpose of FD. In the nominal case the model used by the observer and the real system should correspond well, which would lead to a zero residual. In case of a fault, the residual would be non-zero. To see how such a residual is constructed, consider the state-space system

$$x_{k+1} = Ax_k + Bu_k, \quad (1.1)$$

$$y_k = Cx_k + Du_k, \quad (1.2)$$

where  $x_k$  is the state,  $u_k$  is the input, and  $y_k$  is the output of the system. The state of this system can be estimated with an observer as follows

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - \hat{y}_k), \quad (1.3)$$

$$\hat{y}_k = C\hat{x}_k + Du_k, \quad (1.4)$$

where  $\hat{x}_k$  and  $\hat{y}_k$  denote estimates of  $x_k$  and  $y_k$ , respectively, and matrices  $A$ ,  $B$ ,  $C$ , and  $D$  are the system matrices. The matrix  $L$  is the observer gain, which determines the behavior of the observer. The residual that is of interest for FD is  $y_k - \hat{y}_k$ . Using a single observer is not sufficient for fault isolation. For this purpose several observer schemes can be used (Frank 1990). A well known scheme is the dedicated observer scheme. This scheme consists of a set of observers each of which is driven by a different single sensor output. Each of these observers then estimates the full output vector  $y_k$ , or if this is not possible, part of the output vector. The dedicated observer scheme can be used to detect and isolate multiple faults by analysis of the residuals. Another well known scheme is the generalized observer scheme. This scheme consists of a set of observers each of which is driven by all outputs except for one. The generalized observer scheme can be used for isolation of single faults. Because of its structure, the generalized observer scheme is less sensitive to modeling errors and disturbances than the dedicated observer scheme. Another observer-based diagnosis method that does not require multiple observers to isolate faults is the method based on a fault detection filter (Beard 1971; Jones 1973). Such a filter is characterized by an observer gain that is chosen such that particular faults affect the single residual in a particular manner. This brief description of observer-based methods is concluded by pointing out that these methods are especially suitable for additive faults and that an accurate model is required.

### Parity relation based methods

Parity relations are relations consisting of the plant model, or transformed variants thereof that are zero in the nominal case and non-zero when a fault occurs.



These parity relations can in principle be either based on direct redundancy or on temporal redundancy (Chow and Willsky 1984). Direct redundancy exploits relationships among instantaneous outputs of sensors and temporal redundancy exploits relationships among the histories of sensor outputs and actuator inputs. A simple example that illustrates a parity relation based on direct redundancy is based on the discrete-time relation between velocity  $v_k$  and acceleration  $a_k$  of a system:  $v_{k+1} = v_k + T_s a_k$ , where  $T_s$  denotes the sample time. If both velocity and acceleration can be measured perfectly, then  $v_{m,k+1} = v_{m,k} + T_s a_{m,k}$  is a parity relation (the subscript  $m$  denotes a measured quantity). The residual  $r_k = v_{m,k+1} - v_{m,k} - T_s a_{m,k}$  can be used to detect faults in the pair of sensors. To see how a parity relation based on temporal redundancy can be derived, consider the state-space system defined by (1.1) and (1.2). By repeated substitution of the state and measurement equations the following relation can be obtained

$$\overbrace{\begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+s} \end{bmatrix}}^{Y_s} = \overbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^s \end{bmatrix}}^{\mathcal{O}_s} x_k + \overbrace{\begin{bmatrix} D & 0 & \cdots & 0 \\ CB & D & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ CA^{(s-1)}B & \cdots & CB & D \end{bmatrix}}^{\mathcal{T}_s} \overbrace{\begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+s} \end{bmatrix}}^{U_s}, \quad (1.5)$$

in which a data window of size  $s$  is considered. If it is assumed that the system matrices  $A$ ,  $B$ ,  $C$ , and  $D$  are known, then the only unknown in the above relation is the state  $x_k$ . The dependency on  $x_k$  can be annihilated by computing a row vector  $w_s$  in the left null space of  $\mathcal{O}_s$ , which means that  $w_s \mathcal{O}_s = 0$ . Using this vector, the parity relation  $w_s Y_s = w_s \mathcal{T}_s U_s$  can be obtained. The corresponding residual  $r_s = w_s (Y_s - \mathcal{T}_s U_s)$  is non-zero in case of any fault condition that causes the model described by matrices  $A$ ,  $B$ ,  $C$ , and  $D$  to be invalid for modeling the system under diagnosis. Therefore, this particular residual is only suitable for fault detection, but not for fault isolation. For this purpose, enhanced residuals (Gertler 1997) that require information on how faults affect the state and measurement equation can be designed. Enhanced residuals make it possible to isolate faults by providing well-defined responses to particular faults. In conclusion, it should be remarked that parity relations are especially suitable for additive faults and require the model to be known accurately.

### Parameter estimation based methods

Parameter estimation based methods rely on the fact that faults in systems are often reflected by variation of physical parameters such as, mass, damping, stiffness, etc. Faults can therefore be diagnosed by directly estimating the relevant parameters. If the estimated parameter value deviates from the nominal parameter value, then a fault has occurred. A general procedure for FD using parameter estimation consists of the following 5 steps (Isermann 1984; Frank 1990):

1. Choice of a parametric model of the system that relates the inputs, outputs,

and parameters of the system.

2. Determination of the relationships between the model parameter vector  $\theta$  and the physical parameter vector  $p$ :

$$\theta = f(p). \quad (1.6)$$

3. Estimation of the model parameter vector  $\theta$  using the inputs and outputs of the system, resulting in the estimate  $\hat{\theta}$ .
4. Construction of the physical parameter vector from the estimated parameter vector  $\hat{\theta}$

$$\hat{p} = f^{-1}(\hat{\theta}), \quad (1.7)$$

and computation of the deviation with respect to the nominal value, i.e.  $\Delta p = p - \hat{p}$ . The deviation  $\Delta p$  takes the role of the residual.

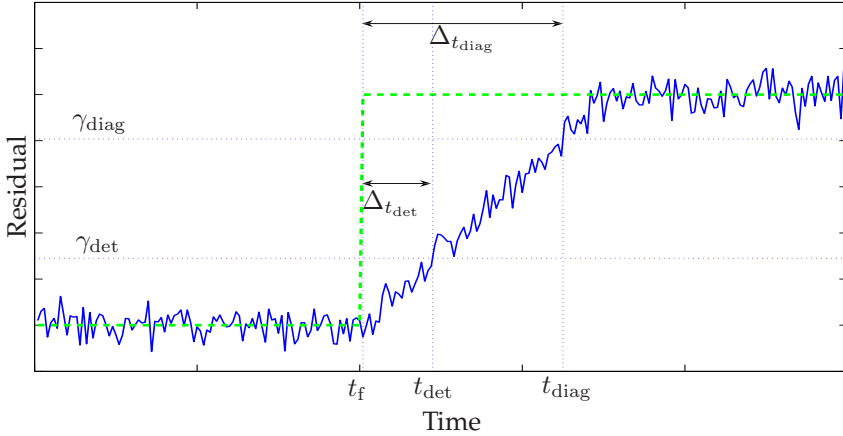
5. Faults can be diagnosed by using  $\Delta p$  and the known relations between the faults and the parameters.

Parameter estimation methods allow for more flexibility in how faults can affect the system than observer-based and parity relation based methods. Therefore, parameter estimation methods are well suitable for both additive faults and multiplicative faults. Furthermore, the requirements on how well the model represents the system are less strict since the parameters to be estimated do not have to be known exactly. Another difference with the previously described diagnosis methods is that parameter estimation methods require sufficient excitation of the system (by the input) to achieve good estimation performance.

## Residual evaluation

Three classes of model-based diagnosis have been described in the preceding sections, each of which produce residuals that are affected by faults. The next step is to make decisions on the faults based on these residuals. This step is known as the residual evaluation step. In this step a change in the residual has to be detected first. The simplest way to do so, is to check whether the residual signal exceeds a certain fixed threshold. Many other algorithms exist that may produce better results depending on the properties of the residual signal. A comprehensive overview of change detection algorithms is given in the two books by Basseville and Nikiforov (1993) and Gustafsson (2000). The type of change detection algorithm that is used has a significant influence on the performance of the diagnosis system. For example, a small threshold could result in too many false alarms and a large threshold could result in too many missed detections, both of which are undesirable. The choice of the threshold directly determines the performance indices of the diagnosis system. In Figure 1.3 an illustration is given of how the choice of a threshold influences two important performance indices, namely the

detection time  $\Delta_{t_{\text{det}}}$  and the diagnosis time  $\Delta_{t_{\text{diag}}}$ . A complete overview of performance indices of diagnosis systems is given by Bartys et al. (2006). In Figure 1.3 a residual signal is depicted together with a signal that represents the evolution of the residual in case of perfect fault information. This latter signal is indicated by a dashed line. An abrupt fault occurs at  $T = t_f$ . Using the threshold  $\gamma_{\text{det}}$  this fault is detected at  $T = t_{\text{det}}$  with a delay of  $\Delta_{t_{\text{det}}}$ . Subsequently, using the threshold  $\gamma_{\text{diag}}$  this fault is diagnosed at  $T = t_{\text{diag}}$  with a delay of  $\Delta_{t_{\text{diag}}}$ . Note that  $\gamma_{\text{diag}}$  cannot be known in advance because the fault size is generally not known before it occurs. Therefore,  $\gamma_{\text{diag}}$  should be interpreted as a parameter that indicates that the estimated fault size is close enough to the actual fault size. This results in  $\gamma_{\text{diag}}$  being dependent on the actual fault size. For the parameter estimation based methods it is possible to determine the size of (i.e. to diagnose) a fault from a single residual. However, for the observer-based methods and parity relation based methods this is not always possible. In this case only the detection part illustrated in Figure 1.3 can be applied.



**Figure 1.3:** Illustration of how fault information can be obtained from a residual signal.

### Extension to nonlinear systems: an LPV approach

Most of the classical methods developed for model-based FD are developed for linear models. However, many real-life systems cannot be modeled by linear models. For this purpose, the linear FD methods should be extended to nonlinear systems. Therefore, methods have been recently developed that make explicit use of nonlinear models (De Persis and Isidori 2001; Zhang et al. 2005). These methods, which are based on nonlinear control theory (Isidori 1995), can result in very elegant solutions for specific nonlinear diagnosis problems. However, these solutions are mostly valid under rather restrictive assumptions, which limit their general applicability. Furthermore, analysis and design of diagnosis methods based

on nonlinear models can become very involved for complex systems. A possible solution that overcomes these limitations is to use linear parameter-varying (LPV) models to approximate nonlinear systems. The notion of LPV models was first introduced by Shamma and Athans (1991). The big advantage of LPV models is that powerful linear design tools for stability and performance can be extended and applied. LPV models have a linear structure in which the model parameters can be time-varying. The most general state-space LPV model has the following form

$$x_{k+1} = A(\rho_k)x_k + B(\rho_k)u_k, \quad (1.8)$$

$$y_k = C(\rho_k)x_k + D(\rho_k)u_k, \quad (1.9)$$

where  $\rho_k$ , which should be known, is the scheduling vector containing the time-varying parameters. There exist several examples of the use of LPV models for diagnosis purposes (Bokor and Balas 2004; Hallouzi et al. 2005; Szász et al. 2005). LPV models can be derived in different ways, for example Hallouzi et al. (2005) identified an LPV model of a small commercial aircraft using the LPV identification techniques developed by Verdult et al. (2002). Szász et al. (2005) used an LPV model of the longitudinal motion of a Boeing 747 that was directly obtained from the nonlinear model using state transformations (Marcos and Balas 2004).

A difficult but important part of deriving an LPV model is the choice of the scheduling vector  $\rho_k$ . The reason for this is that this choice very much depends on the specific system and no systematic approach exists to the author's knowledge that can make this choice. Furthermore, for FD purposes the scheduling vector would ideally consist of the fault parameters. However, these are not known in advance. An LPV model that can overcome this drawback is the polytopic LPV model. This type of LPV model has the following form (Apkarian et al. 1995)

$$x_{k+1} = \sum_{i=1}^{N_m} \mu^{(i)}(\rho_k) \left[ A^{(i)}x_k + B^{(i)}u_k \right], \quad (1.10)$$

$$y_k = \sum_{i=1}^{N_m} \mu^{(i)}(\rho_k) \left[ C^{(i)}x_k + D^{(i)}u_k \right], \quad (1.11)$$

where  $N_m$  is the number of local models described by the state-space quadruple  $\{A^{(i)}, B^{(i)}, C^{(i)}, D^{(i)}\}$ . The model weights  $\mu^{(i)}(\rho_k)$  are constrained to ensure convexity

$$\mu^{(i)}(\rho_k) \geq 0, \quad \forall i \in \{1, 2, \dots, N_m\}, \quad \forall k, \quad (1.12)$$

$$\sum_{i=1}^{N_m} \mu^{(i)}(\rho_k) = 1, \quad \forall k. \quad (1.13)$$

The polytopic LPV model is also referred to as a multiple-model (MM) system (Murray-Smith and Johansen 1997) and will also be referred to as such in this thesis. MM systems can be used for FD without knowledge of the model weights

that depend on the scheduling vector. These model weights are left to be estimated on-line. In this sense FD based on an MM system can be classified as a parameter estimation based approach, with the model weight being the parameter to be estimated. An assumption that is made in this case is that the local models are known. This is not always the case, as can be concluded from the research by Verdult et al. (2002) and Fujimori and Ljung (2006), in which the local models are identified assuming that the scheduling vector is known. However, in this thesis it is assumed that a system model is present from which local models can be generated at desired operating conditions.

### Multiple-model framework

The MM framework is an attractive framework for FD because of its flexible structure that allows intuitive modeling of faults. In a state-space setting a component fault can generally be modeled by a modification of the  $A$ -matrix, an actuator fault can generally be modeled by a modification of the  $B$ -matrix, and a sensor fault can generally be modeled by a modification of the  $C$ -matrix. For this reason FD using the MM framework has attracted significant interest (Zhang and Li 1998; Maybeck 1999; Yen and Ho 2003; Verma et al. 2004; Ru and Li 2008; Ducard and Geering 2008). The MM structure that is usually employed for FD is different from the one that is defined by (1.10)-(1.11). The difference is that the local models are not weighted in the same way as is done in (1.10)-(1.11). Instead, a set of local models, each having a separate state, is used that do not interact with each other. Transition probabilities between the different local models are defined in a transition probability matrix. This type of system is referred to as a jump Markov linear system (JMLS), which is a type of hybrid system. A hybrid system is characterized by the fact that it can suddenly switch between distinct modes of operation. The drawback of using a JMLS for FD is that faults that can be represented by weighted combinations of local models, which is for example the case when the considered faults occur only partially, are difficult to diagnose. The model structure defined by (1.10)-(1.11) is a better option than JMLSs to represent this kind of conditions as a result of its better interpolation properties. This is shown in Chapter 2 of this thesis.

## 1.3 Fault-Tolerant Control

FTC systems are systems that can maintain an acceptable level of control even after the occurrence of faults. A more formal definition of an FTC system is given by Blanke et al. (2006) as “a control system where a fault is accommodated with or without performance degradation, but a single fault does not develop into a fault on subsystem or system level”. Many different methods to perform FTC have appeared in recent literature. Hajiyeve and Caliskan (2003), Jiang (2005), Blanke et al. (2006), and Zhang and Jiang (2006) have authored books and papers that

provide an overview of the field of FTC. In this section an overview is given of the most commonly used methods.

FTC systems can be categorized into two classes: *passive FTC* systems and *active FTC* systems. Passive FTC systems introduce fault tolerance into a control system by the use of a fixed controller that is robust to a set of anticipated faults. Such a robust controller is designed off-line and does not adapt to the anticipated faults on-line. However, the robustness properties of the controller ensure that a pre-defined level of performance is achieved in case of anticipated faults. One disadvantage of this method is apparent: only anticipated faults can be dealt with, since the robust controller design is based on the set of anticipated faults. Another disadvantage is concerned with the conservativeness of the robust controller. If a large set of faults is anticipated, the set of system conditions against which the controller has to be robust is also large. The robust controller in such a case is likely to be conservative, which results in a low overall performance level. It is even possible that a robust controller does not exist for a given set of anticipated faults. An advantage of passive FTC is that a fixed controller has relatively modest hardware and software requirements. Another advantage is that passive FTC, due to its lower complexity with respect to active FTC, can be made more reliable according to classical reliability theory (Stoustrup and Blondel 2004). Examples of passive FTC systems can be found in the research by Liao et al. (2002); Niemann and Stoustrup (2005); Zhang et al. (2007).

Active FTC systems differ from passive FTC systems in that they can adapt on-line to fault information. This on-line adaptation allows active FTC systems to deal with more faults and generally achieve better performance than passive FTC systems. For these reasons, more research has been performed in the field of active FTC systems than in the field of passive FTC systems. In the following an overview and description will be given of a number of relevant approaches to FTC that are reported in current literature.

### 1.3.1 Control Allocation

The control allocation method is concerned with determining the set of actuator signals that produces a desired set of actuation forces. These actuation forces are generated by a controller irrespective of faults that have occurred. This is accomplished by adapting a matrix  $B_f$  that relates the actuation forces to the actuator signals. This matrix can be determined by FD or by system identification. The control allocation method therefore has the ability to adapt the way actuation forces are generated from the available actuators, to the faults that have occurred. For example, if the effectiveness of a certain actuator becomes 0% due to a fault, the corresponding column in  $B_f$  will also become 0. This actuator is then not considered anymore by the control allocation method. Instead, the remaining actuators can be used to generate the desired actuation forces. Given a matrix  $B_f$  and the desired actuation forces  $F_d$ , then the control signal for the actuators  $u_c$  can be

computed by solving

$$u_c = \arg \min_u \|B_f u - F_d\|, \quad (1.14)$$

which is formulated without constraints on  $u_c$ . These constraints must be added if this is required for the controlled system. Note that the optimization problem in (1.14) can have more than one solution. In that case  $u_c$  should be chosen to be one of these solutions. An advantage of the control allocation method is that the control law (the one that generates the actuation forces) does not have to be modified when faults occur. A disadvantage is that preferences to use specific inputs for specific actuation forces can not be straightforwardly integrated. Control allocation has received considerable attention from the field of aerospace engineering, see for example the work by Bodson (2002) and Härkegård (2004).

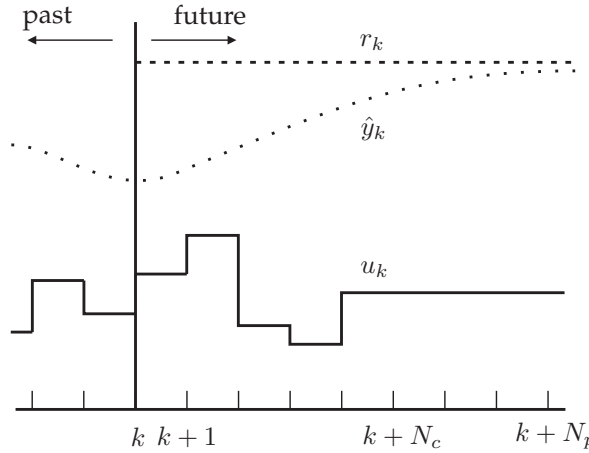
### 1.3.2 Gain-Scheduling

A possible method to achieve fault tolerance is to design separate controllers for each anticipated fault condition. If one of these fault conditions is diagnosed, then the corresponding controller is engaged. In case the diagnosed fault condition corresponds to a combination of the anticipated conditions, then the corresponding controllers are weighted accordingly. This concept, known as gain-scheduling, is regularly used in the field of flight control to deal with changing flight conditions (Nichols et al. 1993; Stilwell 2001). The scheduling is usually based on functions depending on flight parameters such as altitude and speed. The design of these functions can be a difficult task when many parameters are involved since systematic design methods are not readily available. These designs are therefore mostly done heuristically, which can take much effort (Oosterom 2005). The MM framework that can provide the scheduling parameters in the form of the estimated probabilities is therefore an attractive alternative to the heuristic design methods. For this reason, the MM framework has been used for FTC by various researchers (Maybeck 1999; Kanev and Verhaegen 2000; Zhang and Jiang 2001; Rodrigues et al. 2005). LPV controllers also belong to the class of gain-scheduling controllers (Shamma and Athans 1991). LPV controllers are linear controllers that depend on varying parameters, which can also include fault parameters, and can therefore be used for different operating conditions. For this reason, LPV control can also be a suitable method for FTC (Shin and Belcastro 2006; Gáspár and Bokor 2006).

### 1.3.3 Model Predictive Control

After its introduction in the 1970s, model predictive control (MPC) has become a popular strategy in the field of industrial process control. The main reasons for this popularity are the abilities of MPC to control multivariable systems and to handle constraints. MPC is sometimes also referred to as receding horizon control. This name is a result of the operating principle of MPC, which is illustrated





**Figure 1.4:** Illustration of the receding horizon principle of MPC.

in Figure 1.4. After optimization of a cost function over a time horizon in the future, only the first control sample is implemented. Next, the horizon is shifted with one sample, which means that the start of the future time horizon in Figure 1.4 becomes  $k + 1$ . Subsequently, the optimization is performed again using new measurements. The cost function that is optimized for MPC generally consists of a summation of two terms. One term corresponds to the tracking error, i.e. the difference between the reference signal  $r_k$  and the predicted output  $\hat{y}_k$ . The tracking error is computed for a horizon of  $N_p$  samples. This horizon is referred to as the prediction horizon. The second term of the cost function corresponds to the control effort. The control effort is based on the control signal  $u_k$  within a horizon of  $N_c$  samples. This horizon is referred to as the control horizon and it is generally chosen smaller than  $N_p$ . Optimization of the cost function is frequently performed subject to constraints. These constraints are usually related to operating constraints of the system, such as limitations on the control signals. Initially, MPC was primarily applied to relatively slow processes such as the processes that can be encountered in the process industry. The reason for this is that MPC can require a considerable computational effort to compute the control signals. For the relatively slow processes in the process industry, this drawback was not an issue because of the low sampling frequency of the controllers. However, for faster systems, higher frequencies were required that prevented on-line implementation of MPC for such systems. More recently, MPC has become a viable alternative for faster systems as a result of the increase in computational power that is available in modern control systems. For example, Seguchi and Ohtsuka (2003) have used MPC for real-time control of a miniature hovercraft. Another example is the work by Keviczky and Balas (2006), who have used MPC for real-time control of an unmanned aerial vehicle.

Because of its flexibility, MPC offers good possibilities for FTC (Maciejowski 2002). If a sensor of a controlled output fails, control of that output can be dis-



carded by removing the corresponding output from the cost function. Actuator faults can be represented by changing (or adding) constraints to the corresponding control signals. More generally, faults can be accommodated by changing the predictor, which is used to make the predictions  $\hat{y}_k$ , or by changing the objective. In this way, changes in the problem formulation can be made as a result of diagnosed faults. Examples of the application of MPC to FTC are numerous (Maciejowski and Jones 2003; Kanev 2004; Prakash et al. 2005; Kale and Chipperfield 2005; Keviczky and Balas 2006). An important issue when using MPC is the robustness with respect to model uncertainties. Since MPC heavily depends on how well the controlled system is represented by the model used, measures should be taken in case of model uncertainty. One method to do so is to define an uncertainty region around the nominal model and to ensure that the MPC algorithm achieves a certain minimum performance level for the whole uncertainty region. MPC methods that take model uncertainty explicitly into account are referred to as robust MPC methods. One of the first research efforts that addresses the issue of robust MPC was performed by Kothare et al. (1996). The PhD thesis by Kanev (2004) addresses this issue in the context of FTC.

Like most active FTC methods, MPC-based FTC requires availability of fault information to accommodate faults. This requirement limits the ability of MPC-based FTC to deal with unanticipated fault conditions for which fault information cannot be obtained most of the times. An FTC algorithm that has the ability to adapt to unanticipated fault conditions is therefore very desirable. Such an algorithm is subspace predictive control (SPC). This algorithm consists of a predictor that is derived using subspace identification theory (Verhaegen and Verdult 2007), making it a data-driven control method. This subspace predictor is subsequently integrated into a predictive control objective function. The basic SPC algorithm was introduced by Favoreel (1999) and has since been used by various researchers (Woodley et al. 2001; Kadali et al. 2003; Wang et al. 2007). If the subspace predictor is updated on-line with new input-output data when it becomes available, then SPC has the ability to adapt to changing system conditions, which can also include unanticipated faults. Besides having this ability, another important advantage of the SPC algorithm is that the issue of robustness with respect to model uncertainty is implicitly addressed because of the adaptation of the predictor. In Chapter 6 of this thesis the SPC algorithm is modified for the use in an FTC setting.

## 1.4 Research Objectives

It is apparent that improving safety and reliability of control systems is a well-motivated subject. The main research goal of this thesis therefore is to develop and investigate methods that can be used for this purpose. These methods should focus on how to obtain information on faults and how to adapt the controller of the system to accommodate faults. Furthermore, these methods should not focus on one application field only, but they should be as generally applicable as possible. In order to fulfill the main research goal, two more specific research objectives are

formulated:

1. Development of a diagnosis method for determining which system parts are affected by faults. This method should be robust to variations in the model parameters as a result of changing operating conditions.
2. Development of a reconfigurable controller that has the ability to adapt to faults in a data-driven fashion.

The key property that relates these two objectives is the nature of the fault information. Most FTC systems require detailed fault information, which at its turn requires detailed models. Since such models are difficult to obtain, a different philosophy for FTC is used in this thesis. Instead of requiring detailed fault information, the objective for FD is only to determine which system parts, e.g. actuators, are affected by faults. This information is subsequently used by the adaptive controller to reconfigure such that it can optimally accommodate the faults that have occurred. Next, the adaptive controller can adapt to faults in the reconfigured setting using input-output data of the faulty system.

The first research objective is met by using the MM framework. This framework is chosen for its ability to represent a wide variety of faults under different operating conditions. Contrary to mainstream MM methods which are based on a hybrid model structure, in this thesis an alternative structure is used that allows weighted combinations of local models. This structure has better interpolation properties which allow for a smaller model set. The second research objective is met by developing an FTC method based on the SPC algorithm. This algorithm combines a subspace predictor with a predictive control law. The ability to adapt to faults is a result of this subspace predictor being recursively updated using new input-output data when it becomes available. The fault information obtained by application of the MM framework is used to switch between different settings of the SPC-based FTC system.

## 1.5 Boeing 747 Benchmark Model

Throughout this thesis the theoretical results are illustrated, where possible, by means of a running example. This example consists of a detailed nonlinear model of a Boeing 747 aircraft. This model is used as a benchmark in Action Group 16 (AG-16) of the GARTEUR (Group for Aeronautical Research and Technology in EUROpe) project. AG-16 aims at integrating advanced FDI methods with control reconfiguration schemes. AG-16 has participants from both aerospace industry and universities in Europe. During the course of this thesis, the author has contributed actively to AG-16. The benchmark model has originally been developed for aircraft simulation and analysis by Van Der Linden (1998). Next, it has been adapted by Smaili (1999) to include a model of the Boeing 747 for the 1992 EL AL flight disaster in Amsterdam. Subsequently, it has been modified by Marcos

and Balas (2003) such that the model could be used as a benchmark for FTC and FDI. An FDI system for this benchmark by the same authors has also been reported (Marcos et al. 2005). The most recent modifications have been performed within GARTEUR AG-16 and include a benchmark scenario and a number of specific faults to be considered by the participants. These most recent modifications are documented by Smaili et al. (2006). Research performed by participants of GARTEUR AG-16 based on the latest benchmark model has been reported, for example, by Lombaerts et al. (2007); Cieslak et al. (2008); Alwi and Edwards. (2008).

The defined benchmark scenario consists of a number of flight phases including a heading change and a descent. The idea behind this scenario is that a number of elementary maneuvers should be flown even after the occurrence of faults. Another goal of AG-16 is to evaluate the developed FD and FTC methods in a 6 degree-of-freedom research flight simulator called SIMONA (SIMONA 2007). For this purpose the FTC and FD methods should be constructed such that they can run in real-time. The real-time simulator environment used in SIMONA is the Delft University Environment for Communication and Activation (DUECA) (Van Paassen et al. 2000).

GARTEUR AG-16 has been concluded with the organization of a final workshop, which was held in November 2007 in Delft (GARTEUR 2007). This workshop was also open for participants not involved in the GARTEUR project and it included an FTC demonstration on the SIMONA. In addition to the workshop, a book will be published that contains the results and descriptions of the methods developed by the different participants (Edwards et al. 2008). One chapter of that book will be based on Chapter 6 of this thesis.

## 1.6 Contributions

The contributions of this thesis to the current state-of-the-art of FD and FTC are contained in Chapters 2 to 6. The main contributions from these chapters are described in the following:

- MM systems provide a suitable framework for modeling a wide variety of faults. The MM systems that are traditionally applied to the purpose of FD are based on a hybrid model structure. An alternative MM system structure is proposed in this thesis that has better model interpolation properties. These properties allow for the use of smaller model sets. Algorithms for the nonlinear estimation of the state and model weights are developed, analyzed, and compared to existing algorithms based on the hybrid model structure. Part of these contributions have been reported in previous publications (Hallouzi et al. 2006a, 2008).
- Systematic design of model sets for the MM framework is a topic that has received very little attention in literature. In this thesis three novel model set design methods are described and analyzed. Two of these model set

design methods are based on the availability of a model from which local linear models can be generated at any desired operating condition. The third model set design method is based on limit values of the physical variables of the system and has also been described by Verhaegen et al. (2006).

- One of the three proposed model set design methods is used for modeling “lock-in-place” faults of actuators in aircraft. This type of fault is characterized by a control surface that freezes at a certain deflection position irrespective of the actuator command. The resulting model sets are evaluated using the Boeing 747 benchmark model. This contribution has also partly been reported by Hallouzi et al. (2006b).
- The basic SPC algorithm is modified for the purpose of FTC. The modifications include the implementation of an efficient recursive updating scheme for the predictor. Moreover, a condition that ensures persistency of excitation is implemented (Hallouzi and Verhaegen 2008b). This condition is necessary because a valid predictor can only be derived if the input-output data on which it is based contains sufficient information on the system.
- The SPC-based FTC system is integrated with a scheme for MM-based diagnosis and is applied to the Boeing 747 model. The proposed method has been applied to the faults defined in AG-16 of the GARTEUR project. The results for a number of these faults have also been reported in previous publications (Hallouzi and Verhaegen 2007, 2008a).
- An efficient real-time implementation of the SPC-based FTC system is developed. With this real-time implementation it is shown that although the FTC system may seem to be computationally intensive, it can indeed be applied to real-time FTC of a complex aircraft model such as the model of the Boeing 747.

Besides the previously mentioned main contributions, during the course of this thesis the author has contributed to the field of longitudinal control for automated vehicles with three papers (Hallouzi et al. 2004a,b; Gietelink et al. 2007a). Furthermore, contributions have also been made to diagnosis for such automated vehicles (Gietelink et al. 2007b).

## 1.7 Organization of the Thesis

This thesis is organized in two parts. The first part, which consists of Chapters 2 to 4, is concerned with FD. The second part, which consists of Chapters 5 and 6, is concerned with FTC and its integration with FD. The different chapters of this thesis are based on (parts of) different publications as is clear from Section 1.6. A consequence of this is that the notation used in these publications has been modified for this thesis to achieve notational consistency. Although frequently used notations are consistently used throughout the thesis, the different chapters

do still have local notations, which are defined per chapter. These locally defined notations, which should not be confused with local notations in other chapters, are characterized by the fact that they are used only on a small number of occasions within one chapter. In the following a brief description of the chapters is given.

In **Chapter 2** the MM framework is described along with a frequently used method for MM estimation using a conventional hybrid MM structure. Next, an alternative MM structure is proposed that allows for weighted combinations of the local models. It is shown that this structure leads to better interpolation properties than the conventional MM structure. Having better interpolation properties allows for the use of smaller model sets.

An important issue when using the MM framework is how the model sets to be used for MM estimation are designed. Although this is an important issue, few references can be found in the literature that address the structured design of such model sets. **Chapter 3** presents three methods for structured design of model sets. Two of these methods rely on the availability of a large model set that contains local models corresponding to different conditions of the system. This large model set is subsequently reduced by using either orthogonal decompositions or a convex polytope with a limited amount of vertices that contains all local models in the large model set. A third model set design method that is presented is based on the limit values of the system parameters.

In **Chapter 4** the method based on orthogonal decompositions is used to derive model sets for diagnosis of faults in aircraft. The considered faults include lock-in-place faults of control surfaces, which are also one of the main fault types considered in GARTEUR AG-16. The designed model sets are evaluated on the Boeing 747 benchmark for their ability to act as a basis for an MM diagnosis system.

**Chapter 5** introduces the SPC algorithm. The update of the subspace predictor, which is part of the SPC algorithm, requires persistent excitation of the system. Therefore a condition for persistency of excitation is developed for the SPC algorithm that allows an efficient on-line implementation. Next, in **Chapter 6** an FTC system based on a closed-loop version of the SPC algorithm is developed. This FTC system uses the FD scheme developed in **Chapter 4** to switch between different settings of the SPC algorithm. The developed FTC system is applied to the Boeing 747 benchmark model. Results for different fault scenarios, including the “Bijlmerramp” scenario, are also provided in this chapter.

The main conclusions of this thesis as well as the recommendations for further research are given in **Chapter 7**.



**Part I**

**Fault Diagnosis**





# Multiple-Model Estimation: a Convex Model Formulation

---

An important issue when using the multiple-model framework is how the estimation is performed. In this chapter a brief overview is given of the mainstream methods for multiple-model estimation and a new method is proposed. Contrary to existing methods that mostly adopt a hybrid model structure, the proposed method uses a more general multiple-model framework that allows for weighted combinations of the local models. The main advantage of this framework is that it has better model interpolation properties. These improved properties allow for smaller model sets, which is very useful in, for example, fault detection and isolation of partial faults. The improved interpolation properties are demonstrated by two simulation examples, one of which addresses a fault detection and isolation problem, and one of which addresses a target tracking problem. Monte-Carlo simulation results of these two examples are given. In these simulations, the well-known IMM filter is compared to two estimation algorithms based on the proposed model structure.

## 2.1 Introduction

Research on the multiple-model (MM) approach has attracted considerable interest in the last decades. The reason for this is the elegant solutions that the MM approach provides for estimation, control, and modeling problems (Narendra et al. 2003; Li et al. 2005; Fekri et al. 2006b). A well studied example of the application of MM to estimation is the target tracking problem. In this problem the local models usually correspond to kinematic modes, such as straight flight and coordinated turns of the target. An elaborate explanation of different MM algorithms applied to target tracking problems is given by Bar-Shalom et al. (2001). Another important estimation application of the MM framework is fault detection and isolation (FDI).

Numerous research efforts have been made in the field of MM FDI (Zhang and Li 1998; Maybeck 1999; Ni and Fuller 2003; Uppal and Patton 2005; Silva et al. 2007;

Rodrigues et al. 2008; Ducard and Geering 2008). The main motivation for using the MM framework for FDI is that it allows for a large class of fault conditions to be modeled. The reason for this is that in principle each of the local models might have totally different dynamics. Therefore, from an FDI perspective, the MM framework allows for the modeling of actuator, sensor as well as component faults. The basic idea of performing FDI with MM systems is as follows: a model set must be created that contains local models corresponding to different fault conditions of the monitored system. In addition to the fault models, the model set usually includes the nominal model. Faults are isolated by estimating which of the local models is valid using MM estimation algorithms. When there are no faults present in the monitored system, the nominal model will be valid. In case of a fault, one of the other models in the model set will become valid.

Most of the existing MM estimation algorithms provide a solution to the problem of estimating the state and the mode of a jump Markov linear system (JMLS). Numerous solutions are reported for this estimation problem ranging from particle filters (Doucet et al. 2001) to the well-known interacting MM (IMM) filter (Blom and Bar-Shalom 1988). A thorough overview of these different solutions is provided by Li and Jilkov (2005). The underlying model structure of a JMLS is hybrid. This means that it consists of a number of local models that do not interact with each other. Interaction between the different models can be added by the MM estimation algorithms themselves. However, it is important to note that this interaction is not inherent to the model structure itself. Both the MM estimation algorithms that do not have interaction between the models and the ones that do, display deteriorated performance in case the model set does not contain a model corresponding to the true system. This is the result of the assumption that the model corresponding to the true system should be included in the model set (Li and Jilkov 2005).

In case the weighted combinations of the local models in the JMLS model set correspond to physically relevant conditions, it is desirable to interpolate between these models. For example, Fisher and Maybeck (2002) use the MM adaptive estimation (MMAE) algorithm, which is also based on a JMLS, to identify partial actuator faults. For this purpose, model sets are used that contain models of the same fault with different sizes in order to be close to the true system in case of a fault with an arbitrary magnitude. This indicates that the MMAE is not able to interpolate well between models as a result of the chosen model structure. Otherwise, the partial faults could have been modeled by a weighted combination of only the nominal model and the total fault model. Another example in which the poor interpolation properties of the MM methods based on the JMLS are recognized is reported by Ru and Li (2003, 2008). These researchers have added an extra feature to the IMM filter for identifying partial faults. This feature introduces model sets with a finer parametrization (which means a larger model set) after the detection of the fault.

A possible remedy for the poor interpolation properties of MM algorithms based on the JMLS is to use another model structure that does explicitly interpolate between local models. Such a structure is the blended MM structure (Shorten

et al. 1999). In this structure, the model that is valid at a certain time is a weighted combination of the local models in the model set. When the combinations of the local models are restricted to be convex, a subset of the blended MM structure is created that is named the convex model (CM) structure. The convexity restriction is added to ensure that hybrid combinations (i.e. one of the local models is fully valid and the rest is not), are a subset of the set that consists of the convex combinations of the local models. This allows the use of the same type of model sets when MM estimation algorithms are used that are based on either the hybrid structure or the CM structure. MM estimation using the CM structure entails estimating both the state and the model weights of the local models. This estimation problem is nonlinear and nonconvex due to products of the state and the model weights.

In this and subsequent chapters, the MM system with a CM structure is used in the context of estimation for the purpose of diagnosis. However, this is not the only application of this particular type of MM system. Other applications of the MM system with a CM structure can also be encountered in the field of robust control (Wang and Balakrishnan 2002). In this field, the MM system is used to model uncertainties around a nominal model. As a result of the convexity of the model with uncertainty, elegant control synthesis methods can be derived for it by using linear matrix inequalities. Another application field of the MM system with the CM structure is MM control (Fekri et al. 2006a). In this application, local controllers are developed for different local models. The actual control signal of the MM controller is a weighted combination of the control signals from the local controllers. How the control signals are weighted is determined by the estimated model weights.

The main contribution of this chapter is to present the CM structure as an alternative to the hybrid model structure of the JMLS. The CM structure is an improvement upon the hybrid model structure in the following ways:

1. The CM structure has better model interpolation properties because it explicitly allows for interpolations of local models. Better interpolation properties are very desirable when the weighted combinations of local models also correspond to physically relevant conditions. This is the case, for example, for partial fault modeling in FDI problems. In this case, having better interpolation properties allows for smaller model sets.
2. The CM structure does not apply a transition probability matrix. This matrix contains the transition probabilities between the different local models in a JMLS. In theory, this matrix is usually assumed known for MM estimation in a JMLS. However, in practice, the transition probability matrix is considered to be a “design parameter” due to insufficient information being available. Since this parameter can be difficult to design in practice, recently methods have been proposed for online estimation of the transition probability matrix (Jilkov and Li 2004). Instead of using the transition probability matrix, MM estimation methods based on the CM structure rely more on measured data and less on a priori information (i.e. the transition probability matrix).

In this chapter, the IMM filter is chosen as the representative filter for MM estimation based on JMLSs because it is widely accepted for this purpose and because of its simplicity. Although the IMM filter does not provide an exact solution for MM estimation based on JMLSs, it provides a sufficiently approximate solution for many applications (Li and Jilkov 2005). For the nonlinear estimation problem related to the CM structure, two filters are used. These two filters also provide an approximate solution to the original estimation problem. One filter solves the problem in two linear filtering steps using dual filtering methods (Wan and Nelson 2001) and the other filter uses the augmented extended Kalman filter (EKF) (Ljung 1979), which is based on linearization. A comparison between the IMM filter and the two CM filters is presented in this chapter. This comparison is based on an FDI problem and a target tracking problem.

This chapter is organized as follows. First the JMLS and the CM structure will be described in Section 2.2 together with the estimation objectives of the MM estimation algorithms based on these two structures. Subsequently, in Section 2.3 the MM estimation algorithms themselves are described. Section 2.4 provides two Monte-Carlo simulation examples that have the purpose to demonstrate the advantages and disadvantages of the conventional and newly proposed model structure. Finally, Section 2.5 will end this chapter with concluding remarks.

## 2.2 Problem Formulation

Consider the following linear time-varying system

$$x_{k+1} = A(\rho_k)x_k + B(\rho_k)u_k + Q(\rho_k)^{1/2}w_k, \quad (2.1)$$

$$y_k = C(\rho_k)x_k + D(\rho_k)u_k + R(\rho_k)^{1/2}v_k, \quad (2.2)$$

where  $x_k \in \mathbb{R}^n$  is the state,  $u_k \in \mathbb{R}^m$  is the input,  $y_k \in \mathbb{R}^\ell$  is the output,  $w_k \in \mathbb{R}^n$  is the process noise and  $v_k \in \mathbb{R}^\ell$  is the measurement noise. Both  $v_k$  and  $w_k$  are assumed to be zero-mean Gaussian white noise sequences with unit variance.  $A(\rho_k)$ ,  $B(\rho_k)$ ,  $C(\rho_k)$ , and  $D(\rho_k)$  are the system matrices that depend on parameter  $\rho_k$ .  $Q(\rho_k)$  and  $R(\rho_k)$  are noise covariance matrices that also depend on  $\rho_k$ . The parameter  $\rho_k$  can take values in the bounded set  $\mathcal{R}$ . Although the set  $\mathcal{R}$  is bounded, the parameter  $\rho_k$  can have infinitely many values. Let the infinite set of models defined by (2.1)-(2.2) for all  $\rho_k \in \mathcal{R}$  be denoted by  $\mathcal{M}$  and let  $\mathcal{M}$  be represented by the dashed area in Figure 2.1. The goal of MM methods is to approximate  $\mathcal{M}$  by as few models as possible. The JMLS is often used for approximating  $\mathcal{M}$ . In Figure 2.1, the JMLS model set chosen to approximate the dashed area consists of 6 local models. These local models are represented by stars and denoted by  $\mathfrak{M}^{(1)} - \mathfrak{M}^{(6)}$ . Furthermore, model  $\mathfrak{M}^{(c)} \in \mathcal{M}$  is also depicted. If this model corresponds to the true system, then MM estimation algorithms based on the JMLS with model set  $\mathfrak{M}^{(1)} - \mathfrak{M}^{(6)}$  perform less well. In order to maintain performance in this case,  $\mathfrak{M}^{(c)}$  should be added to the existing model set. This principle can lead to large model sets in practice, which is not desirable because of the increased computational

cost. However, it is also possible to represent  $\mathfrak{M}^{(c)}$  by a convex combination of  $\mathfrak{M}^{(1)} - \mathfrak{M}^{(6)}$ . If an alternative model structure is adopted that allows convex combinations of models, then all models from  $\mathcal{M}$  could be represented with the same model set as the JMLS in this case. In this section first the JMLS and its estimation objectives will be described. Subsequently, an alternative structure that allows convex combinations of models is proposed. Also for this alternative structure estimation objectives are given.

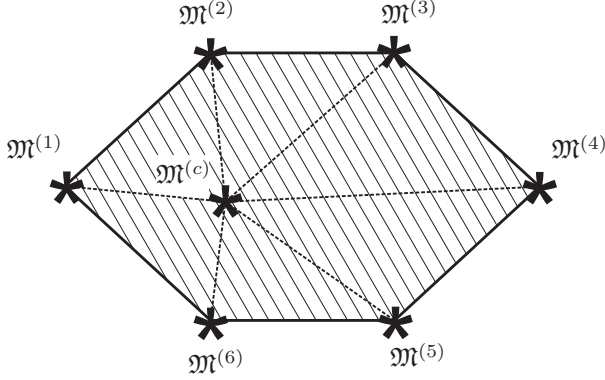


Figure 2.1: Schematic diagram of a multiple-model system.

### 2.2.1 Hybrid Model Structure

The MM system that is commonly adopted for MM estimation algorithms has a hybrid model structure. This structure consists of a set of local models, only one of which is active at a time. Furthermore, a hybrid MM is characterized by sudden switches between the local models and has the following form

$$x_{k+1} = A^{(s_k)}x_k + B^{(s_k)}u_k + (Q^{(s_k)})^{1/2}w_k, \quad (2.3)$$

$$y_k = C^{(s_k)}x_k + D^{(s_k)}u_k + (R^{(s_k)})^{1/2}v_k, \quad (2.4)$$

where  $A^{(s_k)}$ ,  $B^{(s_k)}$ ,  $C^{(s_k)}$ ,  $D^{(s_k)}$ ,  $Q^{(s_k)}$ , and  $R^{(s_k)}$  are system matrices, which are functions of the first-order homogenous Markov chain  $s_k$ , referred to as the system mode.  $w_k$  and  $v_k$  represent unit variance white noise sequences with means  $\mathbf{w}_k$  and  $\mathbf{v}_k$ , respectively.  $Q^{(s_k)}$  and  $R^{(s_k)}$  are the noise covariance matrices. The system mode  $s_k \in \mathbb{Z}^+$  has  $N$  states and the entries of the transition probability matrix  $\Pi \in \mathbb{R}^{N \times N}$  are given by

$$\Pi_{ij} = P(s_{k+1} = j | s_k = i), \quad (2.5)$$

where  $i, j \in \mathbf{S}$ , with  $\mathbf{S} = \{1, 2, \dots, N\}$ , and the notation  $P(A|B)$  denotes the conditional probability density of "A given B" for discrete random variables. The system defined by (2.3)-(2.4) combined with (2.5) is known as a JMLS.

The estimation objective of most MM estimation algorithms based on a JMLS is to obtain minimum mean square error (MMSE) estimates of both the state and the mode of the JMLS (Doucet et al. 2001; Li and Jilkov 2005). There are also MM estimation algorithms that use the maximum a posteriori (MAP) criterion. For example, Logothetis and Krishnamurthy (1999) have proposed a non-recursive algorithm for MAP estimation, which is followed up by a recursive MAP estimation algorithm (Johnston and Krishnamurthy 2001). In this chapter, the MMSE estimate of the state is considered. This estimate can be computed as (Bar-Shalom et al. 2001)

$$\hat{x}_{k|k} = E[x_k|Y_k] = \sum_{n=1}^{N^k} E[x_k|M^{(k,n)}, Y_k]P(M^{(k,n)}|Y_k), \quad (2.6)$$

where  $Y_k = [y_k \ y_{k-1} \ \cdots \ y_0]$ ,  $E[\cdot]$  denotes the expectation of a random variable, and  $M^{(k,n)}$  denotes the  $n$ -th possible mode sequence of  $N^k$  sequences at time step  $k$ . It can be seen that evaluating (2.6) is a very computationally intensive task since the number of possible mode sequences  $N^k$  to be considered increases exponentially with time. Therefore, most MM estimation algorithms apply approximations to deal with this complexity. Different strategies exist for making such approximations resulting in different filters. These strategies differ in the way in which all possible mode scenarios in the past are used for estimating the mode at the current time step. An elaborate explanation of these different strategies is given by Li and Jilkov (2005). In this chapter the approximation of the IMM filter for obtaining the MMSE estimate of the state is considered. This approximation is given by:

$$\hat{x}_{k|k} = E[x_k|Y_k] \approx \sum_{i=1}^N E[x_k|s_k = i, Y_k]\mu_k^{(i)}, \quad (2.7)$$

where  $\mu_k^{(i)} = P(s_k = i|Y_k)$  is the model weight of the  $i$ -th model. The weight vector  $\mu_k$  is composed of the individual model weights as  $\mu_k = [\mu_k^{(1)} \ \mu_k^{(2)} \ \cdots \ \mu_k^{(N)}]^T$ . Evaluation of (2.7) can be performed in a much more computationally efficient manner than (2.6). This reduction of the computational effort has been obtained by not considering all possible mode sequences in the past (as was the case in (2.6)) but instead considering only each possible current model based on the information from the previous time step.

### 2.2.2 Convex Model Structure

An alternative to the hybrid system defined in (2.3)-(2.5) is given by

$$x_{k+1} = \sum_{i=1}^N \mu_k^{(i)} [A^{(i)} x_k + B^{(i)} u_k] + Q_k^{1/2} w_k, \quad (2.8)$$

$$y_k = \sum_{i=1}^N \mu_k^{(i)} [C^{(i)} x_k + D^{(i)} u_k] + R_k^{1/2} v_k, \quad (2.9)$$

$$\mu_k^{(i)} \geq 0, \quad \sum_{i=1}^N \mu_k^{(i)} = 1, \quad (2.10)$$

where  $\mu_k^{(i)}$  is the model weight corresponding to the  $i$ -th model represented by the state space quadruple  $\{A^{(i)}, B^{(i)}, C^{(i)}, D^{(i)}\}$ ,  $w_k$  and  $v_k$  are mutually uncorrelated zero-mean Gaussian white noise sequences with unit variance,  $Q_k$  and  $R_k$  are covariance matrices, and  $N$  is the number of local models. The model structure defined by (2.8)-(2.10) is named the CM structure because of the convex combination of the local models in this structure. The CM structure explicitly uses weighted combinations of the local models contrary to the hybrid model structure. Note also that the CM structure allows modeling of the same systems as the JMLS.

The objective for MM estimation using the CM structure is to estimate both the state and the model weights. This objective is defined by means of the following optimization problem

$$(\hat{\mu}_k, \hat{x}_{k|k}) = \arg \max_{\mu_k, x_k} p(\mu_k, x_k | y_k), \quad (2.11)$$

where the notation  $p(A|B)$  denotes the conditional probability density of “A given B” for continuous random variables. The result of (2.11) is a MAP estimate of the desired parameters. However, because the considered noise sequences in the CM structure have a Gaussian distribution, the MAP estimate also corresponds to the MMSE estimate. The reason for this is that the point at which the Gaussian distribution has maximum probability (location of the MAP optimum) corresponds to its center of probability mass (location of the MMSE optimum).

## 2.3 Multiple-Model Estimation Algorithms

In this section, three algorithms for MM estimation will be described. One of these algorithms is the IMM filter. The IMM filter is probably the most popular solution for the problem of estimating the state and mode of a JMLS. Although there are many variants of the IMM filter, each of which is aimed at improving performance in specific (practical) situations, the main core of these variants is the same. Therefore, the basic version of the IMM filter is considered as the benchmark algorithm for MM estimation using a JMLS. It will be summarized in the following section.



For the CM structure two MM estimation algorithms will be described that provide an approximate solution for (2.11). These filters differ in the sequence of estimation of the state and the weights. The EKF estimates the state and weights at the same time, while the dual CM filter (CMF) estimates the state and weights separately in two steps.

### 2.3.1 IMM Filter

The IMM filter belongs to the second category of a set of three categories (Li and Jilkov 2005). It was preceded by the autonomous MM (AMM) filter in the first category. The third category is the category of the variable structure MM (VSMM) filter. The AMM filter is characterized by the fact that there is no interaction between the different filters, each of which is based on a different local model from the model set. Interaction between filters was introduced in the second category, which explains the name of the IMM filter. In addition to this, the third category allows for a time-varying model set. This means that the model set can contain a different number of models at different times.

The IMM filter consists of four steps that are performed in each cycle of the filter to compute (2.7). These four steps are:

1. Model-conditioned re-initialization
2. Model-conditioned filtering
3. Model probability update
4. Fusion of estimates

In Table 2.1, the steps that are performed in one cycle of the IMM filter are described. For convenience, throughout this table the notation  $A^{(s_k=j)} = A_k^{(j)}$  is used for the system matrices. In the first step, mixing of the estimated states  $\hat{x}_{k|k}^{(j)}$  and error covariances  $P_{k|k}^{(j)}$  takes place. The transition probability matrix  $\Pi$  forms the basis for this mixing step. In the second step, one cycle of the Kalman filter is performed for each local model. In the third step, the likelihood for each local model is computed based on the probability distribution, which is assumed to be Gaussian

$$L_k^{(j)} \triangleq p(y_k^{(j)} | s_{(j)}^k, Y_{k-1}) \stackrel{\text{assume}}{=} \mathcal{N}(\tilde{z}_k^{(j)}; 0, \Sigma_k^{(j)}), \quad (2.12)$$

where  $s_{(j)}^k$  is the mode sequence  $\{s_1 = j, \dots, s_k = j\}$  and  $\mathcal{N}(z; \bar{z}, \Sigma) = \exp[-(z - \bar{z})^T \Sigma^{-1} (z - \bar{z}) / 2] / (\sqrt{|2\pi\Sigma|})$ . In this relation, the determinant of matrix  $2\pi\Sigma$  is denoted by  $|2\pi\Sigma|$ . In the fourth and final step, the fusion of the local estimates of the state vector and covariance matrix takes place in order to obtain the overall values of these two quantities. The fusion is based on the model weights computed in the third step of the algorithm. Note that the overall values of the state and covariance are not used in the next cycle of the filter. The main tuning parameter of the IMM filter is the transition probability matrix  $\Pi$ .



**Table 2.1:** One cycle of the IMM filter (Li and Jilkov 2005).**1. Model-conditioned re-initialization** (for  $j = 1, 2, \dots, N$ )

$$\begin{aligned}
\text{predicted probability:} \quad & \mu_{k|k-1}^{(j)} = \sum_{i=1}^N \Pi_{ij} \mu_{k-1}^{(i)} \\
\text{mixing weight:} \quad & \mu_{k-1}^{i|j} = \Pi_{ij} \mu_{k-1}^{(i)} / \hat{\mu}_{k-1}^j \\
\text{mixing estimate:} \quad & \bar{x}_{k-1|k-1}^{(j)} = \sum_{i=1}^N \hat{x}_{k-1|k-1}^{(i)} \mu_{k-1}^{i|j} \\
\text{mixing covariance:} \quad & \bar{P}_{k-1|k-1}^{(j)} = \sum_{i=1}^N \mu_{k-1}^{i|j} [(\bar{x}_{k-1|k-1}^{(j)} - \hat{x}_{k-1|k-1}^{(i)}) \\
& \quad \times (\bar{x}_{k-1|k-1}^{(j)} - \hat{x}_{k-1|k-1}^{(i)})^T + P_{k-1|k-1}^{(i)}]
\end{aligned}$$

**2. Model-conditioned filtering** (for  $j = 1, 2, \dots, N$ )

$$\begin{aligned}
\text{predicted state:} \quad & \hat{x}_{k|k-1}^{(j)} = A_{k-1}^{(j)} \bar{x}_{k-1|k-1}^{(j)} + B_{k-1}^{(j)} u_{k-1} + Q_{k-1}^{(j)} \mathbf{w}_{k-1}^{(j)} \\
\text{predicted covariance:} \quad & P_{k|k-1}^{(j)} = A_{k-1}^{(j)} \bar{P}_{k-1|k-1}^{(j)} (A_{k-1}^{(j)})^T + Q_{k-1}^{(j)} \\
\text{measurement residual:} \quad & \tilde{z}_k^{(j)} = y_k - C_k^{(j)} \hat{x}_{k|k-1}^{(j)} - D_k^{(j)} u_k - R_k^{(j)} \mathbf{v}_k^{(j)} \\
\text{residual covariance:} \quad & \Sigma_k^{(j)} = C_k^{(j)} (C_k^{(j)})^T + R_k^{(j)} \\
\text{filter gain:} \quad & K_k^{(j)} = P_{k|k-1}^{(j)} (C_k^{(j)})^T (\Sigma_k^{(j)})^{-1} \\
\text{updated state:} \quad & \hat{x}_{k|k}^{(j)} = \hat{x}_{k|k-1}^{(j)} + K_k^{(j)} \tilde{z}_k^{(j)} \\
\text{updated covariance:} \quad & P_{k|k}^{(j)} = P_{k|k-1}^{(j)} - K_k^{(j)} \Sigma_k^{(j)} (K_k^{(j)})^T
\end{aligned}$$

**3. Model probability update** (for  $j = 1, 2, \dots, N$ )

$$\begin{aligned}
\text{model likelihood:} \quad & L_k^{(j)} \stackrel{\text{assume}}{=} \frac{\exp[-(1/2)(\tilde{z}_k^{(j)})^T (\Sigma_k^{(j)})^{-1} \tilde{z}_k^{(j)}]}{|2\pi \Sigma_k^{(j)}|^{1/2}} \\
\text{model probability:} \quad & \mu_k^{(j)} = \frac{\mu_{k|k-1}^{(j)} L_k^{(j)}}{\sum_{i=1}^N \mu_{k|k-1}^{(i)} L_k^{(i)}}
\end{aligned}$$

**4. Fusion of estimates**

$$\begin{aligned}
\text{overall state:} \quad & \hat{x}_{k|k} = \sum_{j=1}^N \hat{x}_{k|k}^{(j)} \mu_k^{(j)} \\
\text{overall covariance:} \quad & P_{k|k} = \sum_{j=1}^N [P_{k|k}^{(j)} + (\hat{x}_{k|k} - \hat{x}_{k|k}^{(j)})(\hat{x}_{k|k} - \hat{x}_{k|k}^{(j)})^T] \mu_k^{(j)}
\end{aligned}$$

**2.3.2 Dual CMF**

In the following, an objective function is derived that must be minimized to solve the optimization problem formulated in (2.11). Using Bayes' rule, the joint condi-

tional density  $p(\mu_k, x_k | y_k)$  from (2.11) can be expanded as

$$\begin{aligned} p(\mu_k, x_k | y_k) &= \frac{p(y_k | x_k, \mu_k) p(x_k, \mu_k)}{p(y_k)}, \\ &= \frac{p(y_k | x_k, \mu_k) p(x_k | \mu_k) p(\mu_k)}{p(y_k)}. \end{aligned} \quad (2.13)$$

Next, assuming that process and measurement noise have a Gaussian distribution, the following relation (Wan and Nelson 2001) can be derived

$$p(y_k, x_k | \mu_k) = p(y_k | x_k, \mu_k) p(x_k | \mu_k), \quad (2.14)$$

which can be substituted in (2.13) to obtain

$$p(\mu_k, x_k | y_k) = \frac{p(y_k, x_k | \mu_k) p(\mu_k)}{p(y_k)}. \quad (2.15)$$

The problem of maximizing  $p(\mu_k, x_k | y_k)$ , i.e. solving the problem formulated in (2.11), boils down to maximizing  $p(y_k, x_k | \mu_k)$  with respect to  $\mu_k$  and  $x_k$ . The reason for this is that  $p(y_k)$  is a function of neither  $x_k$  nor  $\mu_k$  and furthermore the term  $p(\mu_k)$  can also be discarded if it is assumed that there is no prior information available on the distribution of the weights (Nelson and Wan 1998). This assumption is motivated by the observation that the transition probability matrix in a JMLS (i.e. the prior information on the weights) in practice is often not known in advance and is therefore considered to be a tuning parameter (Jilkov and Li 2004). The same lack of a priori information is also assumed for the CM structure.

An expansion of (2.14) results in the following relation

$$\begin{aligned} p(y_k, x_k | \mu_k) &= \overbrace{\frac{1}{\sqrt{|2\pi Q_k|}} \exp \left( -\frac{1}{2} (x_k - x_k^-)^T Q_k^{-1} (x_k - x_k^-) \right)}^{p(x_k | \mu_k)} \\ &\times \underbrace{\frac{1}{\sqrt{|2\pi R_k|}} \exp \left( -\frac{1}{2} (y_k - \tilde{y}_k)^T R_k^{-1} (y_k - \tilde{y}_k) \right)}_{p(y_k | x_k, \mu_k)}, \end{aligned} \quad (2.16)$$

where

$$x_k^- = A_{\mu_k} \hat{x}_{k-1|k-1} + B_{\mu_k} u_{k-1}, \quad (2.17)$$

$$\tilde{y}_k = C_{\mu_k} x_k + D_{\mu_k} u_k, \quad (2.18)$$

and

$$A_{\mu_k} = \sum_{i=1}^N \mu_k^{(i)} A^{(i)}, \quad B_{\mu_k} = \sum_{i=1}^N \mu_k^{(i)} B^{(i)}, \quad (2.19)$$

$$C_{\mu_k} = \sum_{i=1}^N \mu_k^{(i)} C^{(i)}, \quad D_{\mu_k} = \sum_{i=1}^N \mu_k^{(i)} D^{(i)}. \quad (2.20)$$

The notation from (2.19)-(2.20) will also be used in the remainder of the chapter in a similar manner for indices other than  $\mu_k$ . Taking the logarithm of (2.16), dropping the terms that are independent of both  $\mu_k$  and  $x_k$ , and changing the sign, reduces it to the following objective function

$$J(x_k, \mu_k) = (x_k - x_k^-)^T Q_k^{-1} (x_k - x_k^-) + (y_k - \tilde{y}_k)^T R_k^{-1} (y_k - \tilde{y}_k). \quad (2.21)$$

Now, (2.11) can be rewritten as the following optimization problem

$$(\hat{\mu}_k, \hat{x}_{k|k}) = \arg \min_{\mu_k, x_k} J(x_k, \mu_k). \quad (2.22)$$

This problem is nonlinear (and nonconvex) due to the product of  $x_k$  and  $\mu_k$  that is present in  $J(x_k, \mu_k)$ . In general, solving problem (2.22) can be done by using two approaches. These approaches can be *decoupled* or *direct* (Wan and Nelson 2001). In the direct approach, both  $x_k$  and  $\mu_k$  are estimated jointly by solving a multivariate nonlinear optimization problem. A filter that uses the direct approach will be described in Section 2.3.3. The principle of decoupled approaches is to optimize with respect to one variable at a time, while keeping the other variable fixed and vice versa. Filters based on the decoupled approach are also referred to as dual filters (Wan and Nelson 2001).

In the following, a filter that uses the decoupled approach is described. The decoupled filter, which is given the name dual CMF, uses two linear filtering steps to minimize  $J(x_k, \mu_k)$ . In the first step of the dual CMF, the weight vector is assumed to be fixed and it is chosen to be the current estimate, i.e.  $\mu_k = \hat{\mu}_{k-1}$ . This assumption leads to the following objective function

$$J_x(x_k, \hat{\mu}_{k-1}) = (x_k - \bar{x}_k)^T Q_k^{-1} (x_k - \bar{x}_k) + (y_k - \bar{y}_k)^T R_k^{-1} (y_k - \bar{y}_k). \quad (2.23)$$

This objective function forms the basis for the optimization problem

$$\begin{aligned} \hat{x}_{k|k} &= \arg \min_{x_k} J_x(x_k, \hat{\mu}_{k-1}), \\ \text{s.t. } \bar{x}_k &= A_{\hat{\mu}_{k-1}} \hat{x}_{k-1|k-1} + B_{\hat{\mu}_{k-1}} u_{k-1}, \\ \bar{y}_k &= C_{\hat{\mu}_{k-1}} x_k + D_{\hat{\mu}_{k-1}} u_k, \end{aligned} \quad (2.24)$$

the solution of which is the state estimate.

The solution of (2.24) can be computed by applying one step of the Kalman filter to the CM structure in which the current estimates of the weights are used

(i.e.  $\mu_k = \hat{\mu}_{k-1}$ ). The best way to see that this can indeed be done is to rewrite the Kalman filter as a weighted least squares problem. This is done, for example, by Verhaegen and Verdult (2007). The solution of (2.24) has been obtained in Step 1 of Table 2.2 by using one iteration of the square root covariance filter (SRCF) implementation of the Kalman filter. The SRCF is a numerically more robust implementation of the Kalman filter. Kanev and Verhaegen (2005) give a detailed treatment of the SRCF. In Step 1, the matrices  $S_{k|k}$  and  $S_{k|k-1}$  correspond to the error covariance matrices defined as

$$S_{k|k} S_{k|k}^T = E[(x_k - \hat{x}_{k|k})(x_k - \hat{x}_{k|k})^T], \quad (2.25)$$

$$S_{k|k-1} S_{k|k-1}^T = E[(x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})^T]. \quad (2.26)$$

In Step 2 the state is assumed to be fixed and it is chosen to be the current estimate, i.e.  $x_k = \hat{x}_{k|k}$ . This assumption leads to the following objective function

$$J_\mu(\hat{x}_{k|k}, \mu_k) = (\hat{x}_{k|k} - x_k^-)^T Q_k^{-1} (\hat{x}_{k|k} - x_k^-) + (y_k - \check{y}_k)^T R_k^{-1} (y_k - \check{y}_k). \quad (2.27)$$

This objective function forms the basis for the optimization problem

$$\begin{aligned} \hat{\mu}_k &= \arg \min_{\mu_k} J_\mu(\hat{x}_{k|k}, \mu_k), \\ \text{s.t. } \check{y}_k &= C_{\mu_k} \hat{x}_{k|k} + D_{\mu_k} u_k, \end{aligned} \quad (2.28)$$

the solution of which is the estimate of the model weight. This solution is computed in Step 2 of the dual CMF. In this step, one iteration of a (square root information) recursive least squares estimator (Verhaegen 1989) is used to obtain  $\hat{\mu}_k$ . This estimator has been implemented with a forgetting factor  $\lambda \in [0, 1]$ . The forgetting factor determines to what extent old data is taken into account. In case  $\lambda = 1$ , the recursive least squares estimator equally weights all data from the past. The smaller  $\lambda$  is chosen, the more past data is “forgotten”. The forgetting factor  $\lambda$  forms the main tuning parameter of the dual CMF.

**Table 2.2:** The dual CMF.

**1. State estimation:** Given  $A_{\hat{\mu}_{k-1}}, B_{\hat{\mu}_{k-1}}, C_{\hat{\mu}_{k-1}}, D_{\hat{\mu}_{k-1}}, Q_k, R_k, S_{k|k-1}, \hat{x}_{k|k-1}, u_k$ , and  $y_k$ , compute:

- Using the  $QR$ -decomposition, find the orthogonal matrix  $U_1$  and matrices  $R_e, G_{k,1}, G_{k,2}, S_{k|k}, V_{k,1}$ , and  $V_{k,2}$ , such that

$$\begin{bmatrix} R_e & 0 & 0 \\ G_{k,1} & S_{k|k} & 0 \\ G_{k,2} & V_{k,1} & V_{k,2} \end{bmatrix} = \begin{bmatrix} C_{\hat{\mu}_{k-1}} S_{k|k-1} & -R_k^{1/2} & 0 \\ S_{k|k-1} & 0 & 0 \\ A_{\hat{\mu}_{k-1}} S_{k|k-1} & 0 & Q_k^{1/2} \end{bmatrix} U_1 \quad (2.29)$$

- Compute

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} - G_{k,1} R_e^{-1} (C_{\hat{\mu}_{k-1}} \hat{x}_{k|k-1} + D_{\hat{\mu}_{k-1}} u_k - y_k) \quad (2.30)$$

- Using the  $QR$ -decomposition, find orthogonal matrix  $U_2$  and matrix  $S_{k+1|k}$ , such that

$$[S_{k+1|k} \ 0] = [V_{k,1} \ V_{k,2}] U_2 \quad (2.31)$$

- Compute

$$\begin{aligned} \hat{x}_{k+1|k} &= A_{\hat{\mu}_{k-1}} \hat{x}_{k|k-1} + B_{\hat{\mu}_{k-1}} u_k \\ &\quad - G_{k,2} R_e^{-1} (C_{\hat{\mu}_{k-1}} \hat{x}_{k|k-1} + D_{\hat{\mu}_{k-1}} u_k - y_k) \end{aligned} \quad (2.32)$$

**2. Model weight estimation:** Given  $\begin{bmatrix} A^{(1)} & B^{(1)} \\ C^{(1)} & D^{(1)} \end{bmatrix} \cdots \begin{bmatrix} A^{(N)} & B^{(N)} \\ C^{(N)} & D^{(N)} \end{bmatrix}, \hat{x}_{k-1|k-1}, \hat{x}_{k|k}, \hat{\xi}_{k-1}, T_{k-1}, \lambda, u_{k-1}, u_k$ , and  $y_k$ , compute:

- Let

$$\Phi = \begin{bmatrix} Q_k^{-1} \left[ \begin{bmatrix} A^{(1)} & B^{(1)} \end{bmatrix} \begin{bmatrix} \hat{x}_{k-1|k-1} \\ u_{k-1} \end{bmatrix} \cdots \begin{bmatrix} A^{(N)} & B^{(N)} \end{bmatrix} \begin{bmatrix} \hat{x}_{k-1|k-1} \\ u_{k-1} \end{bmatrix} \right] \\ R_k^{-1} \left[ \begin{bmatrix} C^{(1)} & D^{(1)} \end{bmatrix} \begin{bmatrix} \hat{x}_{k|k} \\ u_k \end{bmatrix} \cdots \begin{bmatrix} C^{(N)} & D^{(N)} \end{bmatrix} \begin{bmatrix} \hat{x}_{k|k} \\ u_k \end{bmatrix} \right] \end{bmatrix} \quad (2.33)$$

and

$$\Gamma = \begin{bmatrix} Q_k^{-1} \hat{x}_{k|k} \\ R_k^{-1} y_k \end{bmatrix}. \quad (2.34)$$

Using the  $QR$ -decomposition, find an orthogonal matrix  $U_3$  and matrices  $T_k, \hat{\xi}_k$ , and  $\epsilon_k$ , such that

$$\begin{bmatrix} T_k & \hat{\xi}_k \\ 0 & \epsilon_k \end{bmatrix} = U_3 \begin{bmatrix} \sqrt{\lambda} T_{k-1} & \sqrt{\lambda} \hat{\xi}_{k-1} \\ \Phi & \Gamma \end{bmatrix} \quad (2.35)$$

- Compute

$$\hat{\mu}_k = T_k^{-1}(\hat{\xi}_k) \quad (2.36)$$

### Covariance propagation

In the two previously described steps of the dual CMF, a direct substitution of the current estimates of the weights and states into the objective function (2.21) is performed. With this direct substitution the uncertainty of the current estimates is overlooked. If this uncertainty were also to be considered, in the first step the substitution  $\mu_k = \hat{\mu}_{k-1} - S_{k-1}^\mu \tilde{\mu}_k$  should have been made. In this substitution, the uncertainty of  $\hat{\mu}_{k-1}$  is expressed by the term  $S_{k-1}^\mu \tilde{\mu}_k$  consisting of the error covariance matrix  $S_{k-1}^\mu = (T_{k-1}^T T_{k-1})^{-1}$  and a zero-mean white noise signal  $\tilde{\mu}_k$ . For the second step of the dual CMF, the substitution  $x_k = \hat{x}_{k|k} - S_{k|k} \tilde{x}_k$  should have been made. In this substitution  $\tilde{x}_k$  represents a zero-mean white noise signal. With these two modified substitutions the two steps of the dual CMF would have changed to nonlinear estimation steps due to cross-products of the introduced noise terms and the quantities to be estimated. Therefore, in this chapter the direct substitution of current estimates is preferred as is also done by e.g. Wan and Nelson (2001). Nelson (2000) also attempts to incorporate the errors of the estimates. This attempt resulted in modified versions of (2.24) and (2.28), which had to be linearized to allow the two estimation steps to be linear again. However, it was concluded that the performance of the resulting algorithm was not better than the algorithm that used direct substitution of the current estimates.

### Constraint implementation

The algorithm described in Table 2.2 does not take the constraints from (2.10) into account. This can be done in an additional step by using the theory for implementing equality and inequality constraints in Kalman filters as is described by Simon and Chia (2002) and Simon and Simon (2006), respectively. For implementing the equality constraint, the projection method described by Simon and Chia (2002) is used. This method directly projects the unconstrained estimate  $\hat{\mu}_k$  onto the constraint surface to obtain the constrained estimate  $\hat{\mu}_k^c$ . This problem is formulated as follows

$$\min_{\hat{\mu}_k^c} (\hat{\mu}_k^c - \hat{\mu}_k)^T W (\hat{\mu}_k^c - \hat{\mu}_k) \text{ such that } D\hat{\mu}_k^c = d, \quad (2.37)$$

where  $W$  is a symmetric positive definite weighting matrix, which is chosen to be an identity matrix. The solution to this problem is given by

$$\hat{\mu}_k^c = \hat{\mu}_k - W^{-1} D^T (D W^{-1} D^T)^{-1} (D \hat{\mu}_k - d). \quad (2.38)$$

The inequality constraint can be also cast as an equality constraint if it is a priori known that the inequality constraint indeed holds. This idea was also suggested by Simon and Simon (2006). For estimating  $\hat{\mu}_k^c$  this means that first it has to be determined for which  $i$  it holds that  $\hat{\mu}_k^{(i)} < 0$ . Subsequently, for these  $i$ 's an extra equality constraint is added in the form of  $\hat{\mu}_k^{(i)} = 0$  to the already existing equality constraint  $\sum_{k=1}^N \mu_k^{(i)} = 1$ . With this augmented set of constraints, a constrained estimate  $\hat{\mu}_k^c$  is computed. This constrained estimate is to be checked again for  $\hat{\mu}_k^{(i)}$ 's

that violate the inequality constraint. If this is the case, then a new constrained estimate should be computed in the same way as the previous one. This sequence is repeated until the whole constrained weight vector  $\hat{\mu}_k^c$  satisfies the constraints.

### 2.3.3 EKF

The nonlinear problem of estimating both the state and weights as is formulated in (2.11) has been reformulated as (2.22). In the previous section this objective function is minimized in a decoupled fashion. In this section the direct approach in which the state and model weights are jointly estimated is adopted. A novel algorithm for this nonlinear estimation problem is proposed by Hallouzi et al. (2006a). The main feature of this algorithm is that it reduces the number of parameters in the nonlinear estimation problem. In this algorithm only the weight vector is estimated in a nonlinear estimation step. The state is estimated in a linear estimation step. Because the nonlinear estimation step can become computationally intensive, especially if the model set is large, another type of algorithm will be considered in this chapter. This algorithm uses linearization as an approximation to the nonlinear problem. This approximation is performed by an EKF. In this filter, the state is augmented with the weights. At each time step the nonlinear augmented state-space model is linearized around the current estimate. This approach is quite common and is proposed for example by Bar-Shalom et al. (2001) and Ljung (1979) for general parameter estimation problems. In the augmented state, the weights are assumed to evolve as a random walk process

$$\mu_k = \mu_{k-1} + w_{\mu_k}, \quad (2.39)$$

where  $w_{\mu_k}$  is a zero-mean Gaussian white noise sequence with covariance matrix  $Q_k^\mu$ . The resulting augmented state model obeys the following equations

$$\begin{aligned} \overbrace{\begin{bmatrix} x_k \\ \mu_k \end{bmatrix}}^{z_k} &= \begin{bmatrix} A_{\mu_{k-1}} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \mu_{k-1} \end{bmatrix} + \begin{bmatrix} B_{\mu_{k-1}} \\ 0 \end{bmatrix} u_{k-1} \\ &\quad + \underbrace{\begin{bmatrix} Q_{k-1}^{1/2} & 0 \\ 0 & (Q_{k-1}^\mu)^{1/2} \end{bmatrix}}_{(Q_{k-1}^a)^{1/2}} \bar{w}_{k-1}, \end{aligned} \quad (2.40)$$

$$y_k = [C_{\mu_k} \ 0] \begin{bmatrix} x_k \\ \mu_k \end{bmatrix} + D_{\mu_k} u_k + R_k^{1/2} \bar{v}_k, \quad (2.41)$$

where  $\bar{w}_{k-1}$  and  $\bar{v}_k$  are zero-mean white noise sequences of appropriate dimensions with unit variance. The augmented state and covariance matrix are denoted by  $z_k$  and  $Q_{k-1}^a$ , respectively. Let (2.40) and (2.41) be represented by the following

shorthand notation

$$z_k = F(z_{k-1}, u_{k-1}, \bar{w}_{k-1}), \quad (2.42)$$

$$y_k = H(z_k, u_k, \bar{v}_k), \quad (2.43)$$

then the Jacobian matrices required for the EKF can be computed as

$$\bar{A}_{k-1} = \left. \frac{\partial F}{\partial z} \right|_{z=\hat{z}_{k-1|k-1}}, \quad (2.44)$$

$$\bar{C}_k = \left. \frac{\partial H}{\partial z} \right|_{z=\hat{z}_{k|k-1}}. \quad (2.45)$$

The complete EKF algorithm is described in Table 2.3. The main tuning parameter of the EKF is the covariance matrix  $Q_k^\mu$ . The convexity constraints can be included in the same manner as is explained in Section 2.3.2 for the dual CMF.

### 2.3.4 Properties of Multiple-Model Filters

At this stage three different MM filters have been described. The IMM filter, which uses the hybrid model structure, and the dual CMF and EKF, which use the CM structure. In Table 2.4 an overview is given of the most important properties of the described filters. One of the main differences between the IMM filter and the filters based on the CM structure is the number of local filters. Contrary to the IMM, which uses  $N$  local filters, the CM based filters use only one global filter. The main tuning parameter of the IMM filter is the transition probability matrix  $\Pi$ . In practice, tuning this parameter might prove to be a difficult task (Jilkov and Li 2004). The counterparts of  $\Pi$  in the other two filters are  $\lambda$  and  $Q_k^\mu$ . These parameters are much easier to tune in practice since  $\lambda$  is a single scalar and  $Q_k^\mu$  can be chosen as a diagonal matrix of which the size of its entries relative to  $Q_k$  is of importance. More information on how to choose the tuning parameters of the different filters can be found in Section 2.4. Model interaction between the different local models is present in the IMM filter only as a step of the filter. There is no interaction between models in the model structure (JMLS) itself. In the other two filters there is an explicit interaction present in the CM structure.

## 2.4 Experimental Results

In order to compare the performance of the three filters described in the previous section, Monte-Carlo type simulation experiments are performed using two problem cases. Monte-Carlo simulations in the context of this section are characterized by the consideration of a number of different simulation runs, based on the same model, but each with different random noise sequences for signals such as process noise, measurement noise, and input signals. The first simulation experiment considers an FDI problem for a linearized model of the Boeing 747 benchmark model. The second simulation experiment considers a target tracking problem.



**Table 2.3:** The EKF.

Given  $\begin{bmatrix} A^{(1)} & B^{(1)} \\ C^{(1)} & D^{(1)} \end{bmatrix} \cdots \begin{bmatrix} A^{(N)} & B^{(N)} \\ C^{(N)} & D^{(N)} \end{bmatrix}$ ,  $\hat{\mu}_{k-1|k-1}$ ,  $\hat{x}_{k-1|k-1}$ ,  $Q_{k-1}^a$ ,  $R_k$ ,  $P_{k-1|k-1}$ ,  $y_k$ ,  $u_k$ , and  $u_{k-1}$ , compute:

- **Time update**

$$\hat{x}_{k|k-1} = A_{\hat{\mu}_{k-1|k-1}} \hat{x}_{k-1|k-1} + B_{\hat{\mu}_{k-1|k-1}} u_{k-1} \quad (2.46)$$

$$\hat{\mu}_{k|k-1} = \hat{\mu}_{k-1|k-1} \quad (2.47)$$

$$P_{k|k-1} = \bar{A}_{k-1} P_{k-1|k-1} \bar{A}_{k-1}^T + Q_{k-1}^a \quad (2.48)$$

- **Measurement update**

$$K_k = P_{k|k-1} \bar{C}_k^T (\bar{C}_k P_{k|k-1} \bar{C}_k^T + R_k)^{-1} \quad (2.49)$$

$$\begin{bmatrix} \hat{x}_{k|k} \\ \hat{\mu}_{k|k} \end{bmatrix} = \begin{bmatrix} \hat{x}_{k|k-1} \\ \hat{\mu}_{k|k-1} \end{bmatrix} + K_k (y_k - C_{\hat{\mu}_{k|k-1}} \hat{x}_{k|k-1} - D_{\hat{\mu}_{k|k-1}} u_k) \quad (2.50)$$

$$P_{k|k} = (I - K_k \bar{C}_k) P_{k|k-1} \quad (2.51)$$

with

$$\bar{A}_{k-1} = \begin{bmatrix} A_{\hat{\mu}_{k-1|k-1}} & [A^{(1)} \ B^{(1)}] \begin{bmatrix} \hat{x}_{k-1|k-1} \\ u_{k-1} \end{bmatrix} \cdots [A^{(N)} \ B^{(N)}] \begin{bmatrix} \hat{x}_{k-1|k-1} \\ u_{k-1} \end{bmatrix} \\ 0 & I \end{bmatrix} \quad (2.52)$$

$$\bar{C}_k = \begin{bmatrix} C_{\hat{\mu}_{k|k-1}} & [C^{(1)} \hat{x}_{k|k-1} + D^{(1)} u_k \cdots C^{(N)} \hat{x}_{k|k-1} + D^{(N)} u_k] \end{bmatrix} \quad (2.53)$$

**Table 2.4:** Overview of the properties of the described filters.

|                           | IMM    | Dual CMF  | EKF       |
|---------------------------|--------|-----------|-----------|
| Structure                 | Hybrid | CM        | CM        |
| Number of filters         | $N$    | 1         | 1         |
| Primary tuning parameters | $\Pi$  | $\lambda$ | $Q_k^\mu$ |
| Model interaction         | filter | structure | structure |

### 2.4.1 Aircraft FDI

For this simulation experiment a linearized model of the Boeing 747 benchmark is used. This model is linearized at a constant altitude of 600 m with a constant velocity of 133.8 m/s. The continuous linear time invariant model of the longitudinal dynamics of the Boeing 747 at the previously mentioned operating point is given by

$$\dot{x}(t) = Ax(t) + Bu(t) + w(t), \quad (2.54)$$

$$y(t) = Cx(t) + v(t), \quad (2.55)$$

The state of the model is given by  $x = [q \ V_{TAS} \ \alpha \ \theta \ h]^T$ , where  $q$  [rad/s] is the pitch rate,  $V_{TAS}$  [m/s] is the true airspeed,  $\alpha$  [rad] is the angle of attack,  $\theta$  [rad] is the roll angle and  $h$  [m] is the altitude. A graphical representation of these flight parameters is provided in Appendix A. The input of the model is given by  $u = [\delta_c \ \delta_s \ T_{n1} \ T_{n2} \ T_{n3} \ T_{n4}]^T$ , where  $\delta_c$  [rad] is the column deflection,  $\delta_s$  [rad] is the stabilizer deflection and  $T_{n1} - T_{n4}$  [N] represent the thrust generated by the four engines. The measurement vector  $y$  contains measurements of  $q$ ,  $V_{TAS}$ ,  $\alpha$ , and  $h$ , respectively. This means that the state  $\theta$  is not measured.

The goal of this simulation experiment is to isolate four classes of faults for the aircraft operating in the regime described by the linear model (2.54)-(2.55). These fault classes include two types of actuator faults: loss-of-effectiveness in the column and stabilizer commands. Furthermore, two classes of sensor faults are modeled: multiplicative faults in the  $V_{TAS}$  and  $\alpha$  sensors. A model set consisting of five models is used for the purpose of isolating these four classes of faults. Table 2.5 gives a description of this model set.

**Table 2.5:** Model set for MM estimation.

| Model # | Model description                         |
|---------|-------------------------------------------|
| 1       | Nominal model                             |
| 2       | Total loss-of-effectiveness of $\delta_c$ |
| 3       | Total loss-of-effectiveness of $\delta_s$ |
| 4       | Total fault in $V_{TAS}$ -sensor          |
| 5       | Total fault in $\alpha$ -sensor           |

The system matrices of the local models in the model set are given by:

$$A = \begin{bmatrix} -5.91 \cdot 10^{-1} & -1.56 \cdot 10^{-4} & -1.36 & 0 & -1.23 \cdot 10^{-6} \\ -1.47 \cdot 10^{-1} & -8.72 \cdot 10^{-3} & 4.83 & -9.81 & 5.31 \cdot 10^{-5} \\ 9.80 \cdot 10^{-1} & -1.18 \cdot 10^{-3} & -6.72 \cdot 10^{-1} & 0 & 6.83 \cdot 10^{-1} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1.34 \cdot 10^2 & 1.34 \cdot 10^2 & 0 \end{bmatrix}, \quad (2.56)$$

$$B = \begin{bmatrix} 1.41 & 2.681 & 2.08 \cdot 10^{-8} & 5.59 \cdot 10^{-8} & 5.59 \cdot 10^{-8} & 2.08 \cdot 10^{-8} \\ 0 & 0 & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} & 3.314 \cdot 10^{-6} \\ 4.70 \cdot 10^{-2} & 0 & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (2.57)$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (2.58)$$

$$B^{(2)} = \begin{bmatrix} 1.41 & \underline{0} & 2.08 \cdot 10^{-8} & 5.59 \cdot 10^{-8} & 5.59 \cdot 10^{-8} & 2.08 \cdot 10^{-8} \\ 0 & 0 & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} \\ 4.70 \cdot 10^{-2} & 0 & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (2.59)$$

$$B^{(3)} = \begin{bmatrix} \underline{0} & 2.681 & 2.08 \cdot 10^{-8} & 5.59 \cdot 10^{-8} & 5.59 \cdot 10^{-8} & 2.08 \cdot 10^{-8} \\ 0 & 0 & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} & 3.31 \cdot 10^{-6} \\ \underline{0} & 0 & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} & -2.91 \cdot 10^{-9} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (2.60)$$

$$C^{(4)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \underline{0} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad C^{(5)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \underline{0} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.61)$$

The state-space matrices for the nominal model are given in (2.56)-(2.58). The system matrices for Model 2 in the model set are the same as the nominal model except for the  $B$ -matrix, which is given by  $B^{(2)}$  in (2.59). The same holds for Model 3, for which the  $B$ -matrix is given by  $B^{(3)}$  defined in (2.60). Note that  $B^{(2)}$  and  $B^{(3)}$  are created by zeroing the relevant column. Models 4 and 5 differ from the nominal model in the  $C$ -matrices, which are defined by  $C^{(4)}$  and  $C^{(5)}$ , respectively, in (2.61). The differences of the matrices defined in (2.59)-(2.61) with respect to the nominal matrices are pointed out by an underlining of the changed entries.

For MM estimation the models in the model set are discretized by means of zero-order-hold discretization with a sample time of  $T_s = 0.01$  s. Using the discretized models a fault scenario is simulated in closed loop. A linear quadratic state feedback controller (Åström and Wittenmark 1997) synthesized for the nominal system is used in the closed loop system. This state feedback controller uses the nominal states, which means that it is assumed that the non-faulty state signal is always available for computation of the control input. This choice is motivated by the fact that the goal of this experiment is not focused on controller design, but on evaluation of the estimation performance of the proposed algorithms. Additional noise is added to the control signal to provide different input realizations for the Monte-Carlo simulations. The simulated fault scenario consists of a sequence of faults corresponding to partial occurrences of the faults modeled by the

models in the model set. These partial faults do not correspond exactly to the models from the model set. Instead, they correspond to weighted combinations of the models from the model set. Therefore, this experiment allows evaluation of the interpolation properties of the different filters. The simulated fault scenario is described in Table 2.6. Noise with a covariance of  $R = \text{diag}([10^{-5} \ 10^{-4} \ 10^{-5} \ 10^{-3}])$

**Table 2.6:** Description of the fault scenario.

| Fault description                       | Time interval (samples) |
|-----------------------------------------|-------------------------|
| 30% fault in $\alpha$ -sensor           | 200 – 300               |
| 40% loss-of-effectiveness of $\delta_s$ | 400 – 500               |
| 50% loss-of-effectiveness of $\delta_c$ | 600 – 700               |
| 20% fault in $V_{\text{TAS}}$ -sensor   | 800 – 900               |

is added to the measurements and noise with a covariance of  $Q = 10^{-6}I_5$  is added to the process. This particular choice of parameters results in a signal-to-noise ratio (SNR) of approximately 35 dB for all four measurements. The SNR of a signal is a measure for the ratio between the useful information of a signal and the noise on the signal. Given the signal  $t_k = \bar{t}_k + \tilde{t}_k$ , where  $\bar{t}_k$  corresponds to the useful information of  $t_k$  and  $\tilde{t}_k$  corresponds to the noise, then the SNR of signal  $t$  in the sample interval  $k = 1, 2, \dots, n_k$  is defined as

$$\text{SNR}(t) = 20 \log_{10} \frac{\sqrt{\frac{1}{n_k} \sum_{k=1}^{n_k} (\bar{t}_k)^2}}{\sqrt{\frac{1}{n_k} \sum_{k=1}^{n_k} (\tilde{t}_k)^2}}. \quad (2.62)$$

The tuning parameter  $\Pi$  of the IMM filter in this experiment is chosen as

$$\Pi = \begin{bmatrix} 0.9 & 0.025 & 0.025 & 0.025 & 0.025 \\ 0.1 & 0.9 & 0 & 0 & 0 \\ 0.1 & 0 & 0.9 & 0 & 0 \\ 0.1 & 0 & 0 & 0.9 & 0 \\ 0.1 & 0 & 0 & 0 & 0.9 \end{bmatrix}. \quad (2.63)$$

The reasoning behind this choice is that if the system is in the nominal (fault-free) condition, the probability of occurrence for each of the four possible faults is 0.025. Once one of the four faults has occurred, the system remains in the same fault condition with a probability of 0.9. The probability that the system returns from each of the faulty conditions to the nominal condition is 0.1. This particular choice of the transition probability matrix is inspired by a similar choice for similar local models made by Zhang and Jiang (2001). The covariance matrices for the IMM filter are chosen as  $Q_k^{\text{IMM}} = Q$  and  $R_k^{\text{IMM}} = R$  for all  $k$  and for all five models. The covariance matrices for the dual CMF are chosen as  $Q_k^{\text{DCMF}} = Q$ ,  $R_k^{\text{DCMF}} = R$  for all  $k$ , and the forgetting factor is chosen as  $\lambda = 0.85$ . The covariance matrices for the EKF are chosen as  $Q_k^{\text{EKF}} = Q$ ,  $R_k^{\text{EKF}} = R$ , and  $Q_k^\mu = 0.01I_5$  for all  $k$ . The values of the main tuning parameters of the different filters ( $\Pi$ ,  $\lambda$ , and  $Q_k^\mu$ ) are obtained

by careful tuning. Tuning of  $\lambda$  and  $Q_k^\mu$  is relatively easy: if the entries of diagonal matrix  $Q_k^\mu$  are chosen too large or  $\lambda$  is chosen too small, then the estimated model weights vary too much. However, if the diagonal entries of  $Q_k^\mu$  are chosen too small or  $\lambda$  is chosen too large (e.g.  $\lambda = 1$ ) then the estimated model weights hardly vary. So, an acceptable compromise between these two extremes must be found through tuning. For the IMM filter the tuning is not so straightforward. The transition probability matrix  $\Pi$  contains information on the probabilities of jumping from one model to other models. These probabilities can be chosen different for each jump, which makes  $\Pi$  difficult to tune. Therefore, usually a number of jumps are chosen to have the same probability as is done in this experiment.

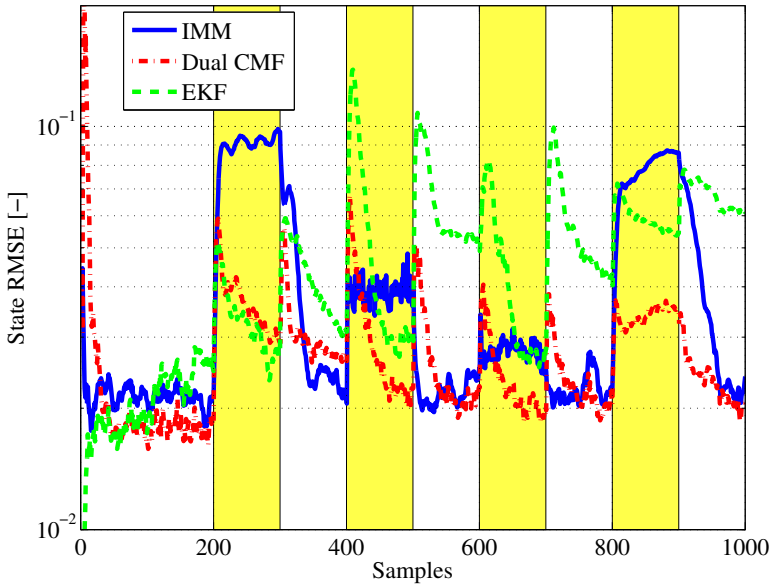
The performance of the MM estimation algorithms has been evaluated by performing 100 Monte-Carlo simulations. The root mean square error (RMSE), which is closely related to standard deviation, is used as a performance measure. The RMSE is defined as

$$\text{RMSE}(\hat{p}_k) = \sqrt{\frac{1}{N_{\text{MC}}} \sum_{i=1}^{N_{\text{MC}}} (p_k - \hat{p}_{k,i})^T (p_k - \hat{p}_{k,i})}, \quad (2.64)$$

where  $N_{\text{MC}}$  is the number of Monte-Carlo simulations,  $p_k$  is the real value of quantity  $p$  at time step  $k$  and  $\hat{p}_{k,i}$  is the estimation of  $p_k$  in the  $i$ -th Monte-Carlo simulation run.  $p_k$  can also be a vector, in which case the RMSE may only be physically interpretable if the entries of the vector have the same units.

The RMSE values for the state and the weight vector for the defined fault scenario are depicted in Figures 2.2 and 2.3, respectively. The shaded intervals in these two figures correspond to regions in which faults are injected into the system. The RMSE value for the state is unitless since the state consists of physical quantities with different units. Although the state RMSE is not physically interpretable it gives a good overview of the state estimation performance. Furthermore, the computation of the state RMSE in this way is justified by the fact that the errors in the 5 individual signals that form the state are of the same order of magnitude. In Figure 2.2 it can be observed that no clear pattern can be recognized on which to base conclusions on state estimation performance. However, Figure 2.2 is still important since it shows that the estimation performance of all three filters is acceptable; the largest RMSE error is about  $10^{-0.85}$ . Figure 2.3 gives a clearer view on how the filters perform relative to each other. In this figure it can be observed that the weight estimation performance of the two filters based on the CM structure is always better than the performance of the IMM filter in the intervals in which faults have occurred. Especially in the sample intervals [400, 500] and [600, 700], the RMSE of the weights of the IMM filter becomes relatively large. The reason for this, is that in these intervals the weight estimates switch quickly between two models (i.e. in these intervals the two models interchange a full probability of 1 in an abrupt manner) because the IMM filter has difficulties interpolating the two concerning models. Furthermore, it can be seen that the IMM filter generally outperforms the two filters based on the CM structure in the intervals without faults.

Additionally, from Figure 2.3 some insights on the convergence properties of the different filters can be gained. From the fact that the weight RMSE of the EKF becomes very small during faults, it can be concluded that the model weights estimated with the EKF get very close to the correct value for all faults. The dual CMF performs worse in this aspect, especially the first fault in the interval  $[200, 300]$  is not identified well. The same holds for the IMM filter; it does not converge to the correct weight for any of the faults. However, a benefit of the dual CMF with respect to the EKF is that it performs better in the cases in which the true system corresponds exactly to a model in the model set, i.e. the sample intervals in which no faults occur. The reason for this is that the EKF is slower to adapt to a nominal condition again after a fault has occurred. So, a general design consideration is that of the two filters based on the CM structure, the dual CMF is to be preferred when most of the expected system conditions are modeled by the models in the model set. If it is expected that many of the anticipated system conditions are represented by convex combinations of the models in the model set (such as is the case for partial faults), then the EKF is filter to be preferred. Therefore, under the conditions in this simulation example, the EKF is the filter to be preferred for isolation of the partial faults.



**Figure 2.2:** RMSE of the estimated states during the FDI scenario.

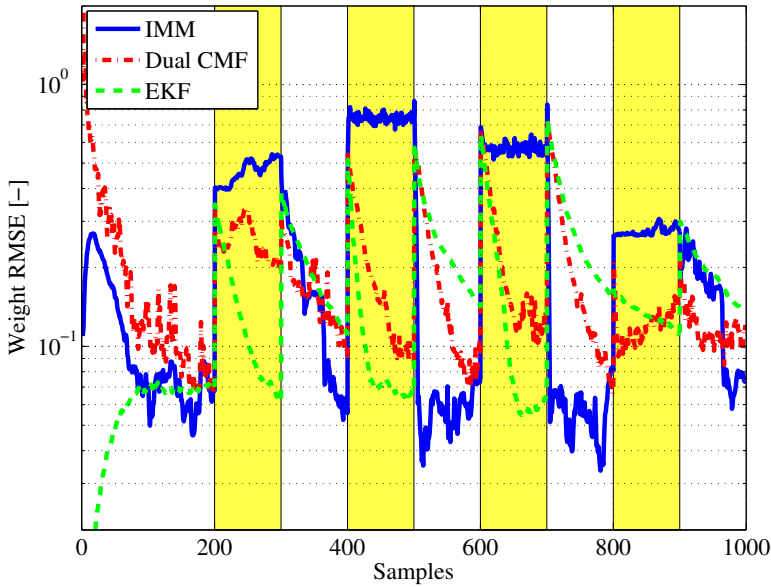


Figure 2.3: RMSE of the estimated weights during the FDI scenario.

### 2.4.2 Tracking of a Maneuvering Target

In this section the tracking problem of a maneuvering target in the presence of noisy measurements is considered. The goal of target tracking is to obtain a consistent estimate of the state even in case of noisy measurements and maneuvers of the target. The target tracking problem considered in this section is largely based on a target tracking problem considered by several other authors (Bar-Shalom and Li 1995; Doucet et al. 2001; Doucet and Andrieu 2001). It can therefore be considered to be a representative target tracking problem. The state of the target is given by  $x = [p_x \ v_x \ p_y \ v_y]^T$ , where  $p_x$  [m] and  $p_y$  [m] represent the positions in the  $x$  and  $y$  directions of the target, and  $v_x$  [m/s] and  $v_y$  [m/s] represent the velocities in the  $x$  and  $y$  directions, respectively. The maneuvering target evolves according to a JMLS, with parameters

$$A = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad C = I_4. \quad (2.65)$$

The switching term in this model is the  $B$ -matrix. This matrix can have three possible values that correspond to three different maneuver commands: straight

flight, left turn, and right turn. These three matrices are given by

$$\begin{aligned} B^{(1)} &= [0 \ 0 \ 0 \ 0]^T, \\ B^{(2)} &= \sqrt{2}[-0.25 \ -0.5 \ 0.25 \ 0.5]^T, \\ B^{(3)} &= \sqrt{2}[0.25 \ 0.5 \ -0.25 \ -0.5]^T. \end{aligned} \quad (2.66)$$

The fact that the  $B$ -matrix is the only system matrix that is time-varying has some consequences for the two filters based on the CM-structure. The consequence for the dual CMF is that the assumption that is required for the first step (i.e.  $\mu_k = \hat{\mu}_{k-1}$ ) is not required anymore. This can be seen by observing that the first three equations of the dual CMF described in (2.29)-(2.31) do not contain any instances of the  $B$ -matrix. The matrix  $B$  appears for the first time in the fourth equation (2.32). This equation can now be postponed until the vector  $\mu_k$  is evaluated in the second step of the dual CMF. This means that (2.32) can be evaluated after (2.36). The consequence for the EKF is that it simplifies to an ordinary Kalman filter with an augmented state consisting of the state  $x_k$  and the weight vector  $\mu_k$ . The reason for this is that there is no product between  $x_k$  and  $\mu_k$  in the system equations since the  $B$ -matrix is the only system matrix that is time-varying.

For the simulation experiment, process and measurement noise are used with covariances

$$Q = 10^{-3}I_4, \quad R = \begin{bmatrix} 2.5 \cdot 10^4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2.5 \cdot 10^4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.67)$$

This particular choice results in a SNR of approximately 20 dB for all four measurements. The system is simulated for 400 samples using  $T_s = 1$  s. In this simulation a left turn is simulated in the sample interval  $[100, 150]$  and a right turn is simulated in the sample interval  $[250, 300]$ . The left turn is simulated with half the acceleration that corresponds to  $B^{(2)}$ . This means that for samples  $k \in [100, 150]$ ,  $B_k = 0.5B^{(2)}$  is simulated, where  $B_k$  denotes the  $B$ -matrix used at sample  $k$ . The right turn is simulated with the same acceleration as  $B^{(3)}$ . The initial state of the target is chosen as  $[-500 \ 0 \ -500 \ 5]^T$ .

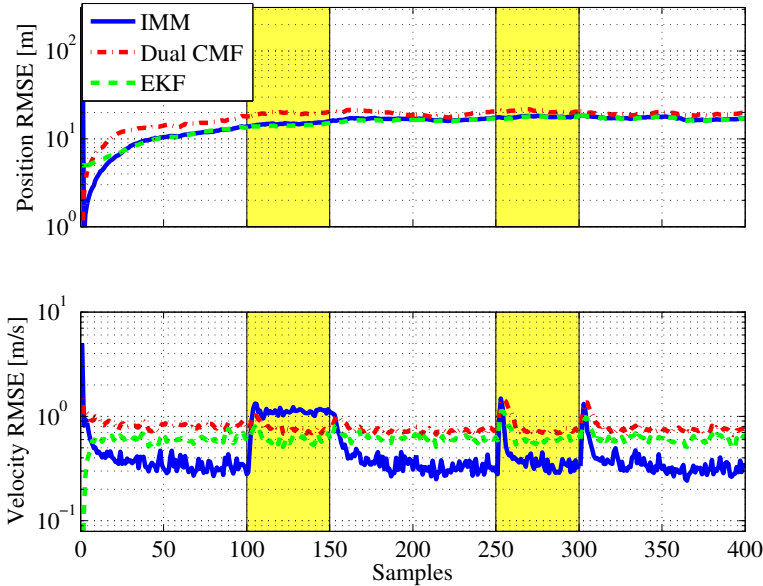
As has been discussed before, one of the benefits of the CM structure is its interpolation property. To illustrate this, the model set used for the two filters based on the CM structure is composed of only two models. These models are the two turn models. This can be done because the possible convex combinations of these two models also include the model for straight flight ( $B^{(1)} = 0.5B^{(2)} + 0.5B^{(3)}$ ). The model set used for the IMM filter contains all three models. This choice has been made to allow an honest comparison with the other two filters. The transition probability of the IMM filter in this experiment is chosen as

$$\Pi = \begin{bmatrix} 0.98 & 0.01 & 0.01 \\ 0.01 & 0.98 & 0.01 \\ 0.01 & 0.01 & 0.98 \end{bmatrix}. \quad (2.68)$$



The covariance matrices are chosen as  $Q_k^{\text{IMM}} = Q$  and  $R_k^{\text{IMM}} = R$  for all  $k$  and for all three models. The covariance matrices for the dual CMF are chosen as  $Q_k^{\text{DCMF}} = Q$ ,  $R_k^{\text{DCMF}} = R$  for all  $k$ , and the forgetting factor is chosen as  $\lambda = 0.9$ . The covariance matrices for the EKF are chosen as  $Q_k^{\text{EKF}} = Q$ ,  $R_k^{\text{EKF}} = R$ , and  $Q_k^\mu = 10^{-2}I_2$  for all  $k$ .

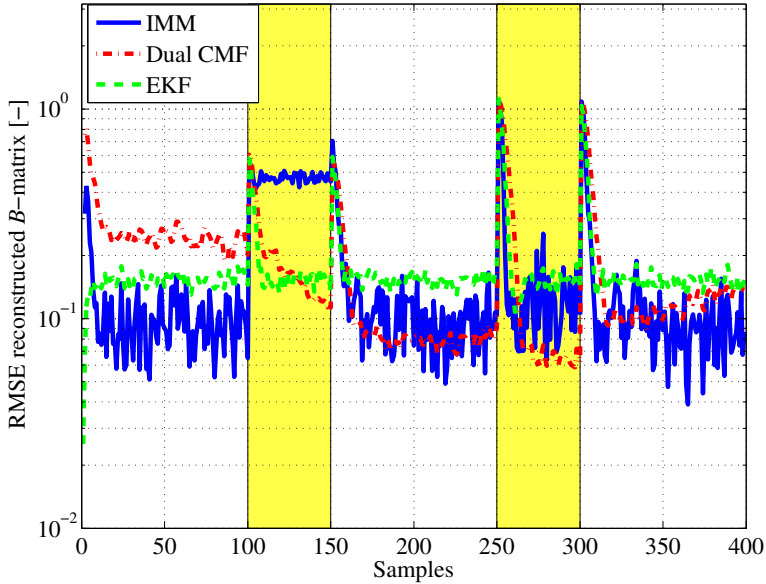
The state estimation results obtained with 100 Monte-Carlo simulations of the three filters are given in Figure 2.4. In this figure the RMSE values of the position states and velocity states have been computed separately to obtain physically interpretable RMSE values. The shaded intervals in Figure 2.4 correspond to the intervals in which a maneuver takes place. It can be observed that the RMSE values of the position is approximately the same for all three filters. However, the RMSE values of the velocity show significant differences. In general, the IMM filter has the lowest RMSE value, but in the sample interval  $[100, 150]$  it clearly does not. The reason for this is that in this interval a maneuver is performed for which the explicit model is not present in the model set used by the IMM filter. However, the same also holds for the other two filters. In the sample interval  $[100, 150]$ , the IMM filter tends to switch rapidly between two models, which clearly has a negative influence on the state estimation performance. The two CM estimation algorithms on the other hand manage to interpolate between the two turn models, resulting in better state estimation performance.



**Figure 2.4:** State estimation results for the target tracking example.

Since the three filters do not all use the same model set, the weight estimation performance cannot be evaluated by analyzing the weights directly. There-

fore, the weight estimation performance is evaluated by reconstructing the  $B$ -matrix using the estimated weights. For the two filters using the CM structure, the  $B$ -matrix is reconstructed by  $\hat{B}_k = \hat{\mu}_k^{(1)} B^{(2)} + \hat{\mu}_k^{(2)} B^{(3)}$ . For the IMM filter, which uses a model set containing three models, the  $B$ -matrix is reconstructed by  $\hat{B}_k = \hat{\mu}_k^{(1)} B^{(1)} + \hat{\mu}_k^{(2)} B^{(2)} + \hat{\mu}_k^{(3)} B^{(3)}$ . The RMSE of the reconstructed  $B$ -matrix for the three filters is computed using (2.64) and depicted in Figure 2.5. Note that this RMSE is dimensionless since it is based on physical quantities with different units. Note also that the 4 entries of the  $B$ -matrix are in the same order of magnitude, which justifies the use of the RMSE. In Figure 2.5, the same important observation can be made as in Figure 2.4. This observation is that in the sample interval  $[100, 150]$  the IMM filter is clearly outperformed by the other two filters. In general, the performance of the three filters is in the same order of magnitude.



**Figure 2.5:**  $B$ -matrix reconstruction results for the target tracking example.

An important conclusion that can be drawn from this simulation experiment is that the CM structure is to be preferred when interpolation between models from the model set is required. Furthermore, it can be concluded that for the considered target tracking example, the CM structure can indeed allow for smaller model sets, while at the same time maintaining the same estimation performance as an IMM filter with a larger model set. In fact, for situations in which model interpolation is required the estimation performance can be even better with smaller model sets when using the CM structure.

## 2.5 Conclusions

In this chapter an alternative model structure is proposed for the hybrid model structure of jump Markov linear systems. This structure, which is given the name convex model (CM) structure, explicitly allows for interpolation between models. The benefit of the estimation algorithms based on this structure is that they have better model interpolation properties. Having better model interpolation properties allows for smaller model sets. Furthermore, the CM structure does not require knowledge of the transition probabilities, which greatly simplifies the tuning process. Two estimation algorithms based on the proposed CM structure are explained and compared to the well-known IMM filter. This comparison has been performed by means of Monte-Carlo simulations of an FDI problem and a target tracking problem. In this comparison, conditions have been simulated that require interpolation of the local models in the model sets used by the three estimation algorithms. It has been concluded that the two filters based on the CM structure clearly outperform the IMM filter for these conditions.



**B**esides the choice of a model structure with an accompanying estimation algorithm, model set design is an important issue when using the multiple-model framework. In this chapter three structured model set design methods are developed. The first method uses orthogonal decompositions, the second method uses an algorithm that creates convex polytopes, and the third method efficiently represents the model parameters by only considering the limit values of these parameters. The key assumption for the first and second method is that a large number of local models can be generated that correspond to the different conditions to be modeled. For the third method it is assumed that physical model equations that describe the parameter variations are available.

---

## 3.1 Introduction

An important aspect of the application of multiple-model (MM) estimation methods is the design of the model set. Performance of the MM estimation methods is directly related to how well the system conditions to be modeled are represented by the models in the model set. Although model set design is such an important issue, it has been given little attention. This is in contrast to the many different MM estimation algorithms and applications of these algorithms that have been addressed in the literature (Li and Jilkov 2005). A straightforward method to design a model set is by gridding the set of varying model parameters and to include the resulting gridded models in the model set (Fisher and Maybeck 2002; Ru and Li 2008). However, this approach has two drawbacks. In case a system must be modeled for many different conditions, the model set designed with this approach can become very large resulting in a computationally demanding MM algorithm. Furthermore, the models in this large model set could be close to each other in terms of input-output behavior, which could lead to performance deterioration of the MM estimation algorithm (Zhang and Li 1998). Therefore, the need for a more structured model set design method that overcomes these drawbacks is apparent.

One of the few research efforts on the topic of structured model set design is the work by Li et al. (2005). In this work, three classes of model set design methods are presented, all of which are based on the probability density function (PDF) of the system to be approximated by an MM system. The PDF is a function of the varying model parameters. A drawback of this method is that it is not always an easy task to derive such a PDF. Therefore, this method is difficult to apply to systems for which the PDF is not readily available as is also remarked by Verhaegen et al. (2006). For example, for many components (sensor/actuators) used in control systems, information on the probability of occurrence of faults is not present. In this case, the PDF of the fault parameters cannot be computed exactly and has to be approximated. Since the model set design methods themselves are approximating procedures, an approximation of an approximated PDF is then made. This can result in unacceptable performance of the MM system.

In this chapter three methods for model set generation are proposed that do not require a PDF of the varying parameters. Instead, the methods are based on the varying parameters themselves. All three methods have the objective to efficiently represent the parameter space of the varying parameters. This parameter space is related to the physical model equations, which can generally be determined more accurately than the PDF of the varying parameters. Two of the proposed methods rely on the availability of a large model set that contains samples of the system model taken at different operating conditions. This large model set is then reduced by using two different methods. The first method to be presented in this chapter reduces the large model set by using a rank revealing QR (RRQR) decomposition (Chan and Hansen 1992), which is an orthogonal decomposition (OD). The second method reduces the large model set by computing a convex polytope (CP) enclosing all models in the large model set (Kanev 2006). The reduced model set then consists of the vertices of the CP. The third model set design method is different from the first two in the fact that it does not initiate with a large model set. Instead, this method is based directly on the limit values of the varying model parameters. Furthermore, the model structure of the MM system of the third method is slightly different from the model structure of the first two methods, which both use the model structure proposed in Chapter 2.

This chapter is organized as follows. In Section 3.2 a general problem formulation is given. In Section 3.3 a model set generation method is proposed that reduces an initially created large model set to a smaller model set by using orthogonal decompositions. In Section 3.4 also model set reduction is performed, but now by using an algorithm that encloses all models in the large model set in a CP. In Section 3.5 a model set design method is proposed that constructs local models based on the limit values of the varying model parameters. In Section 3.6 the three proposed model set design methods are compared in simulation by using a linear parameter-varying (LPV) model of a multiple input, multiple output mass-spring-damper (MSD) system that includes both faults and other varying system parameters. Finally, the conclusions can be found in Section 3.7.

## 3.2 Problem Formulation

Consider the following LPV state-space system

$$\dot{x}(t) = A(\rho(t))x(t) + B(\rho(t))u(t) + w(t), \quad (3.1)$$

$$y(t) = C(\rho(t))x(t) + D(\rho(t))u(t) + v(t), \quad (3.2)$$

where  $x(t) \in \mathbb{R}^n$  is the state,  $u(t) \in \mathbb{R}^m$  is the input,  $y(t) \in \mathbb{R}^l$  is the output,  $w(t) \in \mathbb{R}^n$  is the process noise, and  $v(t) \in \mathbb{R}^l$  is the measurement noise. Both  $v(t)$  and  $w(t)$  are assumed to be mutually uncorrelated Gaussian zero-mean white noise sequences. The parameter  $\rho(t)$  can take values in the bounded set  $\mathcal{R}$ . Although the set  $\mathcal{R}$  is bounded, the parameter  $\rho(t)$  can have infinitely many values. The goal in this chapter is to develop three model set design methods that can generate compact model sets for the approximation of the model defined in (3.1)-(3.2) for all  $\rho(t) \in \mathcal{R}$ . Details concerning the MM system structure and how the approximation is done, is explained separately for each model set design method in the following three sections. All three model set design methods are formulated in a state-space setting. However, it must be noted that the main principles of these methods can also be applied to other model types such as input-output models.

## 3.3 Orthogonal Decomposition Based Method

The goal of the OD-based model set design method, is to approximate a large model set by approximating the space spanned by the parameters of the individual local models in this model set. The local models in the large model set are created by sampling the model (3.1)-(3.2) at different values of  $\rho(t) \in \mathcal{R}$ . A similar approximation approach is applied to complexity reduction of fuzzy models (Yen and Wang 1999; Setnes and Babuška 2001). For example, Yen and Wang (1999) determine the most important rules from an existing rule base by applying different orthogonal decompositions on the firing matrix based on a large number of inputs. The firing matrix contains the firing strengths per rule for each input data point. Another example of the use of OD-based methods for model approximation is the work by Bos et al. (2005). In this work, a nonlinear state-space model is approximated by computing a model set that contains the most important directions of the parameter space using a singular value decomposition (SVD). This parameter space was obtained by linearizing the nonlinear model for many operating conditions along a pre-defined trajectory. A property of this approach is that the local models do not necessarily represent physically interpretable conditions. It is the linear combinations of the local models that correspond to physically interpretable conditions.

### 3.3.1 Motivation for Using Orthogonal Decompositions

The MM system that is used to approximate (3.1)-(3.2) is given by

$$\dot{x}(t) = \sum_{i=1}^N \mu^{(i)}(t) [A^{(i)}x(t) + B^{(i)}u(t)] + w(t), \quad (3.3)$$

$$y(t) = \sum_{i=1}^N \mu^{(i)}(t) [C^{(i)}x(t) + D^{(i)}u(t)] + v(t), \quad (3.4)$$

where  $\mu^{(i)}(t)$  is the model weight corresponding to the  $i$ -th local model represented by the state space quadruple  $\{A^{(i)}, B^{(i)}, C^{(i)}, D^{(i)}\}$  and  $N$  is the number of local models in the model set. Note that the CM structure, which is used in Chapter 2, is a discretized version of the MM system described by (3.3)-(3.4). The objective of these algorithms is to estimate both the state and the model weights. How well this estimation problem can be solved depends on the observability of the system. A system is called observable if for any sequence of the state and inputs, the current state can be determined in finite time by using only the outputs. Observability of a system depends on the system matrices. So, observability is a valid criterion for a model set design method to make the estimation problem related to the resulting MM system as well defined as possible.

Since the combined state and model weights estimation problem is nonlinear due to the product of state and model weights, using the well-known linear observability matrix (Verhaegen and Verdult 2007) to determine observability of the MM system is not possible. Therefore, a notion for nonlinear observability should be used instead. Such a notion is strong local observability (Nijmeijer 1982). In the following a definition will be given that can be used for determining strong local observability of a system.

**Definition 3.1** (Verdult et al. 2004) Consider the nonlinear state-space system

$$x_{k+1} = f(x_k, u_k), \quad (3.5)$$

$$y_k = h(x_k), \quad (3.6)$$

where  $x_k \in \mathbb{R}^n$  is the state,  $u_k \in \mathbb{R}^m$  is the input,  $y_k \in \mathbb{R}^l$  is the output and where  $f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ , and  $h : \mathbb{R}^n \rightarrow \mathbb{R}^l$  are smooth functions. This nonlinear system is strongly locally observable at  $(x_k, u_k)$  if

$$\text{rank} \left( \frac{\partial h^n(x_k, \bar{u}_k^n)}{\partial x_k} \right) = n, \quad (3.7)$$

where

$$h^n(x_k, \bar{u}_k^n) = \begin{bmatrix} h(x_k, u_k) \\ h \circ f^1(x_k, \bar{u}_k^1) \\ \vdots \\ h \circ f^{n-1}(x_k, \bar{u}_k^{n-1}) \end{bmatrix}, \quad \bar{u}_k^n = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+n-1} \end{bmatrix}, \quad (3.8)$$



with  $h \circ f^i(x_k, \bar{u}_k^i) = h(f^i(x_k, \bar{u}_k^i), u_{k+i})$  and  $f^i(x_k, \bar{u}_k^i) = f_{u_{k+i-1}} \circ f_{u_{k+i-2}} \circ \dots \circ f_{u_{k+1}} \circ f_{u_k}(x_k)$ , where  $f_{u_{k+i}} \circ f_{u_k}(x_k) = f(f(x_k, u_k), u_{k+1})$ .

The working principle of orthogonal decompositions, when applied for model set reduction, is to develop a reduced set of local models that approximates the initial large model set. The reduced set of local models has the property that it consists of the most influential models of the large model set. Which models are most influential is determined by the condition number of the matrix whose columns form a basis for the parameter space of these models. The smaller the condition number of the described matrix, the more influential the models. The condition number of a matrix can be computed with the function  $\text{cond}(\cdot)$ , which is defined as follows

$$\text{cond}(\Omega) = \frac{\sigma_{\max}(\Omega)}{\sigma_{\min}(\Omega)}, \quad (3.9)$$

where  $\sigma_{\max}(\Omega)$  and  $\sigma_{\min}(\Omega)$  are the largest and smallest singular values of  $\Omega$ , respectively. Matrices with a small condition number are referred to as well-conditioned matrices. This section started with describing the relevance of observability of the MM system and subsequently addressed well-conditioning of the parameter space basis of the model sets designed with the OD-based method. These two criteria are not conflicting. In fact, they are closely related to each other, as is demonstrated in Example 3.1.

---

**Example 3.1** In this example an MM system consisting of 4 state-space models with the following structure is considered

$$x_{k+1} = \sum_{i=1}^4 \mu_k^{(i)} [A^{(i)} x_k + B^{(i)} u_k], \quad (3.10)$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k, \quad (3.11)$$

where  $A^{(i)} \in \mathbb{R}^{2 \times 2}$  and  $B^{(i)} \in \mathbb{R}^{2 \times 1}, \forall i \in \{1, 2, 3, 4\}$ . Matrices  $A^{(1)}, A^{(2)}, B^{(1)}$ , and  $B^{(2)}$  are randomly generated. It is ensured that the eigenvalues of  $A^{(1)}$  and  $A^{(2)}$  lie within the unit circle in the complex plane to ensure stable behavior of at least the individual local models. Matrices  $A^{(3)}, A^{(4)}, B^{(3)}$ , and  $B^{(4)}$  are created from  $A^{(1)}, A^{(2)}, B^{(1)}$ , and  $B^{(2)}$  in the following way

$$A^{(3)} = A^{(1)} \bullet \begin{bmatrix} 1.05 & 0.95 \\ 0.95 & 1.05 \end{bmatrix}, \quad B^{(3)} = B^{(1)} \bullet \begin{bmatrix} 1.05 \\ 0.95 \end{bmatrix}, \quad (3.12)$$

$$A^{(4)} = 0.4A^{(1)} + 0.6A^{(2)}, \quad B^{(4)} = 0.4B^{(1)} + 0.6B^{(2)}, \quad (3.13)$$

where  $\bullet$  denotes the entrywise product, which is also known as the Hadamard product. This particular model set is constructed such that it supports the goal of this example. This goal is to clearly demonstrate the relation between the condition number of the parameter space basis of the reduced model set and the degree of observability of the MM system. The parameter space of the designed model set, consisting of 4 models, is deliberately chosen such that it can be approximated well by only 2 models.

According to Definition 3.1, checking whether condition (3.7) holds, is a valid method for computing strong local observability. This method only indicates whether a system is strongly locally observable or not. In order to get an idea of the degree of observability, the condition number of the nonlinear observability matrix  $\frac{\partial h^n(x_k, \bar{u}_k^n)}{\partial x_k}$  is used instead.

Based on the MM structure defined in (3.10)-(3.11), different state realizations for different (randomly created) model sets are obtained through simulation. The simulated model weights are chosen as sinusoidal signals and are the same for each of the different realizations. The input signal is chosen to have a constant value of 1. For each of the randomly created model sets consisting of 4 models, the condition number of the parameter space basis of the 6 different choices of two models is determined as follows

$$c^{(i_p)} = \text{cond} \left( \left[ \text{vec} \left( [A^{(i_q)} \ B^{(i_q)}] \right) \ \text{vec} \left( [A^{(i_r)} \ B^{(i_r)}] \right) \right] \right),$$

where the indices  $i_p$ ,  $i_q$ , and  $i_r$  can take values  $[i_p, i_q, i_r] \in \left\{ [1, 1, 2], [2, 1, 3], [3, 2, 3], [4, 1, 4], [5, 2, 4], [6, 3, 4] \right\}$ , and the operator  $\text{vec}(\cdot)$  stacks a matrix into a vector in a column-wise fashion. Next, the rank of the nonlinear observability matrix is determined for the different state realizations and for the different choices of two models. Note that the nonlinear observability matrix is computed for an MM system with two local models. The results of the simulations are given in Table

**Table 3.1:** Mean of condition numbers of the observability matrix based on 6 different pairs of models. The mean of the condition numbers is computed based on 100 samples from the state realization. The results of this procedure are presented for 15 different model sets and their state realizations. The model pair with the smallest condition number is indicated in **bold** font for each model set.

| Model set index | Model pair: {1, 2} | Model pair: {1, 3} | Model pair: {2, 3} | Model pair: {1, 4} | Model pair: {2, 4} | Model pair: {3, 4} |
|-----------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| 1               | <b>37</b>          | $6.31 \cdot 10^3$  | 125                | 85.2               | 128                | 85.8               |
| 2               | 5.58               | 226                | <b>5.53</b>        | 8.75               | 12.9               | 8.49               |
| 3               | <b>10.5</b>        | $6.29 \cdot 10^3$  | 27.5               | 22.6               | 33.8               | 21.7               |
| 4               | 110                | $9.45 \cdot 10^3$  | <b>87.4</b>        | 178                | 264                | 126                |
| 5               | 291                | 440                | <b>128</b>         | 321                | 468                | 154                |
| 6               | 23.4               | 424                | <b>22.4</b>        | 37.8               | 56.3               | 35                 |
| 7               | <b>48.7</b>        | $1.41 \cdot 10^4$  | 61.7               | 77.3               | 116                | 89.5               |
| 8               | 278                | $1.17 \cdot 10^3$  | <b>178</b>         | 464                | 695                | 234                |
| 9               | 45.9               | 809                | <b>44.1</b>        | 72.2               | 108                | 75                 |
| 10              | 29.2               | 234                | <b>24.6</b>        | 48.3               | 72.2               | 35.9               |
| 11              | <b>30.1</b>        | $2.71 \cdot 10^3$  | 31                 | 50.5               | 75.6               | 48.4               |
| 12              | 24.2               | 477                | <b>23.4</b>        | 40.1               | 60.1               | 36.3               |
| 13              | <b>42.8</b>        | $4.75 \cdot 10^3$  | 105                | 640                | 666                | 353                |
| 14              | 367                | $4.25 \cdot 10^3$  | <b>73.2</b>        | 564                | 662                | 74.8               |
| 15              | <b>53.1</b>        | $3.01 \cdot 10^3$  | 64                 | 94                 | 141                | 90.7               |

3.1. In this table the mean of the condition number of the observability matrix is computed for the 6 different model pairs for 15 different model sets. It can be seen that either model pair {1, 2} or {2, 3} has the smallest mean condition num-

ber. This is according to what could be expected from the generated model sets since the parameter space bases of these two pairs have the smallest condition number. It can also be seen that the mean condition numbers for pair  $\{1, 3\}$  are the highest of all pairs since models 1 and 3 are almost similar (see (3.12)). Furthermore, it can be observed that pair  $\{2, 4\}$  has a higher mean condition number than pairs  $\{1, 4\}$  and  $\{3, 4\}$ . The reason for this is that model 4 consists for 60% of model 2, while it consists only for 40% of model 1 (see (3.13)). These different observations all indicate that the choices of two models with the highest condition number of the parameter space basis also have the highest condition number of the nonlinear observability matrix. This leads to the conclusion that model sets with well-conditioned parameter space bases are better suited for the combined state and model weight estimation problem.

### 3.3.2 Derivation of the OD-based Model Set Design Method

In this section first the problem of designing a model set with a parameter space basis that is as well-conditioned as possible is formally posed. Next, the solution for this problem using orthogonal decompositions is given. The local models of (3.3)-(3.4) can be represented in a compact form by using the vectorizing operator  $\text{vec}(\cdot)$  as follows

$$\mathfrak{M}^{(i)} = \text{vec} \left( \begin{bmatrix} A^{(i)} & B^{(i)} \\ C^{(i)} & D^{(i)} \end{bmatrix} \right), \quad (3.14)$$

where  $\mathfrak{M}^{(i)} \in \mathbb{R}^{n_q}$  and  $n_q$  is the number of entries that define the  $i$ -th model. It should be noted that it is assumed that all local models have the same structure and therefore all models have  $n_q$  entries, as is also clear from (3.3)-(3.4). The model representation from (3.14) allows the large model set to be represented as

$$\mathbf{M}_\ell = \begin{bmatrix} \mathfrak{M}^{(1)} & \mathfrak{M}^{(2)} & \dots & \mathfrak{M}^{(N_\ell)} \end{bmatrix}, \quad (3.15)$$

where  $N_\ell$  denotes the number of models in the large model set and in general it holds that  $n_q \ll N_\ell$ . The models in this large model set are obtained by taking a large number of samples from the system to be approximated, i.e. the system defined by (3.1)-(3.2), at different operating conditions.

Now that the notation has been explained, the model set design problem can be formulated. Let a large model set  $\mathbf{M}_\ell$  be given, find a column selection matrix  $\Pi \in \mathbb{R}^{N_\ell \times N_r}$  that selects  $N = N_r$  models from  $\mathbf{M}_\ell$  such that a reduced model set can be computed as  $\mathbf{M}_r = \mathbf{M}_\ell \Pi$ . Given the number of models to be selected ( $N_r$ ), this problem can be cast into the following optimization problem

$$\begin{aligned} \min_{\Pi \in \mathbb{R}^{N_\ell \times N_r}} \quad & \text{cond}(\mathbf{M}_\ell \Pi) \\ \text{s.t.} \quad & \Pi^T \Pi = I_{N_r} \\ & \Pi_{i,j} \in \{0, 1\}, \quad \forall i \in \{1, 2, \dots, N_\ell\}, j \in \{1, 2, \dots, N_r\}, \end{aligned} \quad (3.16)$$

where  $\Pi_{i,j}$  denotes the element of  $\Pi$  on the  $i$ -th row and  $j$ -th column. Note that the two constraints regarding  $\Pi$  ensure that it is indeed a column selection matrix.

A possible solution for the optimization problem defined in (3.16) is to use the rank revealing QR (RRQR) decomposition. An RRQR decomposition of a matrix  $A$  has the following form

$$A\Pi_{\text{RR}} = QR = Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}, \quad (3.17)$$

where  $\Pi_{\text{RR}}$  is a permutation matrix,  $Q$  has orthonormal columns and  $R_{11}$  is a  $k \times k$  matrix, with  $k$  the numerical rank of  $A$  with respect to a tolerance  $\epsilon$ , which is defined as (Golub and Van Loan 1996)

$$k = k(A, \epsilon) = \min_{\|A-B\| \leq \epsilon} \text{rank}(B). \quad (3.18)$$

In words this definition means that the numerical rank of  $A$  is equal to the number of columns in  $A$  that are linearly independent for any perturbation of  $A$  with norm less than or equal to the tolerance  $\epsilon$ . The RRQR decomposition is also referred to as pivoted QR, for example by Setnes and Babuška (2001), who have used it for fuzzy rule base reduction. The RRQR decomposition of a rectangular matrix can also be used as a reliable and efficient computational alternative to the SVD for problems that involve rank determination (Chan and Hansen 1992).

Before the RRQR decomposition can be applied to a model set that should be reduced, two issues should be dealt with. The first issue is concerned with how the dimension of  $R_{11}$  in (3.17) should be chosen. This dimension determines the number of models in the reduced model set and is related to the numerical rank of the matrix to be decomposed. The most obvious method to determine the numerical rank of a matrix is to compute an SVD of the matrix and to determine the position at which a significant gap occurs between subsequent singular values (Verhaegen and Verdult 2007). However, determining this rank can also be done, without computing an SVD, by analyzing the absolute values of the diagonal entries of matrix  $R$  from (3.17). These values are decreasing by definition of the RRQR decomposition and tend to track the singular values well enough to expose gaps (Setnes and Babuška 2001). So, the number of models to be chosen for the reduced model set can be determined by determining the position of a significant gap between subsequent absolute values of the diagonal of  $R$ . The second issue to be dealt with is concerned with the property that a linear combination of physically interpretable state-space models only delivers a physically interpretable state-space model if the model weights sum up to 1. For example, consider a number of local models that all have a measurement matrix  $C$  equal to the identity matrix (i.e. all states are measured). If the weights of the linear combination of these measurement matrices do not sum up to 1, then the linear combination can never be equal to the identity matrix. Therefore, this property is explicitly taken into account in the model selection procedure. This procedure is described in the following:

1. Given the large model set to be reduced  $\mathbf{M}_\ell$ , compute its RRQR decomposition

$$\mathbf{M}_\ell \Pi_{\mathbf{M}_\ell} = \mathbf{Q}_{\mathbf{M}_\ell} \mathbf{R}_{\mathbf{M}_\ell}, \quad (3.19)$$

and determine the position of the first significant gap in the absolute values of the diagonal of  $\mathbf{R}_{\mathbf{M}_\ell}$ . This position  $N_r$  determines the size of the reduced model set.

2. Compute the reduced model set as follows:

$$\mathbf{M}_r = [\mathbf{M}_\ell \Pi_{\mathbf{M}_\ell}]_{(:,1:N_r+1)}, \quad (3.20)$$

where the notation  $[\cdot]_{(:,1:N_r+1)}$  denotes the first  $N_r + 1$  columns of the concerning matrix. Note that the size of the reduced model set is increased with 1 to accommodate the fact that the weights should sum up to 1.

3. Compute the projection of the models in the large model set onto the space spanned by the reduced model set  $\mathbf{M}_r$ , with an additional constraint that the model weights of the models in the reduced model sum up to 1. This projection is performed as follows:

$$\widetilde{\mathbf{M}}_\ell = \begin{bmatrix} \mathbf{M}_r \\ \mathbf{1}_{N_r+1} \end{bmatrix}^\dagger \begin{bmatrix} \mathbf{M}_\ell \\ \mathbf{1}_{N_\ell} \end{bmatrix}, \quad (3.21)$$

where  $\mathbf{1}_N$  denotes an  $N$ -dimensional row vector consisting of only ones and  $[\cdot]^\dagger$  denotes the pseudo-inverse of the corresponding matrix.

4. Compute the RRQR decomposition of  $\widetilde{\mathbf{M}}_\ell$

$$\widetilde{\mathbf{M}}_\ell \Pi_{\widetilde{\mathbf{M}}_\ell} = \mathbf{Q}_{\widetilde{\mathbf{M}}_\ell} \mathbf{R}_{\widetilde{\mathbf{M}}_\ell}, \quad (3.22)$$

and finally compute the reduced model set using the initial large model set  $\mathbf{M}_\ell$ :

$$\widetilde{\mathbf{M}}_r = [\mathbf{M}_\ell \Pi_{\widetilde{\mathbf{M}}_\ell}]_{(:,1:N_r+1)} = [\widetilde{\mathfrak{M}}^{(1)} \quad \widetilde{\mathfrak{M}}^{(2)} \quad \dots \quad \widetilde{\mathfrak{M}}^{(N_r+1)}]. \quad (3.23)$$

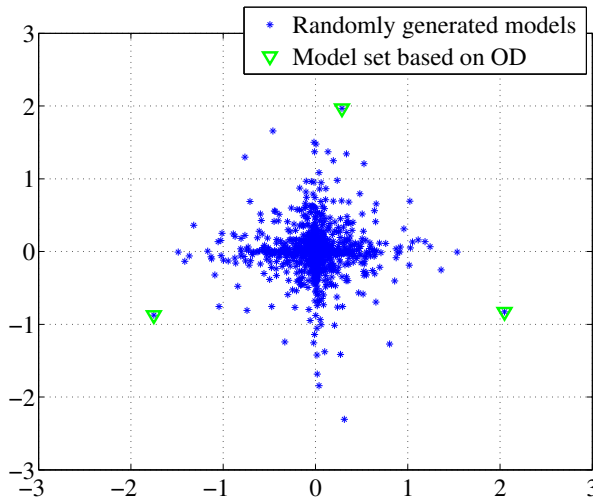
The state-space matrices for the models  $\widetilde{\mathfrak{M}}^{(i)}, \forall i \in \{1, 2, \dots, N_r + 1\}$  can be reconstructed as

$$\begin{bmatrix} A^{(i)} & B^{(i)} \\ C^{(i)} & D^{(i)} \end{bmatrix} = \text{vec}^{-1} \left( \widetilde{\mathfrak{M}}^{(i)} \right). \quad (3.24)$$

For illustration purposes, the proposed model set design method is applied to a set of 1000 randomly generated local models with a 2-dimensional parameter space. This means that for all local models it holds that  $\mathfrak{M}^{(i)} \in \mathbb{R}^2$ . The result of this method is shown in Figure 3.1. In this figure it can be seen that 3 models are chosen from the models in the 2-dimensional parameter space (which means  $N_r = 2$  in this case). A choice of only two models would have lead to a linear combination of the two models that would have been represented by a line because

of the constraint that the model weights should sum up to 1. In this way not all models could be approximated well. With the choice of 3 models each randomly generated model can be exactly represented even with the constraint. Note also that the 3 chosen models are well separated from each other as a consequence of the use of the RRQR decomposition.

The model sets designed with the OD-based method have the property that the local models correspond to physically interpretable conditions. An important advantage of this property is that the weights corresponding to the local models are more easily interpretable. Consider, for example, the case that a model set is designed for a fault diagnosis (FD) problem that contains only one model that corresponds to the nominal condition. In this case, the weight of the local model corresponding to the nominal condition has a value of 1 when there are no faults. So, when this model weight deviates from a value of 1, it is immediately clear that the system has deviated from its nominal condition.



**Figure 3.1:** Illustration of the OD-based model set design method applied to a set of 1000 randomly generated models with a 2-dimensional parameter space.

### 3.4 Convex Polytope Based Method

The starting point for the model set design method presented in this section is the same initial large model set  $M_\ell$  defined in (3.15). Contrary to the OD-based method, the method presented here does not select models from the large model set. Instead, a convex polytope (CP) is computed that encloses all models in this set (Kanev 2006). The reduced model set is then constructed from the vertices of the CP. The MM structure used for this method is the same as the one defined in

(3.3)-(3.4), with the addition of the constraints  $\mu^{(i)}(t) \geq 0, \forall i \in \{1, 2, \dots, N\}$  and  $\sum_{i=1}^N \mu^{(i)}(t) = 1$ , the combination of which ensures convexity of the model set.

Let the models from  $\mathbf{M}_\ell$  define a set

$$\mathbf{M}_\ell^p = \left\{ \mathfrak{M}^{(1)}, \mathfrak{M}^{(2)}, \dots, \mathfrak{M}^{(N_\ell)} \right\}, \quad (3.25)$$

and let the convex hull of  $\mathbf{M}_\ell^p$  be denoted as

$$\mathcal{M} = \text{conv}(\mathbf{M}_\ell^p), \quad (3.26)$$

where  $\text{conv}(\cdot)$  is the operator that computes a convex hull of a set. The goal now is to construct a CP, denoted by  $\mathcal{V}$ , with a desired number of vertices  $N_h$  that encloses the polytope  $\mathcal{M}$ , i.e.  $\mathcal{M} \subseteq \mathcal{V}$ . It would be undesirable if the volume of the polytope  $\mathcal{V}$ , denoted by  $\text{vol}(\mathcal{V})$ , would be much larger than  $\text{vol}(\mathcal{M})$ . The reason for this is that many irrelevant system conditions would also unnecessarily be represented by  $\mathcal{V}$ . Therefore an objective function should be introduced that prevents this.

The most straightforward objective function would be one that contains the difference between  $\text{vol}(\mathcal{M})$  and  $\text{vol}(\mathcal{V})$ . However, computing the volume of a polytope is a computationally burdensome task. Therefore, Kanev (2006) has proposed an alternative objective function. This objective function is based on the projection of a point  $z$  onto the polytope  $\mathcal{M}$ . This projection is defined as a point on  $\mathcal{M}$  that is closest to  $z$

$$\phi(z, \mathcal{M}) = \arg \min_{\tilde{\phi} \in \mathcal{M}} \|z - \tilde{\phi}\|_2. \quad (3.27)$$

A graphical representation of this projection is depicted in Figure 3.2 (left). The distance from point  $z$  to its projection on  $\mathcal{M}$  is defined as

$$d(z, \mathcal{M}) = \|z - \phi(z, \mathcal{M})\|_2. \quad (3.28)$$

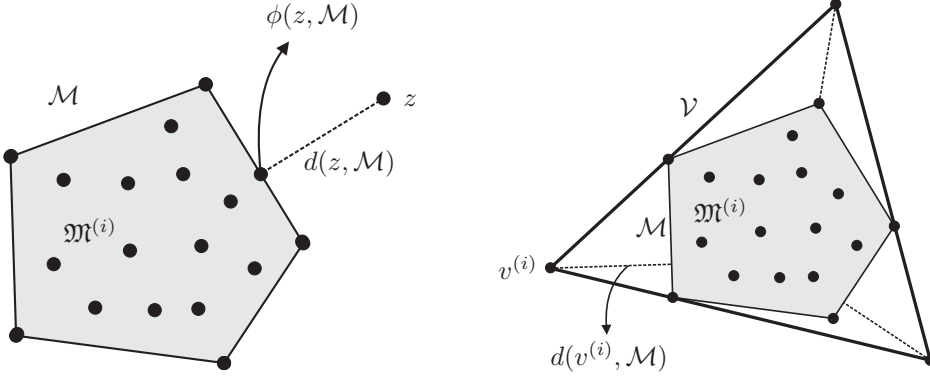
Using this distance measure, the following objective function can be formulated

$$J_{\text{CP}}(\mathbf{M}_\ell^p, \mathcal{V}) = \sum_{v \in \text{vert}(\mathcal{V})} d^2(v, \text{conv}(\mathbf{M}_\ell^p)). \quad (3.29)$$

The vertices of a polytope  $\mathcal{V}$  are denoted as  $\text{vert}(\mathcal{V}) = \{v^{(1)}, v^{(2)}, \dots, v^{(N_h)}\}$ . The objective function  $J_{\text{CP}}(\mathbf{M}_\ell^p, \mathcal{V})$  is a measure for how well a polytope  $\mathcal{V}$  approximates the set  $\mathbf{M}_\ell^p$  and it is a viable alternative to an objective function that uses volumes of polytopes. In Figure 3.2 (right) a graphical representation is given of such an approximating polytope  $\mathcal{V}$ . The distances of the vertices of  $\mathcal{V}$  to  $\mathcal{M}$  are represented by dotted lines. The optimization problem to be solved to obtain the vertices of  $\mathcal{V}$  can now be formulated as

$$\begin{aligned} \begin{bmatrix} v^{(1)} & \dots & v^{(N_h)} \end{bmatrix} &= \arg \min_{\{\tilde{v}^{(1)}, \dots, \tilde{v}^{(N_h)}\}} J_{\text{CP}}(\mathbf{M}_\ell^p, \mathcal{V}) \\ \text{s.t.} \quad &\mathcal{M} \subseteq \mathcal{V} = \text{conv} \left( \left\{ \tilde{v}^{(1)}, \dots, \tilde{v}^{(N_h)} \right\} \right). \end{aligned} \quad (3.30)$$

A big advantage of choosing  $J_{\text{CP}}(\mathbf{M}_\ell^p, \mathcal{V})$  as an objective function is that optimization problem (3.30) can be solved in a relatively efficient way. It has been shown by Kanev (2006) how (3.30) can be cast as a non-convex optimization problem with quadratic equality and inequality constraints.



**Figure 3.2:** The projection of a point  $z$  onto the 2-dimensional polytope  $\mathcal{M}$  is denoted by the point  $\phi(z, \mathcal{M})$ . The distance between  $z$  and its projection onto  $\mathcal{M}$  is denoted by  $d(z, \mathcal{M})$  (left). The 2-dimensional polytope  $\mathcal{V}$  encloses polytope  $\mathcal{M}$ . The objective function of the algorithm that computes the vertices  $v^{(i)}$  of  $\mathcal{V}$  is based on the distances  $d(v^{(i)}, \mathcal{M})$  (right).

The models from  $\mathbf{M}_\ell$  all have  $n_q$  entries. However, not all these  $n_q$  entries are depending on the varying model parameters. Furthermore, some of the entries that do vary can vary in a similar manner to other entries. It is therefore not required to consider all  $n_q$  entries in the process of finding an approximating CP. This allows decreasing the computational complexity of the optimization problem. The reduction of entries to be considered can be achieved by using an SVD. In the following the complete CP-based model set design method, including the reduction step, is described.

1. Given the large model set to be reduced  $\mathbf{M}_\ell$ , compute its SVD

$$\mathbf{M}_\ell = [\mathbf{U}_{\mathbf{M}_\ell}^1 \quad \mathbf{U}_{\mathbf{M}_\ell}^2] \begin{bmatrix} \Sigma_{\mathbf{M}_\ell}^1 & 0 \\ 0 & \Sigma_{\mathbf{M}_\ell}^2 \end{bmatrix} \begin{bmatrix} (\mathbf{V}_{\mathbf{M}_\ell}^1)^T \\ (\mathbf{V}_{\mathbf{M}_\ell}^2)^T \end{bmatrix}, \quad (3.31)$$

where  $\Sigma_{\mathbf{M}_\ell}^1 \in \mathbb{R}^{n_g \times n_g}$  and  $n_g$  is the position at which a gap occurs in the singular value plot ( $n_g$  is equivalent to the numerical rank defined in (3.18)). Next, compute

$$\overline{\mathbf{M}}_\ell = \Sigma_{\mathbf{M}_\ell}^1 (\mathbf{V}_{\mathbf{M}_\ell}^1)^T. \quad (3.32)$$

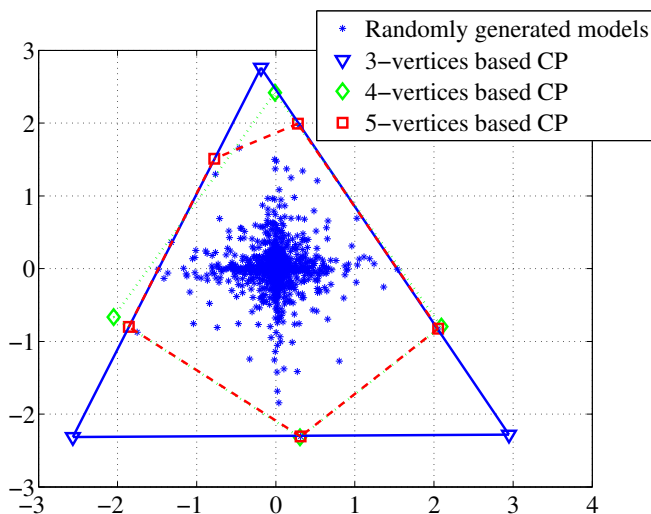
2. Let  $\overline{\mathbf{M}}_\ell^p$  denote a set consisting of the models from  $\overline{\mathbf{M}}_\ell$ . Now, the optimization problem (3.30) can be solved by using the cost function  $J_{\text{CP}}(\overline{\mathbf{M}}_\ell^p, \mathcal{V})$  and  $N_h \geq n_g + 1$ .



3. With the result obtained from the optimization problem, i.e.  $[v^{(1)} \dots v^{(N_h)}]$ , the state-space matrices of the reduced model set can be reconstructed  $\forall i \in \{1, 2, \dots, N_h\}$  as

$$\begin{bmatrix} A^{(i)} & B^{(i)} \\ C^{(i)} & D^{(i)} \end{bmatrix} = \text{vec}^{-1} \left( U_{\mathbf{M}_\ell}^1 v^{(i)} \right). \quad (3.33)$$

In Figure 3.3 a graphical illustration is given of the CP-based model set design method. In this figure the same set of 1000 randomly generated models is used as in Figure 3.1. Results are shown of convex polytopes with 3, 4, and 5 vertices. It can be seen that the higher the number of vertices, the smaller the volume of the CP. A small CP volume is beneficial in the sense that less irrelevant space is enclosed in the polytope. However, a larger number of vertices also means a larger model set, which results in increased computational load. Therefore, a trade-off has to be made between these two properties.



**Figure 3.3:** Illustration of the CP-based model set design method applied to a set of 1000 randomly generated models with a 2-dimensional parameter space. Results for polytopes with 3, 4, and 5 vertices are shown.

## 3.5 Limit Values Based Method

Unlike the two previously presented model set design methods, the starting point of the method presented in this section is not a randomly generated large model set. Instead, this method is based on limit values (LV) of the varying entries of the system matrices. Consider the LPV system defined in (3.1)-(3.2). Let the entry on

the  $i$ -th row and  $j$ -th column of  $A(\rho(t))$  be denoted by  $a_{ij}(t)$ . As a result of the variation of  $\rho(t)$ ,  $a_{ij}(t)$  varies in the interval  $a_{ij}(t) \in [a_{ij}^-, a_{ij}^+]$ . This entry can be modeled as follows

$$a_{ij}(t) = \mu_{ij}^a(t)a_{ij}^- + (1 - \mu_{ij}^a(t))a_{ij}^+ \quad (3.34)$$

$$= a_{ij}^+ + \mu_{ij}^a(t)(a_{ij}^- - a_{ij}^+) \quad (3.35)$$

$$= a_{ij}^+ + \mu_{ij}^a(t)a_{ij}^\Delta, \quad (3.36)$$

where  $\mu_{ij}^a(t) \in [0, 1]$ . Let  $A^+$  be defined as the matrix in which all entries are at their maximal values and let  $A_{ij}^\Delta$  be defined as the matrix in which all entries are 0, except for  $a_{ij} = a_{ij}^\Delta$ , i.e.

$$A^+ = \{A : a_{ij} = a_{ij}^+ \forall i, j \in \{1, 2, \dots, n\}\}, \quad (3.37)$$

$$A_{ij}^\Delta = \{A : a_{ij} = a_{ij}^\Delta \wedge a_{i\tilde{j}} = 0 \forall i \neq i, \tilde{j} \neq j\}. \quad (3.38)$$

If a similar notation is also used for the  $B$ ,  $C$ , and  $D$  matrix, then the following MM system can be formulated

$$\dot{x}(t) = \left( A^+ + \sum_{ij} A_{ij}^\Delta \mu_{ij}^a(t) \right) x(t) + \left( B^+ + \sum_{ij} B_{ij}^\Delta \mu_{ij}^b(t) \right) u(t) + w(t), \quad (3.39)$$

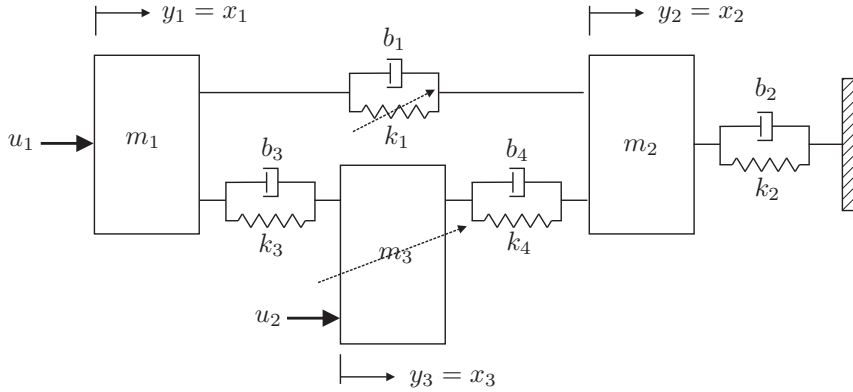
$$y(t) = \left( C^+ + \sum_{ij} C_{ij}^\Delta \mu_{ij}^c(t) \right) x(t) + \left( D^+ + \sum_{ij} D_{ij}^\Delta \mu_{ij}^d(t) \right) u(t) + v(t), \quad (3.40)$$

with the additional constraint that all weights, i.e.  $\mu_{ij}^a(t)$ ,  $\mu_{ij}^b(t)$ ,  $\mu_{ij}^c(t)$ , and  $\mu_{ij}^d(t)$  should lie in the interval  $[0, 1]$  for all possible combinations of indices  $i$  and  $j$ . Note that the MM system defined in (3.39)-(3.40) differs from the MM system defined in (3.3)-(3.4) in two points. The first point is that the system matrices all have offset terms ( $A^+$ ,  $B^+$ ,  $C^+$ , and  $D^+$ ) that are not weighted. The second point is that the weights now only weigh a single matrix instead of a full state-space quadruple  $\{A^{(i)}, B^{(i)}, C^{(i)}, D^{(i)}\}$  as is the case in (3.3)-(3.4). Because of these differences, the MM estimation algorithm to be used for the MM system defined in (3.39)-(3.40) also differs from the algorithm used for the MM system defined in (3.3)-(3.4). However, this difference is small. The only required modification is to restructure the weights in the MM estimation algorithm for the system defined in (3.3)-(3.4) such that it fits the system defined in (3.39)-(3.40).

## 3.6 Simulation Example

In this section, the three proposed model set design methods are evaluated. Furthermore a comparison is made with a model set design method developed by Li et al. (2005). For these purposes, an LPV model of an MSD system is used. This type of system is generally considered to be a realistic testbed for the study of

control and estimation algorithms. The particular MSD system that is used here is adopted from the work by Fekri et al. (2006b), in which it is used in an MM adaptive control setting. This MSD system consists of 3 masses, 4 springs, and 4 dampers that are linked together. In Figure 3.4 a schematic overview is given of this system.



**Figure 3.4:** Overview of the MSD system.

The dynamics of the continuous-time MSD system are described by a state-space system, with a state  $x_{\text{MSD}}$  and system matrices  $A_{\text{MSD}}$ ,  $B_{\text{MSD}}$ , and  $C_{\text{MSD}}$ , which are defined as

$$A_{\text{MSD}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -\frac{k_1+k_3}{m_1} & \frac{k_1}{m_1} & \frac{k_3}{m_1} & -\frac{b_1+b_3}{m_1} & \frac{b_1}{m_1} & \frac{b_3}{m_1} \\ \frac{k_1}{m_2} & -\frac{k_1+k_2+k_4}{m_2} & \frac{k_4}{m_2} & \frac{b_1}{m_2} & -\frac{b_1+b_2+b_4}{m_2} & \frac{b_4}{m_2} \\ \frac{k_3}{m_3} & \frac{k_4}{m_3} & -\frac{k_3+k_4}{m_3} & \frac{b_3}{m_3} & \frac{b_4}{m_3} & -\frac{b_3+b_4}{m_3} \end{bmatrix}, \quad (3.41)$$

$$x_{\text{MSD}} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ v_1 \\ v_2 \\ v_3 \end{bmatrix}, \quad B_{\text{MSD}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ f_1 \frac{1}{m_1} & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_3} \end{bmatrix}, \quad C_{\text{MSD}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & f_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad (3.42)$$

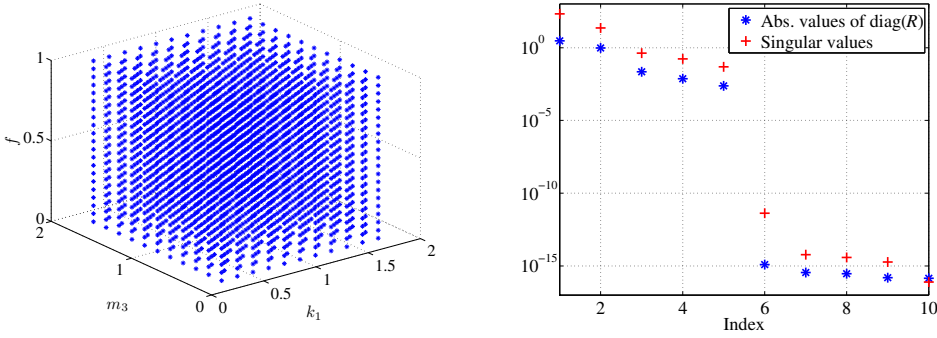
where  $k_i$  are the spring constants,  $b_i$  are the damping coefficients,  $x_i$  and  $v_i$  are the positions and velocities of masses  $m_i$ , and  $u_i$  are the applied control forces. Besides the nominal system parameters, two fault parameters are introduced. Parameter  $f_1 \in [0, 1]$  implements an actuator fault in the first actuator. In case  $f_1 = 1$ , the actuator is fully functional and in case  $f_1 < 1$  the actuator operates with degraded effectiveness. Parameter  $f_2 \in [0, 1]$  implements a sensor fault in the second sensor and can be interpreted similarly as  $f_1$ . In this simulation example only single faults are considered. So, only one of the two fault parameters can have a value smaller than 1 at the same time. Note that the system has no  $D$ -matrix since

it does not have a direct feedthrough term. The nominal system parameters in these system matrices are all known, except for  $k_1$  and  $m_3$ . The values for the known parameters are

$$\begin{aligned} m_1 &= m_2 = 1, \\ b_1 &= b_2 = b_3 = b_4 = 0.01, \\ k_2 &= 0.15, \quad k_3 = 1, \quad k_4 = 0.2, \end{aligned} \tag{3.43}$$

and the upper and lower bounds for the unknown parameters are given by

$$\begin{aligned} k_1 &\in [k_1^{\min} \quad k_1^{\max}] = [0.25 \quad 1.75], \\ m_3 &\in [m_3^{\min} \quad m_3^{\max}] = [0.20 \quad 1.80]. \end{aligned}$$



**Figure 3.5:** Overview of points in the parameter space of  $k_1$  and  $m_3$  at which model samples are taken for the large model set (left). Singular values of the large model set and absolute values of the diagonal of  $R$  (right).

### 3.6.1 Model Set Design For a Mass-Spring-Damper System

The OD-based and CP-based model set design methods both require a large model set that is obtained by sampling of the parameter space of the LPV model. For these two methods a model set is generated by performing a uniform sampling in the parameter space. The points at which the model is sampled are depicted in Figure 3.5 (left). Since only single faults are considered, the sampling pattern depicted in this figure has been implemented separately for both  $f_1$  and  $f_2$  (while at the same time the value of the non-sampled fault parameter is held constant at a value of 1).

The dynamics of the MSD system is described in continuous-time. Since the model weight estimation algorithms operate in discrete-time, the continuous-time model set has to be discretized. In Appendix B it is shown that the local models

can be individually discretized if a first-order Taylor approximation is made of the discretized system matrices. A requirement for this type of discretization is that the sample time must be chosen relatively small such that the truncation error is also small. Therefore, the sample time used in this example is chosen as  $T_s = 2.5 \cdot 10^{-3}$  s. The first-order discretization is performed before the models are assembled into the matrix  $\mathbf{M}_\ell$  defined in (3.15).

Once matrix  $\mathbf{M}_\ell$  is constructed using the previously described sampled local models, the OD-based and CP-based model set design methods can be used to obtain a reduced model set. The number of local models to be used in the model sets can be determined by either locating a gap in the absolute values of the diagonal of the  $R$  matrix (of the RRQR decomposition) or by locating a gap in the singular value plot. Which of these two procedures should be followed depends on which of the two model set design methods is used. In Figure 3.5 (right) the values obtained with the two procedures are depicted. In this figure it can be clearly seen that both procedures lead to the conclusion that the numerical rank of the model set is equal to 5. Moreover, it can be seen that the previously made statement that the absolute values of the diagonal of the  $R$  matrix tend to track the singular values well enough to expose gaps, holds true. According to the indications given in the model set design procedures for the OD-based method and the CP-based method, 6 local models are chosen for both methods. The model set obtained with the OD-based method contains a selection from the models in  $\mathbf{M}_\ell$ . The parameters of the 6 selected models are given in Table 3.2. In this table it can be seen that the selected models are well separated from each other. Two models correspond to the nominal situation (no faults). Two models correspond to a maximal fault in the first actuator ( $f_1 = 0$ ) and two models correspond to a maximal fault in the second sensor ( $f_2 = 0$ ).

**Table 3.2:** Parameters of the models selected with the OD-based model set design method.

|       | Model 1 | Model 2 | Model 3 | Model 4 | Model 5 | Model 6 |
|-------|---------|---------|---------|---------|---------|---------|
| $k_1$ | 1.75    | 1.75    | 0.25    | 0.25    | 1.75    | 1       |
| $m_3$ | 0.2     | 1.8     | 1.8     | 0.2     | 0.2     | 0.2     |
| $f_1$ | 1       | 0       | 1       | 0       | 1       | 1       |
| $f_2$ | 0       | 1       | 1       | 1       | 1       | 0       |

The MSD system has 12 varying entries in its system matrices as can be seen in (3.41) and (3.42). When the LV-based method is used separately for each of these 12 entries, a model set consisting of 12 models would result. However, many of these entries vary in a similar manner. Therefore, the number of local models can be reduced to 4 local models in case of the considered MSD system. Each of these local models corresponds to one of the 4 varying parameters. The model set

constructed with the LV-based method then has the following form

$$x_{k+1} = \left( T_s(A_{\text{MSD}}^+ + \mu_{k_1}A_{\text{MSD}}^{k_1} + \mu_{m_3}A_{\text{MSD}}^{m_3}) + I \right) x_k + T_s(B_{\text{MSD}}^+ + \mu_{m_3}B_{\text{MSD}}^{m_3} + \mu_{f_1}B_{\text{MSD}}^{f_1})u_k + w_k, \quad (3.44)$$

$$y_k = (C_{\text{MSD}}^+ + \mu_{f_2}C_{\text{MSD}}^{f_2})x_k + v_k, \quad (3.45)$$

where

$$A_{\text{MSD}}^+ = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -\frac{k_1^{\max}+k_3}{\frac{m_1}{k_1^{\max}}} & \frac{k_1^{\max}}{\frac{m_1}{k_1^{\max}}} & \frac{k_3}{\frac{m_1}{k_1^{\max}}} & -\frac{b_1+b_3}{\frac{m_1}{k_1^{\max}}} & \frac{b_1}{\frac{m_1}{k_1^{\max}}} & \frac{b_3}{\frac{m_1}{k_1^{\max}}} \\ \frac{k_1^{\max}}{\frac{m_2}{k_1^{\max}}} & -\frac{k_1^{\max}+k_2+k_4}{\frac{m_2}{k_1^{\max}}} & \frac{k_4}{\frac{m_2}{k_1^{\max}}} & \frac{b_1}{\frac{m_2}{k_1^{\max}}} & -\frac{b_1+b_2+b_4}{\frac{m_2}{k_1^{\max}}} & \frac{b_4}{\frac{m_2}{k_1^{\max}}} \\ \frac{k_3}{\frac{m_3}{k_3^{\max}}} & \frac{k_4}{\frac{m_3}{k_3^{\max}}} & -\frac{k_3+k_4}{\frac{m_3}{k_3^{\max}}} & \frac{b_3}{\frac{m_3}{k_3^{\max}}} & \frac{b_4}{\frac{m_3}{k_3^{\max}}} & -\frac{b_3+b_4}{\frac{m_3}{k_3^{\max}}} \end{bmatrix}, \quad (3.46)$$

$$A_{\text{MSD}}^{k_1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{k_1^{\max}-k_1^{\min}}{\frac{m_1}{k_1^{\max}}} & \frac{k_1^{\min}-k_1^{\max}}{\frac{m_1}{k_1^{\max}}} \\ \frac{k_1^{\min}-k_1^{\max}}{\frac{m_2}{k_1^{\max}}} & \frac{k_1^{\max}-k_1^{\min}}{\frac{m_2}{k_1^{\max}}} \\ 0 & 0 \end{bmatrix} \mathbf{0}_{6 \times 4}, \quad B_{\text{MSD}}^+ = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ f_1^{\max} \frac{1}{m_1} & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_3^{\max}} \end{bmatrix}, \quad (3.47)$$

$$A_{\text{MSD}}^{m_3} = \begin{bmatrix} \mathbf{0}_{5 \times 6} \\ \left[ \frac{1}{m_3^{\min}} \quad \frac{1}{m_3^{\max}} \right] \begin{bmatrix} k_3 & k_4 & -(k_3+k_4) & b_3 & b_4 & -(b_3+b_4) \\ -k_3 & -k_4 & k_3+k_4 & -b_3 & -b_4 & b_3+b_4 \end{bmatrix} \end{bmatrix}, \quad (3.48)$$

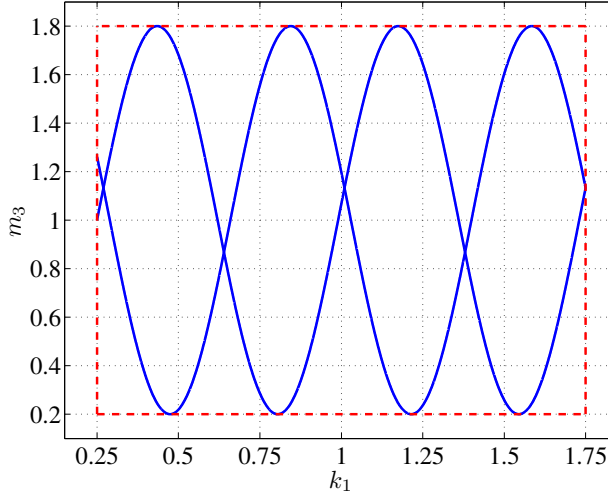
$$B_{\text{MSD}}^{f_1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ f_1^{\min} - f_1^{\max} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad B_{\text{MSD}}^{m_3} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_3^{\min}} - \frac{1}{m_3^{\max}} \end{bmatrix}, \quad (3.49)$$

$$C_{\text{MSD}}^+ = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & f_2^{\max} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad C_{\text{MSD}}^{f_2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & f_2^{\min} - f_2^{\max} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (3.50)$$

and  $\mathbf{0}_{i \times j} \in \mathbb{R}^{i \times j}$  denotes a zero matrix.

### 3.6.2 Simulation Results

The three model sets constructed in the previous section are evaluated by assessing the performance of the three corresponding MM filters, all of which are joint EKFs as described in Chapter 2. For this purpose, a continuous-time simulation has been performed with the parameter-varying MSD system in which the aim was to simulate as many system conditions as possible. During the simulation, the parameters  $k_1$  and  $m_3$  are varied such that a large part of the parameter space of these parameters is covered. The simulated trajectory in the parameter space of  $k_1$  and  $m_3$  is depicted in Figure 3.6. Furthermore, two faults are inserted sequentially during the simulation. A partial actuator fault is inserted in the time interval 30–50 s and a partial sensor fault is inserted in the time interval 70–80 s. Furthermore, measurement noise is added such that the three measurements have an SNR of 60 dB.



**Figure 3.6:** Simulated trajectory in the parameter space of  $k_1$  and  $m_3$ .

Estimates of the parameters are required for some purposes. For example, in case of the considered MSD system estimates of parameters  $f_1$  and  $f_2$  are required in order to perform FD. However, there is no direct relation between the estimated model weights and the parameters when the CP-based model set design method is used. Therefore, a procedure is proposed for (indirectly) reconstructing the values of the parameters from the estimated model weights. This procedure determines which model from  $\mathbf{M}_\ell$  has the smallest distance to the estimated model at each time step. The parameters of the model with the smallest distance are then used as the reconstructed parameters. The estimated model at time step  $k$  can be computed as

$$\hat{M}_k^{\text{CP}} = U_{\mathbf{M}_\ell}^1 \begin{bmatrix} v^{(1)} & \dots & v^{(N_h)} \end{bmatrix} \hat{\mu}_k^{\text{CP}}, \quad (3.51)$$

where  $\hat{\mu}_k^{\text{CP}}$  denotes the vector with the estimated model weights of the local models in the CP-based model set. Subsequently, the model from  $\mathbf{M}_\ell$  that is closest to  $\hat{M}_k^{\text{CP}}$ , can be computed as

$$n_s = \arg \min_{i \in \{1, 2, \dots, N_\ell\}} \left\| \mathfrak{M}^{(i)} - \hat{M}_k^{\text{CP}} \right\|_2, \quad (3.52)$$

where  $n_s$  corresponds to the index of the model from the large model set that is closest to  $\hat{M}_k^{\text{CP}}$ . It should be noted that also for the OD-based model set, there is not always a direct relation between the estimated model weights and the parameters. In this case, the described procedure can also be used.

Using the model sets designed with the three proposed methods and the input-output data from the previously described simulation, the state of the system and the model weights of the local models are estimated. The state estimation per-

formance is evaluated by computing the variance accounted for (VAF) for all six estimated states. The VAF is defined as

$$\text{vaf}(x, \hat{x}_k) = \max \left\{ 1 - \frac{\text{var}(x_k - \hat{x}_k)}{\text{var}(x_k)}, 0 \right\} \times 100\% \quad (3.53)$$

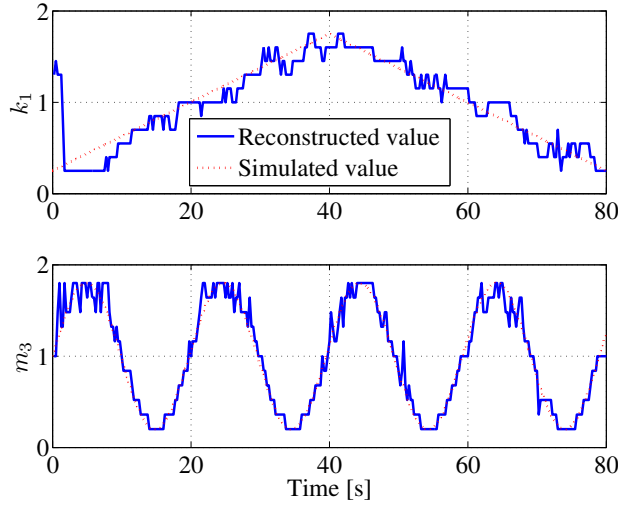
where  $\text{var}(\cdot)$  denotes the variance of the corresponding signal. In Table 3.3 it can be seen that the VAF values are well above 90%. The estimation results of the model weights are presented separately for each of the three model sets in the following.

**Table 3.3:** VAF values of the states estimated by using the three model sets designed with the three proposed methods.

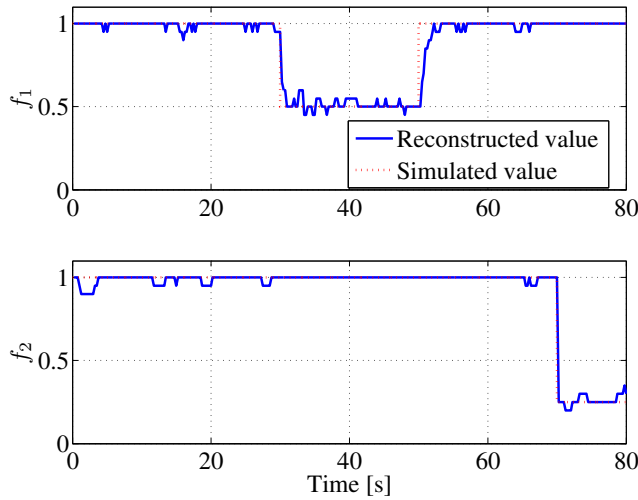
|                              | OD-based<br>model set | CP-based<br>model set | LV-based<br>model set |
|------------------------------|-----------------------|-----------------------|-----------------------|
| $\text{vaf}(x_1, \hat{x}_1)$ | 100%                  | 100%                  | 100%                  |
| $\text{vaf}(x_2, \hat{x}_2)$ | 95.9%                 | 95.4%                 | 91.5%                 |
| $\text{vaf}(x_3, \hat{x}_3)$ | 100%                  | 100%                  | 100%                  |
| $\text{vaf}(v_1, \hat{v}_1)$ | 99.2%                 | 99.5%                 | 99.9%                 |
| $\text{vaf}(v_2, \hat{v}_2)$ | 97.1%                 | 96.8%                 | 95.7%                 |
| $\text{vaf}(v_3, \hat{v}_3)$ | 99.1%                 | 99.5%                 | 99.7%                 |

The estimated model weights of the model set designed with the OD-based method do not have a direct relation with all parameters. Therefore, the previously described procedure to reconstruct these parameters from the estimated model weights is used. The results of this procedure are depicted in Figures 3.7 and 3.8. In these two figures it can be seen that the simulated values are generally tracked well by the reconstructed values. Note that in both figures the parameters are discretely valued. The reason for this is that the reconstruction procedure is based on  $M_L$ , which contains models sampled at discrete levels of the parameter space (see Figure 3.5 (left)). Note also that the reconstructed values of the fault parameters, depicted in Figure 3.8, allow an accurate and quick diagnosis of the simulated faults. As is previously mentioned, the estimated model weights of the local models from the OD-based model set do not have a direct relation with all parameters. However, for the two fault parameters  $f_1$  and  $f_2$  there exists such a relation. This relation becomes clear if the estimated model weights of the models that correspond to the same system condition, i.e. the nominal condition or one of the two fault conditions, are summed together. This means that for the model set described in Table 3.2, the weights of models 3 and 5, the weights of models 1 and 6, and the weights of models 2 and 4 should be summed together. The result of these three summations are depicted in Figure 3.9. From this figure it is clear that the relations  $f_1 = 1 - (\hat{\mu}^2 + \hat{\mu}^4)$  and  $f_2 = 1 - (\hat{\mu}^1 + \hat{\mu}^6)$  hold. Moreover, it can be seen that diagnosis of the faults is very well possible from the summed weights. Therefore, it can be concluded that if FD were the only purpose of the MM system and the values of parameters  $k_1$  and  $m_3$  were not needed, the parameter recon-





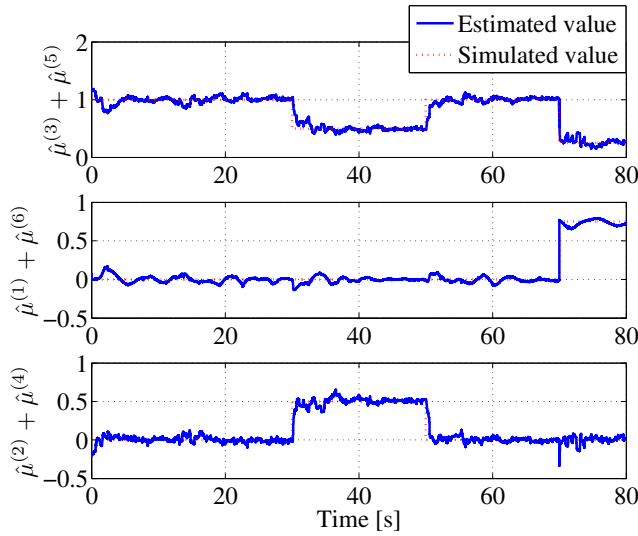
**Figure 3.7:** Reconstructed values of  $k_1$  and  $m_3$  using the model set designed with the OD-based method.



**Figure 3.8:** Reconstructed values of  $f_1$  and  $f_2$  using the model set designed with the OD-based method.

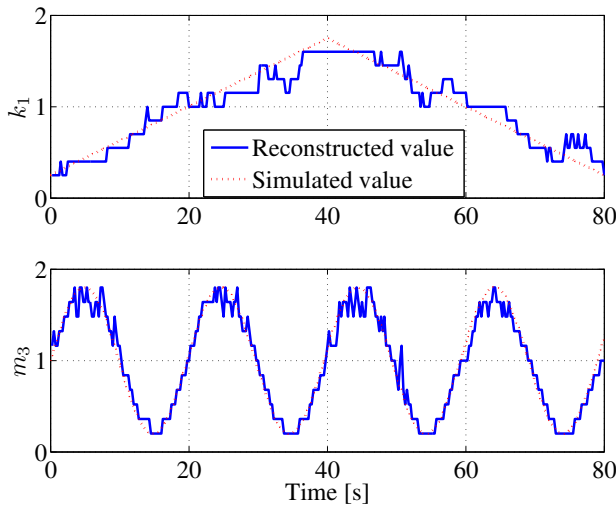
struction procedure could be omitted for the OD-based model set.

The estimated model weights of the model set designed with the CP-based method cannot be directly related to the varying parameters because the local models do not correspond to physically interpretable conditions. Therefore, also here the previously described procedure to reconstruct these parameters is used.

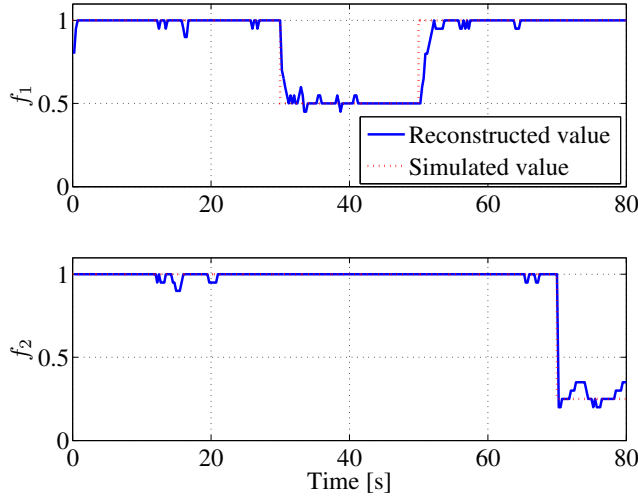


**Figure 3.9:** Sums of the estimated model weights of models 3 and 5 (both corresponding to a nominal condition), models 1 and 6 (both corresponding to a sensor fault) and models 2 and 4 (both corresponding to an actuator fault).

The results of this procedure are shown in Figures 3.10 and 3.11. In these figures it can be seen that the simulated values are tracked well by the reconstructed values. Again the reconstructed values are discretely valued as a result of the parameter reconstruction procedure.



**Figure 3.10:** Reconstructed values of  $k_1$  and  $m_3$  using the model set designed with the CP-based method.



**Figure 3.11:** Reconstructed values of  $f_1$  and  $f_2$  using the model set designed with the CP-based method.

The model weights of the local models designed with the LV-based method do have a direct relation to the model parameters. In the case of the considered MSD system, the following relations can be deduced for the different parameters

$$k_1 = k_1^{\max} + \mu_{k_1} (k_1^{\min} - k_1^{\max}), \quad (3.54)$$

$$m_3 = \frac{m_3^{\max} m_3^{\min}}{m_3^{\min} + \mu_{m_3} (m_3^{\max} - m_3^{\min})}, \quad (3.55)$$

$$f_1 = 1 - \mu_{f_1}, \quad (3.56)$$

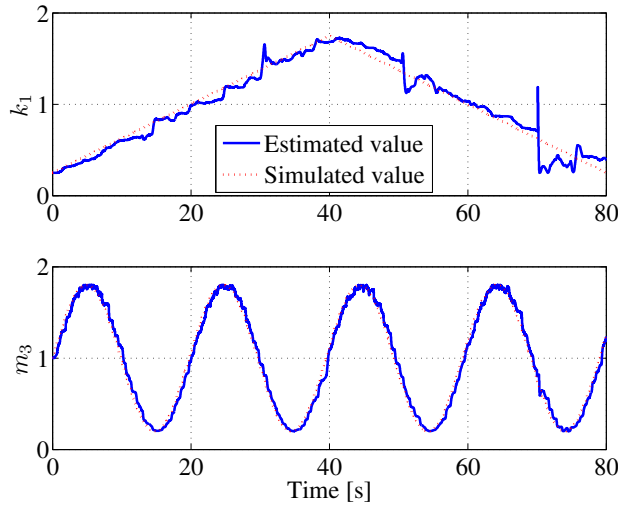
$$f_2 = 1 - \mu_{f_2}. \quad (3.57)$$

It should be noted that the specific way in which the parameters vary in the MSD system, makes their reconstruction relatively easy. If the parameters would vary in a strongly nonlinear way, this reconstruction would be more involved.

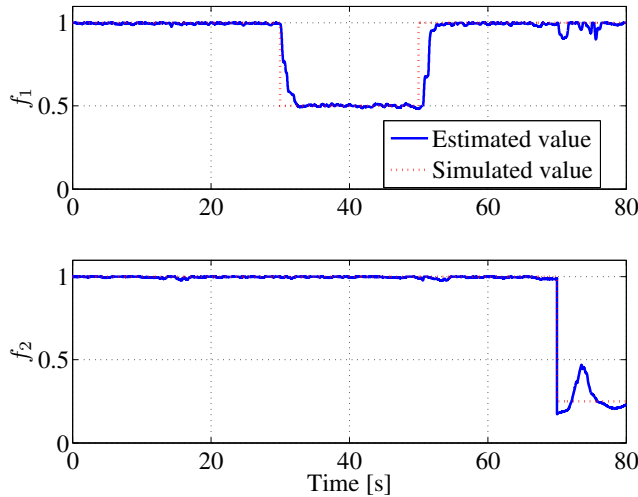
The estimation results obtained with the LV-based method are depicted in Figures 3.12 and 3.13. Because of the direct relation between the model weights and the parameters, the parameters are estimated more smoothly than in case of the OD-based and CP-based model sets. The estimated fault parameters,  $f_1$  and  $f_2$ , depicted in Figure 3.13 allow diagnosis of the faults that are simulated.

### 3.6.3 Differences Between the Model Set Design Methods

The presented simulation results indicate that all three methods can be successfully applied to model set design of the parameter-varying MSD system. However, there are differences between the presented methods that can make one



**Figure 3.12:** Estimated values of  $k_1$  and  $m_3$  using the model set designed with the LV-based method.



**Figure 3.13:** Estimated values of  $f_1$  and  $f_2$  using the model set designed with the LV-based method.

method to be preferred over the other. A specific case is when local models can be generated without exact knowledge of how the system entries vary as a function of the parameters. This knowledge is required for the LV-based method but not for the OD-based and CP-based methods. As long as local models can be generated at desired operating conditions, the latter two methods can be applied. An example

of an application in which local models at different conditions can be generated without exact knowledge of the variation of the entries is the Boeing 747 benchmark model considered in this thesis. This nonlinear model can be trimmed and linearized at a desired flight condition resulting in a local linear model valid at this flight condition. In Chapter 4 model sets for the benchmark model will be generated using the OD-based method.

Another difference between the presented methods is how estimates of the parameters are obtained from the model weights. An important property of the CP-based method is that the estimated model weights do not have a direct relation to the parameters. Therefore, the estimates of the parameters have to be reconstructed with the proposed procedure. In case of the OD-based method, there can be a relation between the weights and the parameters as is shown in the presented example. However, if this is not the case then also for the OD-based method the reconstruction procedure should be used. When the LV-based method is used, the model weights and parameters have a direct relation. The reconstruction procedure is therefore not necessary.

A difference between the OD-based method and CP-based method is that the CP-based method is more flexible in its choice of local models. The reason for this is that the OD-based method is restricted to including only the models from the large model set in the reduced model set. The CP-based does not have this restriction. It is therefore possible that the CP-based model set can result in a better conditioned estimation problem than the OD-based method as a result of this flexibility.

The results obtained with the three model sets design methods are not significantly different for the presented example. However, the three methods do have different properties. It can therefore be concluded that the method that is to be preferred for a specific application depends on these properties.

### 3.6.4 Comparison with the “Minimum-Mismatch” Method

In Section 3.1 the model set design methods presented by Li et al. (2005) were mentioned. These methods use a different design philosophy from the methods presented in this chapter. The main difference is that the methods presented by Li et al. (2005) are based on the PDF of the varying parameters, while the methods presented in this chapter are based on the varying parameters themselves. In order to make a comparison between these design philosophies, the “minimum-mismatch” model set design method presented by Li et al. (2005) is used to create a model set for the MSD system. The main idea of this method is to approximate the cumulative distribution function (CDF) of a varying parameter by a discrete CDF. The model set is computed by partitioning the parameter space into equally probable regions based on the CDF. Subsequently the local models are chosen such that the local models correspond to the centers of each region. A drawback of this method that it is applicable to the CDF of only one parameter. In order to allow comparison with the model set design methods presented in this chapter, the

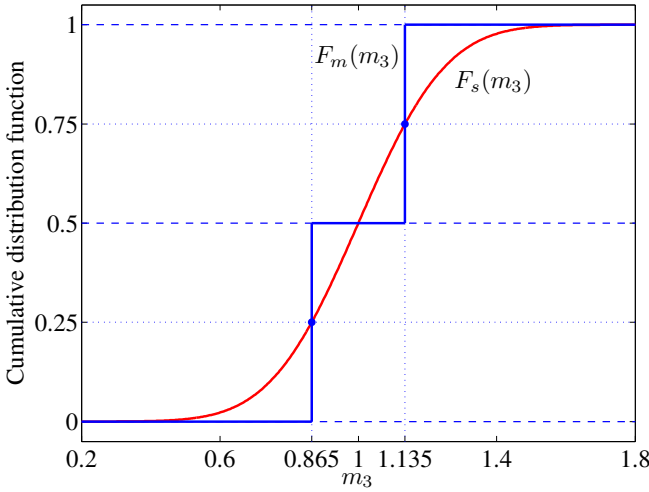
MSD system with only one varying parameter is considered. The parameter  $m_3$  is chosen to be this single varying parameter. Parameters  $k_1$ ,  $f_1$ , and  $f_2$  are chosen as constant parameters, all with a value of 1. The “minimum-mismatch” model set design method creates a model set that consists of (physically interpretable) models of the form (3.1)-(3.2), sampled at some  $\rho(t) \in \mathcal{R}$ . In this sense, this model set can be best compared with the model set obtained with the OD-based model set design method. Therefore, in the following a comparison is made between these two methods only.

The minimum number of local models required to create a model set with the OD-based method is 2, for the case that  $m_3$  is the only varying parameter. The model set created with the “minimum-mismatch” model set design method is therefore also chosen to consist of two local models. It is assumed that the PDF of  $m_3$  is Gaussian:

$$f_s(m_3) = \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{(m_3-1)^2}{2\sigma_0^2}}, \quad (3.58)$$

with a standard deviation of  $\sigma_0 = 0.2$ . The CDF of  $m_3$  is the integral of  $f_s(m_3)$  and is denoted by  $F_s(m_3)$ . The CDF is partitioned into two equally probable regions. These two regions are indicated by dashed lines in Figure 3.14. Next, the centers of these regions, which are indicated by dotted lines, are determined. The discrete CDF  $F_m(m_3)$  can then be constructed as a stair-case type function that goes through the points at which the centers of the regions intersect with  $F_s(m_3)$ . This stair-case type function is also depicted in Figure 3.14. Furthermore, it can be seen in this figure that for one local model obtained with this method it holds that  $m_3 = 0.865$  and for the other local model it holds that  $m_3 = 1.135$ .

The two local models obtained with the OD-based method have values for  $m_3$  of 0.2 and 1.8, i.e. the minimum and maximum value. This model set is compared to the model set obtained with the “minimum-mismatch” method by determining for which model set the combined model weight and state estimation problem is best conditioned. For this purpose, Definition 3.1 of strong local observability is used. Different state realizations are generated by simulating the MSD system with different randomly generated sequences of  $m_3$ . The values of both input signals are chosen as 1. For each state realization, the mean condition number of the nonlinear observability matrix is computed. The model weights that are also required for the computation of the nonlinear observability matrix can be computed from the randomly generated sequences of  $m_3$  for each of the two model sets. The mean condition number of the observability matrix for 5 different state realizations and for both model sets is shown in Table 3.4. In this table it can be seen that the mean condition number of the observability matrix for the “minimum-mismatch” model set is higher than for the OD-based model set in case of all 5 realizations. This observation is not surprising because the OD-based model set is specifically designed such that the combined model weight and state estimation problem is well-conditioned. A conclusion that can be drawn from this experiment is that the model set obtained with the OD-based method is to be preferred over the model set obtained from the “minimum-mismatch” design method when the MM structure defined in (3.3)-(3.4) is used.



**Figure 3.14:** Cumulative distribution function of  $m_3$  denoted by  $F_s(m_3)$ . The dots on this function correspond to the points at which two local models are selected according to the “minimum-mismatch” model set design method presented by Li et al. (2005).

**Table 3.4:** Mean condition number of the nonlinear observability matrix computed for 100 samples and for 5 different state realizations.

|   | OD-based<br>model set | “minimum-mismatch”<br>model set |
|---|-----------------------|---------------------------------|
| 1 | $1.39 \cdot 10^5$     | $2.24 \cdot 10^6$               |
| 2 | $1.55 \cdot 10^5$     | $2.5 \cdot 10^6$                |
| 3 | $1.32 \cdot 10^5$     | $2.13 \cdot 10^6$               |
| 4 | $1.39 \cdot 10^5$     | $2.24 \cdot 10^6$               |
| 5 | $2.21 \cdot 10^5$     | $3.57 \cdot 10^6$               |

### 3.7 Conclusions

An important issue that should be considered when using the multiple-model framework is how the model set is designed. This model set should be as compact as possible and at the same time it should make a good approximation of the system to be modeled. In this chapter three methods have been proposed for the structured design of model sets for multiple-model systems. Two of the proposed methods are based on the availability of a large model set that contains local models sampled from the system to be modeled at the desired operating conditions. One of these two methods reduces the large model set by making a selection of the models such that this selection of models renders the estimation problem as

well-conditioned as possible. This selection is made by using orthogonal decompositions. The other of these methods reduces the large model set by enclosing all local models in a convex polytope. The local models are formed by the vertices of this polytope. The main difference between these two methods is that the local models obtained with the model set design method based on orthogonal decompositions correspond to physically interpretable system conditions. This is not the case for the model set that consists of the vertices of a convex polytope. Moreover, a third method is presented that does not require the availability of a large initial model set. Instead this method creates a model set based on the limit values of the varying parameters of the system. The multiple-model structure for the model sets created with this third method is slightly different from the structure of the first two methods.

The three methods have been applied to the design of model sets for a linear parameter-varying model of a mass-spring-damper system with four varying parameters. Two of these parameters are fault parameters. The simulation of the model has been performed with variations in all four parameters. Subsequently, the state and model weights are estimated based on the three designed model sets. The state estimation results show good performance for all three model sets. The estimated model weights based on the three model sets cannot be compared directly with each other because of the different properties of the local models in the model sets. Instead, the estimated model weights are used to reconstruct the values of the parameters. For the method based on limit values, a direct relation exists between the estimated weights and the parameters. For the other two methods, such a direct relation does not exist. Therefore, a reconstruction procedure is used to reconstruct the parameters from the estimated model weights. The results for all three methods are comparable; the varying parameters are tracked well and a quick and accurate diagnosis is possible from the two fault parameters. For the model set created by using orthogonal decompositions, it should be remarked that the reconstruction procedure is not required for the two fault parameters. It is shown that, for the considered example, these two parameters have a direct relation with the estimated model weights. Since the results obtained with the three model sets design methods are comparable for the presented example, which of the methods is to be preferred over the other depends on the type of application.

A comparison has been made between the OD-based model set design method and the “minimum mismatch” model set design method presented by Li et al. (2005). It has been concluded that when using an MM structure that explicitly interpolates between the local models, the OD-based method is to be preferred for the comparison example.



## Model Sets for Aircraft Fault Diagnosis

---

In Chapter 3, three model set design methods have been described. In this chapter one of these methods, namely the orthogonal decomposition based method, is used to create model sets for fault diagnosis of the Boeing 747 benchmark model. The considered faults include sensor faults, actuator faults, and component faults such as the faults that occurred during the “Bijlmerramp”. Both linear and nonlinear simulations are performed to evaluate the modeling accuracy of the designed model sets. The nonlinear simulations are used to assess the ability to isolate faults, while the linear simulations are used to assess the ability to also identify the faults.

---

### 4.1 Introduction

The main motivation for using the multiple-model (MM) framework for fault detection and isolation (FDI) is that it allows a large class of fault conditions to be modeled. These fault conditions include sensor faults, actuator faults as well as component faults. The MM framework proposed in this thesis distinguishes itself from the commonly used MM framework (Li and Jilkov 2005) by the fact that it explicitly allows weighted combinations of local models. This property allows the use of relatively small model sets that at the same time can represent a large number of fault conditions. The philosophy behind the proposed MM framework is that local models in the model set form a basis of the fault conditions that may occur. In other words, these fault conditions are represented by a weighted combination of the local models in the model set. This is in contrast to the conventional MM framework, in which it is common practice to model each considered fault condition by a separate local model (Zhang and Jiang 2001; Diao and Passino 2002).

In critical situations in aircraft, completely relying on fault-tolerant control (FTC) might not be desirable in some situations. Therefore, keeping the pilot in

the loop is important for FTC (Maciejowski and Jones 2003). For this reason, the pilot should also have information on the faults that have occurred. In this chapter, this fault information is obtained by MM estimation. The main contribution of this chapter is the assessment of the fault isolation and identification capabilities of the proposed fault modeling strategy using a realistic benchmark model. The difference between fault isolation and identification is that for isolation only the location of faults is to be determined. For identification the size of faults should also be determined in addition to their location. The assessment of the identification capabilities is performed using linear simulation, while the assessment of the isolation capabilities is performed using nonlinear simulation. The reason for this is that for identification less model error is allowed between the system under diagnosis (i.e. the simulation model) and the model that represents this real system (i.e. the weighted combination of the local models in the model set). Both simulations use models, either linear or nonlinear, of the Boeing 747 benchmark model.

The primary fault type that is considered in this chapter is lock-in-place of control surfaces. Because of its challenging nature from an FTC point view it is also the main fault type considered in the Boeing 747 benchmark model. In order to display the versatility of the MM framework in combination with a structured model set design method, in this chapter also sensor faults and component faults are considered. The considered components faults are a hydraulic outage of one of the 4 available hydraulic systems and engine separation of two engines on one side of the aircraft such as had occurred during the “Bijlmerramp” (Smaili and Mulder 2000). The method based on orthogonal decomposition (OD) for model set generation is used to create the model sets. This method has been described in Section 3.3.

Modeling of lock-in-place faults in an MM setting has also been considered in the work by Liao et al. (2005). In this work, lock-in-place faults are modeled by model sets containing local models that correspond to conditions at which a control surface is stuck at its upper or lower limit. These model sets are derived rather heuristically instead of using a structured approach. Furthermore, these model sets have only been used for the synthesis of a passive fault-tolerant controller, which is robust with respect to all faults modeled by the convex hull of the models from the designed model sets. This passive FTC method requires no information on the actual condition of the system, which is therefore not estimated. The purpose of the model sets developed by Liao et al. (2005) is therefore different from the purpose of the model sets in this chapter.

This chapter is organized as follows. Section 4.2 provides a discussion on lock-in-place faults. Next, Section 4.3 presents two model sets obtained with the OD-based model set design method. The simulation results are provided in Section 4.4. Finally, conclusions are provided in Section 4.5.

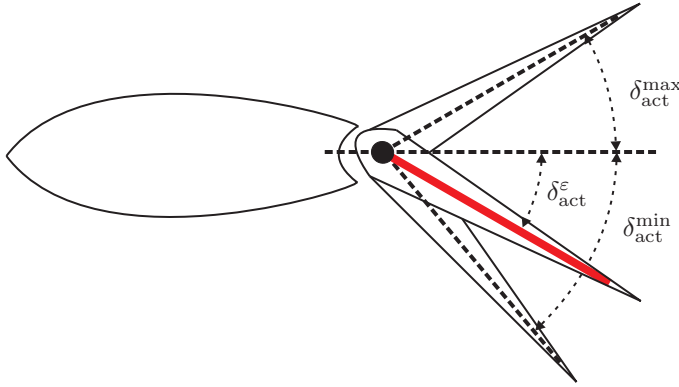
## 4.2 Lock-in-Place Faults

One of the fault types considered in this chapter is lock-in-place of control surfaces. This fault type is characterized by a control surface that freezes at a certain position and does not react to the control commands anymore. The motivation for considering this fault type stems from a number of aircraft incidents. Three examples of such incidents are the incidents that have occurred with earlier versions of the Boeing 737, as is described in Chapter 1 on page 3. All of these three incidents were most probably caused by runaway of the rudder surface to its maximum or minimum deflection position. Note that because runaway of a control surface is also characterized by freezing of the surface, it can be considered to be a lock-in-place fault. This class of faults can result in severe performance degradation of the aircraft depending on the position at which the control surface freezes. For example, rudder runaway is much more severe than lock-in-place of the rudder at its neutral position. The possibly catastrophic nature of this class of faults makes it very challenging from an FTC point of view. For this reason, this class of faults is the main fault type that is considered in GARTEUR AG-16. In fact, besides the “Bijlmerramp” scenario and a loss of the vertical tail plane it is the only fault type considered in GARTEUR AG-16. The project considers lock-in-place faults of different groups of control surfaces such as elevators, ailerons, and rudders. In the benchmark model it is assumed that these groups of control surfaces move and fail together.

A straightforward method to diagnose lock-in-place faults is to install surface sensors on the control surfaces of the aircraft. If the actual deflection of the surfaces measured by the surface sensors does not correspond to the deflection that could be expected from the actuator command signal, then a fault is detected. Such an approach has been pursued by Varga (2007) for the Boeing 747 benchmark model. Although the number of surface sensors is kept minimal, i.e. not all surfaces are equipped with sensors, the additional sensors still add to hardware costs and maintenance of the aircraft. Furthermore, faults in the surface angle sensors should also be considered when using these additional sensors, which has not been done by Varga (2007). In this chapter a different philosophy is used to diagnose lock-in-place faults to overcome these drawbacks. Instead of using additional surface sensors, only the sensors that measure important flight parameters are used. These sensors are already available on the aircraft and they are fitted redundantly, as a result of which they can be considered to be practically fault-free. By using these existing sensors, the method that is adopted for fault diagnosis in this chapter does not require additional hardware. The diagnosis is therefore based on analytical redundancy rather than hardware redundancy.

The problem of modeling lock-in-place faults in aircraft using the MM framework has also been considered by Liao et al. (2005). These authors model lock-in-place faults on a fighter aircraft by using a polytopic model. The vertices of this polytopic model consist of local models that correspond to the cases in which the control surfaces are stuck at one of their two outer limits. The polytopic model, however, also approximately describes the aircraft in case the corresponding con-

trol surfaces were stuck at any other position between the two outer limits. In Figure 4.1, a graphical representation is given of the range of lock-in-place faults that can be approximately described by two local models. In this figure, a control surface is depicted with its maximum and minimum deflection,  $\delta_{act}^{max}$  and  $\delta_{act}^{min}$ , respectively. The two models to be included in the model set, correspond exactly to the conditions in which the aircraft has the control surface stuck at its maximum or minimum deflection. An intermediate lock-in-place fault of the same control surface, as is represented by  $\delta_{act}^{\varepsilon}$  in Figure 4.1, can be approximated by a convex combination of the two models.



**Figure 4.1:** Outer limits of a control surface.

Lock-in-place faults only affect the control surfaces. In a state-space setting it might therefore be expected that this fault type can be modeled by modifying only the  $B$ -matrix of the nominal model. However, a control surface that is stuck at a certain fixed position influences the complete dynamics of the aircraft. It is therefore pointed out in Liao et al. (2005) that the local linear models corresponding to lock-in-place faults differ in both the  $A$ -matrix and the  $B$ -matrix from the nominal model.

### 4.3 Model Set Design

In this section, the OD-based model set design method is used to design two model sets to perform fault diagnosis (FD) of the Boeing 747 benchmark model. The OD-based model set design method is one of the three methods presented in Chapter 3 for structured model set design. The other two methods are the method based on a convex polytope (CP) and the method based on limit values (LV). The CP-based method is not chosen since its local models do not correspond to physically interpretable models. This property prohibits the direct diagnosis of faults from the estimated model weights, which is desired in this section. The LV-based method is not chosen since the exact physical equations that describe the dynamics of the Boeing 747 aircraft are not directly available from the benchmark model.

The first model set to be designed is primarily meant to display the versatility of the model set design method. It therefore considers actuator faults, component faults, and a range of faults in one sensor. The second model set is designed with the purpose to isolate the different expected lock-in-place faults considered in the benchmark model.

### 4.3.1 Description of the Linearized Models

In order to obtain the local linear models required for the OD-based model set design method, the nonlinear model is linearized for a straight-and-level flight at  $h = 980$  m,  $V_{TAS} = 92.6$  m/s, with a flap setting of 20 deg, and a total weight of  $263 \cdot 10^3$  kg. This operating point corresponds to the initial condition of the benchmark trajectory. The same operating point is used for the nominal model as well as the fault models. All linearized models are  $10^{\text{th}}$  order models of the longitudinal and lateral/directional motion of the aircraft and have 6 inputs. An overview of the states and inputs of the linearized models is given in Table 4.1. In the linearization routine of the benchmark it has been assumed that similar surfaces operate and fail together. For example, the inner right, inner left, outer right and outer left elevator surface have been lumped into one generic elevator surface controlled by the input  $\delta_e$ . Note that this has been done for simplification purposes only and that the presented modeling method can be modified to include separate commands for each individual control surface.

**Table 4.1:** Description of the linearized model.

| States                           | Inputs                                      |
|----------------------------------|---------------------------------------------|
| $p$ : roll rate [deg/s]          | $\delta_a$ : aileron deflection [deg]       |
| $q$ : pitch rate [deg/s]         | $\delta_{sp}$ : spoiler deflection [deg]    |
| $r$ : yaw rate [deg/s]           | $\delta_e$ : elevator deflection [deg]      |
| $V_{TAS}$ : true airspeed [m/s]  | $\delta_{st}$ : stabilizer deflection [deg] |
| $\alpha$ : angle of attack [deg] | $\delta_r$ : rudder deflection [deg]        |
| $\beta$ : sideslip angle [deg]   | $\delta_T$ : Thrust [N]                     |
| $\phi$ : roll angle [deg]        |                                             |
| $\theta$ : pitch angle [deg]     |                                             |
| $\psi$ : yaw angle [deg]         |                                             |
| $h$ : altitude [m]               |                                             |

The linearization routine of the benchmark delivers continuous-time models. Therefore, the local models are first discretized according to the procedure described in Appendix B. The values for the parameters required for this procedure are chosen as:  $T = 0.01$  s and  $M=1$ . The error bound defined in (B.5) can be computed to be in the order of  $10^{-4}$  for this particular choice of parameters ( $\|A\| \approx 3$ , for all local models), which results in sufficiently accurate local models for FD.

### 4.3.2 Incorporating Different Trim Offsets

Before the benchmark model is linearized, it first has to be trimmed such that there is an equilibrium of forces and moments at the desired operating point of the aircraft. The resulting trim values for the input vector and the state vector are slightly different in case the local models correspond to different fault conditions near the same operating point. The trimming routine therefore causes that the local models are linearized at different equilibria. This issue has been resolved by using the following model description

$$x_{k+1} = \sum_{i=1}^N \mu_k^{(i)} [A_d^{(i)}(x_k - x^{(o,i)}) + B_d^{(i)}(u_k - u^{(o,i)}) + x^{(o,i)}] + w_k, \quad (4.1)$$

$$y_k = \sum_{i=1}^N \mu_k^{(i)} C^{(i)} x_k + v_k. \quad (4.2)$$

This description is a discretized version of (3.3)-(3.4) and is extended such that it can deal with different trim offsets. In this description,  $x^{(o,i)}$  and  $u^{(o,i)}$  correspond to the offsets in the state and input of the  $i$ -th local model, respectively. The matrices  $A_d$  and  $B_d$  denote discretized versions of their continuous-time counterparts  $A$  and  $B$ . Note that the discretized model does not have a  $D$ -matrix, because the aircraft model does not have direct feedthrough.

### 4.3.3 Model Set I

Besides modeling lock-in-place faults, model set I is designed to model multiplicative sensor faults in the angle of attack ( $\alpha$ ) sensor. Moreover, the model set also models two fault conditions of the aircraft that occurred during the “Bijlmer-ramp”. One condition corresponds to an outage of hydraulic system # 4, which is one of the 4 hydraulic systems on a Boeing 747. This fault results in loss of control of a number of control surfaces. The other condition corresponds to a separation of both engines on the right wing. This condition also includes the weight loss, center of gravity change, and wing damage that resulted from the separation. In Figure 4.2, the locations on the aircraft that are affected by the considered faults are highlighted. The surfaces affected by the hydraulic outage are indicated by hatching or light shading.

As explained in Section 3.3, the OD-based method requires the generation of a large number of models that should be arranged in a matrix  $\mathbf{M}_\ell$ . These models correspond to samples from the set of system conditions to be modeled. For model set I, a matrix  $\mathbf{M}_\ell^I$  is created that consists of the models described in Table 4.2.

By using  $\mathbf{M}_\ell^I$  as a basis, the model set described in Table 4.3 is designed. The number of local models has been chosen to be 8, according to the rules provided in Section 3.3. The order in which the local models are listed in Table 4.3 is directly determined by the OD-based model set design algorithm. So, the “most influential” model is listed at the top of the table. It can be seen in this table that the faults

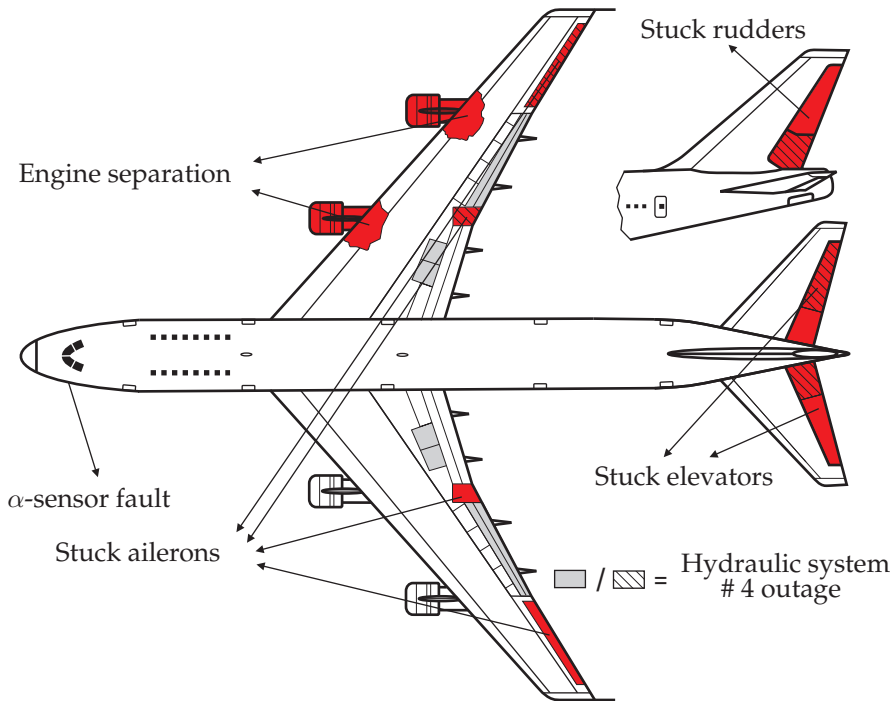


Figure 4.2: Place of occurrence of the faults on the aircraft.

**Table 4.2:** Description of models in  $M_\ell^I$  used to create model set I.

| Fault type                  | Sampled fault sizes           | Number of models |
|-----------------------------|-------------------------------|------------------|
| Rudder lock-in-place        | $\{-25, -24, \dots, 24, 25\}$ | 51               |
| Elevator lock-in-place      | $\{-23, -22, \dots, 16, 17\}$ | 41               |
| Fault in $\alpha$ -sensor   | $\{0, 0.1, \dots, 1\}$        | 11               |
| Hydraulic system # 4 outage | -                             | 1                |
| Engine separation           | -                             | 1                |

that can occur with different sizes are represented by local models corresponding to the maximum fault sizes. For example, rudder lock-in-place faults in the range  $[-25, 25]$  deg are represented by two local models: one model with the rudder stuck at its upper limit and one model with the rudder stuck at its lower limit. For the range of elevator lock-in-place faults and multiplicative faults in the  $\alpha$ -sensor similar fault models appear in the model set. Note that the sensor fault model corresponding to 100% effectiveness is the same as the nominal model. Furthermore, it can be seen that the fault models corresponding to engine separation and the outage of hydraulic system # 4 also appear in the model set. This observation is according to the expectation that these two models have very specific dynamics that cannot be represented by a weighted combination of the other models in the model set.

**Table 4.3:** Description of model set I.

| Model # | Model description                                  |
|---------|----------------------------------------------------|
| 1       | Rudder stuck at upper limit (25 deg)               |
| 2       | Rudder stuck at lower limit (-25 deg)              |
| 3       | Elevator stuck at upper limit (17 deg)             |
| 4       | Elevator stuck at lower limit (-23 deg)            |
| 5       | Engine separation                                  |
| 6       | Total fault in $\alpha$ -sensor (0% effectiveness) |
| 7       | Hydraulic system # 4 outage                        |
| 8       | No fault in $\alpha$ -sensor (100% effectiveness)  |

#### 4.3.4 Model Set II

From model set I it can be concluded that even exceptional faults such as engine separation can be represented well in the model set. However, it is not reasonable to assume that such faults can be anticipated. Therefore, for model set II only faults that can reasonably be anticipated are considered. These faults are lock-in-place of the ailerons, rudders, and elevators. Again, a large number of local models is sampled from the system to be modeled. These sampled models, which are described in Table 4.4, are used to construct the matrix  $M_\ell^{II}$ .



**Table 4.4:** Description of models in  $M_{\ell}^{\text{II}}$  used to create model set II.

| Fault type             | Sampled fault sizes           | Number of models |
|------------------------|-------------------------------|------------------|
| Rudder lock-in-place   | $\{-25, -24, \dots, 24, 25\}$ | 51               |
| Elevator lock-in-place | $\{-23, -22, \dots, 16, 17\}$ | 41               |
| Aileron lock-in-place  | $\{-15, -14, \dots, 14, 15\}$ | 41               |

The application of the OD-based model set design method to  $M_{\ell}^{\text{II}}$ , results in the model set described in Table 4.5. The order in which the local models are listed is again directly determined by the model set design algorithm, which means that the “most influential” model is located at the top of the table. It can again be seen that the lock-in-place faults are represented in the model set by local models corresponding to lock-in-place faults at the maximum or minimum positions. In addition to these limit models, the elevator lock-in-place fault is represented by a local model corresponding to a lock-in-place fault at  $-6$  deg, which is located close to the middle of the deflection range of the elevator, i.e.  $[-23, 17]$  deg. From the fact that elevator lock-in-place faults are modeled by one additional model it can be concluded that this fault type is more difficult to model than the other lock-in-place faults.

**Table 4.5:** Description of model set II.

| Model # | Model description                           |
|---------|---------------------------------------------|
| 1       | Rudders stuck at upper limit (25 deg)       |
| 2       | Rudders stuck at lower limit ( $-25$ deg)   |
| 3       | Elevators stuck at lower limit ( $-23$ deg) |
| 4       | Ailerons stuck at upper limit (15 deg)      |
| 5       | Elevators stuck at upper limit (17 deg)     |
| 6       | Elevators stuck at $-6$ deg                 |
| 7       | Ailerons stuck at lower limit ( $-25$ deg)  |

Both model set I and model set II represent lock-in-place faults by local models with the corresponding control surfaces stuck at their limit position. This way of modeling lock-in-place faults has also been proposed by Liao et al. (2005), and therefore confirms the correctness of the OD-based model set design method.

## 4.4 Simulation Results

In this section the proposed fault modeling strategy is evaluated using the Boeing 747 benchmark model. Model set I is evaluated by linear simulation of the aircraft. In the linear simulation the full fault diagnosis capabilities of the proposed modeling strategy can be demonstrated. In order to also demonstrate the usefulness of the proposed method for the nonlinear aircraft model, in a second simulation the

full nonlinear Boeing 747 model is simulated. For this second simulation model set II is used. An exact identification of the fault is difficult to perform for the second simulation due to nonlinear behavior of the aircraft model. This behavior cannot be represented well by the designed model set because it is valid in a limited operating region near the operating condition at which the model set is designed. However, it will be shown that an isolation of faults is still very well possible.

#### 4.4.1 Linear Simulation

A fault scenario is simulated in which 5 faults are injected sequentially into the aircraft model. This scenario is described in Table 4.6. The simulation is performed in closed-loop and in continuous-time. A closed-loop simulation has been performed because the local models corresponding to the faults in Table 4.6 are unstable. For this purpose a linear quadratic regulator is used. The MM estimation algorithm used for this simulation study is the extended Kalman filter (EKF), which has been described in Table 2.3 on page 39. It uses the measurements of all states, except for  $\beta$  and  $\phi$ . Although these two variables are normally also measured in the Boeing 747, they are omitted to test the state estimation performance of the MM estimation algorithm combined with the designed model set. During the simulation, process noise and measurement noise are added to the system. The added process noise has a covariance of

$$Q_s = \text{diag}([10^{-8} \ 10^{-8} \ 10^{-8} \ 10^{-1} \ 10^{-4} \ 10^{-4} \ 10^{-4} \ 10^{-1}]), \quad (4.3)$$

where  $\text{diag}([a])$  denotes a diagonal matrix with vector  $a$  on its diagonal. The added measurement noise has a covariance of

$$R_s = \text{diag}([1.73 \cdot 10^{-4} \ 1.73 \cdot 10^{-4} \ 1.73 \cdot 10^{-4} \ 10^{-1} \ 1.73 \cdot 10^{-4} \ 1.73 \cdot 10^{-4} \ 1.73 \cdot 10^{-4} \ 10^{-1}]), \quad (4.4)$$

which corresponds to the realistic noise covariance of the sensors on a Boeing 747 (Breeman 2006).

**Table 4.6:** Simulated fault scenario for the linear simulation.

| Fault description             | Time interval |
|-------------------------------|---------------|
| Elevators stuck at 10 deg     | 15-20 s       |
| 50% fault in $\alpha$ -sensor | 35-40 s       |
| Rudders stuck at $-15$ deg    | 52.5-60 s     |
| Hydraulic system # 4 outage   | 75-80 s       |
| Engine separation             | 95 s-         |

The model weight estimation results, which are obtained by using Model set I are given in Figure 4.3. In this figure the time intervals in which a deviation is

to be expected from the nominal situation, i.e.  $\mu^{(8)} = 1$  and  $\mu^{(1)}, \dots, \mu^{(7)} = 0$ , are indicated by a shaded background. The first fault that is injected is an elevator lock-in-place fault at a deflection of 10 deg. This fault results in the model weight  $\mu^{(3)}$  becoming approximately 0.7. From this weight it can be deduced that the elevator is stuck at a position of  $0.7 \cdot 17 = 11.9$  deg, which closely approximates the injected fault of 10 deg. The second fault is a 50% fault in the  $\alpha$ -sensor. This fault is correctly identified since the model that corresponds to a total  $\alpha$ -sensor fault has a weight of approximately 0.5, while the nominal model also has a weight of approximately 0.5. The third fault is a lock-in-place fault of the rudders at a deflection of  $-15$  deg. This fault is described by the weight  $\mu^{(2)} \approx 0.55$  in the time interval 52.5 – 60 s. From this model weight it can be deduced that the rudder surfaces are locked at a position of  $0.55 \cdot -25 = -13.75$  deg, which again closely approximates the injected fault. The fourth fault is a total outage of hydraulic system 4. This fault is identified by  $\mu^{(7)}$ , which approaches a value of 1 during insertion of the fault. The last fault corresponds to the separation of two engines on the right wing. It can be seen that this fault is correctly identified at  $T = 95$  s by  $\mu^{(5)}$ , which becomes almost 1. Moreover, it can be seen that throughout the whole simulation  $\mu^{(8)}$  approaches 1 again when there are no faults injected. The two unmeasured states  $\beta$  and  $\phi$  are both estimated well. The variance accounted for (VAF) (defined in (3.53) on page 70) values for the estimates of both unmeasured states are above 90%. The VAF values for the estimates of the measured states are all above 99%, which indicates an almost perfect state reconstruction.

#### 4.4.2 Nonlinear Simulation

The simulation of the nonlinear model is initiated with the same initial condition as the linear simulation. Again, the simulation is performed in closed-loop. The controller used for the simulation is a subspace predictive controller, which will be described in Chapter 6. The simulations are performed under turbulent conditions in order to excite the aircraft, which is beneficial for fault diagnosis. The simulated fault scenario differs from that of the linear simulation in the fact that the faults are not injected sequentially. Instead, a number of nonlinear simulations is performed, each time starting at the same operating point. The operating point of the aircraft throughout the nonlinear simulation is held near the operating point at the start of the simulation by the controller. This is done deliberately to ensure that the linear models in the model set are able to adequately represent the nonlinear aircraft model.

In each simulation, the lock-in-place faults are injected at  $T = 20$  s. For each of the considered surfaces, the lock-in-place positions cover the whole deflection range with intervals of 0.5 deg. For example, the first simulation to assess the isolation performance of rudder lock-in-place faults starts at a lock-in-place fault at  $-25$  deg. In the subsequent simulations, the lock-in-place position is increased with 0.5 deg until the maximum deflection of 25 deg is reached. In this way, 101 different simulations are performed. For elevator lock-in-place faults 81 simulations are performed and for aileron lock-in-place faults 61 simulations are per-

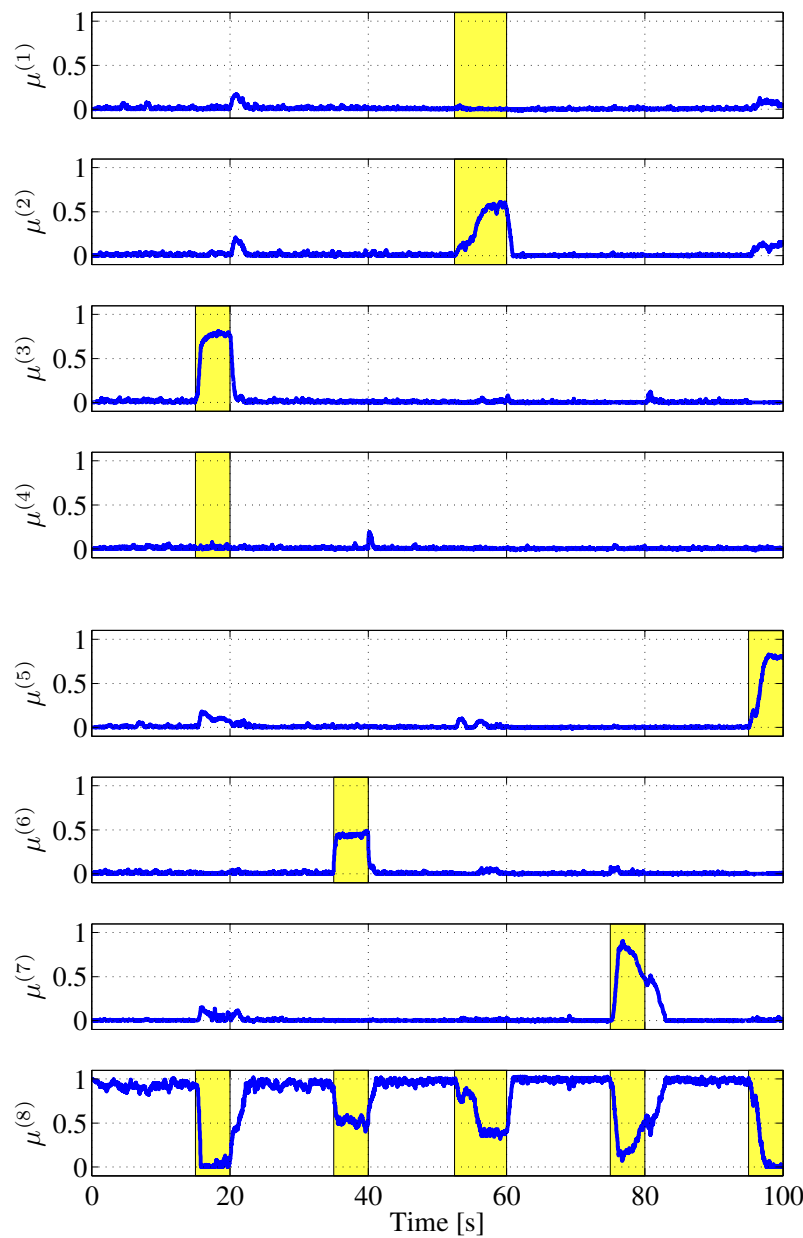


Figure 4.3: Estimated model weights.

formed. Note that the deflection position at which the lock-in-place faults are injected does not necessarily correspond to the actual position of the surfaces at the moment of fault injection. So, just after injection of the faults the surfaces move with their maximum operation rate to the lock-in-place position.

The model set used for the simulations is model set II, which has been described in Table 4.5. In addition to the local models in model set II, the nominal model is added to this model set. This has been done to simplify the isolation process. If the nominal model would not be present in the model set, it would not be immediately clear from the estimated model weights whether the system is operating in a nominal condition. In this case, the system condition must be reconstructed by using (3.52) on page 69. With the addition of the nominal model in the model set, however, this can be omitted. For the purpose of fault isolation from the estimated model weights, a decision logic is designed. For this logic, 4 system modes are defined. Each of these modes has a different fault signature. The system modes and their signatures are described in Table 4.7. In this table, “AND” denotes the logical “AND” operator,  $S_m$  denotes the system mode,  $T_f$  is the fixed fault threshold, which is chosen as  $T_f = 0.2$ , and the signatures  $F_0, \dots, F_4$  are boolean variables. The fault threshold is chosen such that false alarms, which are highly undesirable in aircraft, are completely prevented in the simulations that have been performed.

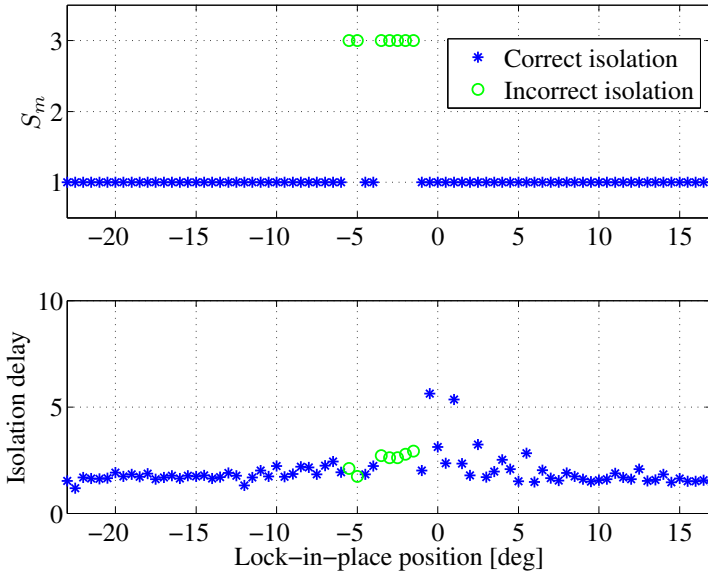
**Table 4.7:** Fault types and their signatures.

| $S_m$ | Description    | Signature                                                                           |
|-------|----------------|-------------------------------------------------------------------------------------|
| 0     | Nominal        | $F_0 =  1 - \mu^{(8)}  < T_f$                                                       |
| 1     | Elevator fault | $F_2 =  \mu^{(3)} + \mu^{(5)} + \mu^{(6)}  \geq T_f$ AND $ 1 - \mu^{(8)}  \geq T_f$ |
| 2     | Aileron fault  | $F_3 =  \mu^{(4)} + \mu^{(7)}  \geq T_f$ AND $ 1 - \mu^{(8)}  \geq T_f$             |
| 3     | Rudder fault   | $F_1 =  \mu^{(1)} + \mu^{(2)}  \geq T_f$ AND $ 1 - \mu^{(8)}  \geq T_f$             |

The isolation results for elevator lock-in-place faults are shown in Figure 4.4. It can be seen that these faults are generally isolated correctly and also quite quickly with a mean isolation delay of 1.9 s. There is also a small number of faults that is incorrectly isolated. The deflection positions at which faults are incorrectly isolated, are close to the positions at which the elevator surfaces are normally operating for the simulated flight conditions. This has the result that it is more difficult to notice a stuck surface. Since the aircraft dynamics for these specific faults can not be modeled only by the nominal model, an incorrect isolation is made.

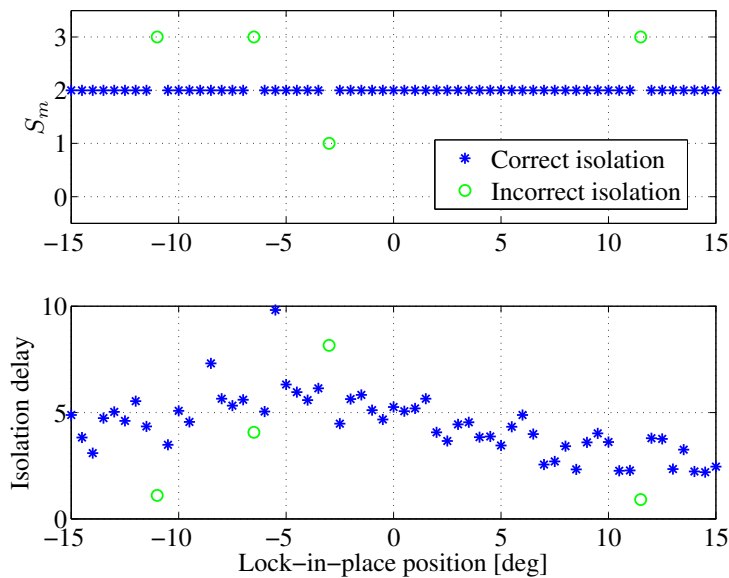
The isolation results for aileron lock-in-place faults are shown in Figure 4.5. It can be seen that generally the faults are correctly isolated. The mean isolation delay for aileron lock-in-place faults is 4.5 s. This delay can be accounted to the relatively slow dynamics of the Boeing 747 benchmark model. Since the isolation is directly depending on how the faults affect the aircraft dynamics, the isolation delay can also be expected not to be very fast. Furthermore, the threshold  $T_f$  is chosen quite large to prevent false alarms.

The isolation results for rudder lock-in-place faults are shown in Figure 4.6. It can be observed that although most of the time the faults are isolated correctly, there are also quite some faults that are missed. These faults are in the deflection range 0 – 13 deg. The reason for this is that the rudder surface is already in this deflection range at the moment of fault injection. This is also the reason why the isolation results are asymmetric with respect to the deflection range of the rudder. Another reason that adds to the missed alarms is the choice of the threshold  $T_f$ . The mean isolation delay for rudder lock-in-place faults is 3.8 s.

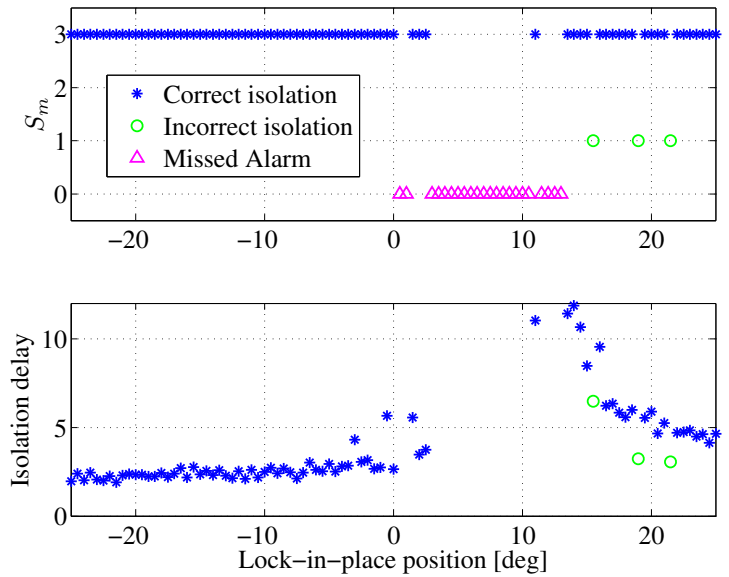


**Figure 4.4:** Isolation results for elevator lock-in-place. The asterisk (\*) denotes a correct isolation and the circle (o) denotes an incorrect isolation.

A summary of the isolation results is given in Table 4.8. In this table, the percentages for correct isolation, incorrect isolation, missed alarms, and false alarms are listed. It can be seen that no false alarms were encountered in any of the 243 simulations that have been performed. This, however, was a specific objective which was used for tuning of the threshold  $T_f$ . An apparent downside of this choice is that a larger isolation delay should be accepted and that faults are sometimes not isolated at all, resulting in missed alarms. This is one of the reasons that the percentage of missed alarms for rudder lock-in-place faults is quite high. Furthermore, the missed alarms are caused by the simulations being performed near the same operating point. Allowing the aircraft to operate in varying operating regions would generally lead to a heavier use of the control surfaces, which would lead to a smaller missed alarm rate.



**Figure 4.5:** Isolation results for aileron lock-in-place. The asterisk (\*) denotes a correct isolation and the circle (o) denotes an incorrect isolation.



**Figure 4.6:** Isolation results for rudder lock-in-place. The asterisk (\*) denotes a correct isolation, the circle (o) denotes an incorrect isolation, and the triangle (Δ) denotes a missed alarm.

**Table 4.8:** Summary of the isolation results.

|                     | Elevator<br>lock-in-place | Aileron<br>lock-in-place | Rudder<br>lock-in-place |
|---------------------|---------------------------|--------------------------|-------------------------|
| Correct isolation   | 91.4%                     | 93.4%                    | 75.2%                   |
| Incorrect isolation | 8.6%                      | 6.6%                     | 3%                      |
| Missed alarm        | 0%                        | 0%                       | 21.8%                   |
| False alarm         | 0%                        | 0%                       | 0%                      |

## 4.5 Conclusions

Two model sets have been designed for the Boeing 747 benchmark model with the method based on orthogonal decompositions, which has been described in Chapter 3. The first model set is designed with the purpose of showing the versatility of the used model set design method. It therefore models two types of lock-in-place faults, a range of multiplicative sensor faults, and two components faults that occurred during the “Bijlmerramp”. A second model set is designed that only includes those faults considered in the benchmark model that can reasonably be anticipated. These faults are lock-in-place faults in three groups of control surfaces, namely rudders, ailerons, and elevators. An important observation that could be made from the designed model sets is that lock-in-place faults are represented by local models that correspond to conditions in which the control surfaces are stuck at their limit position. This way of modeling lock-in-place faults has also been proposed by Liao et al. (2005), which confirms the correctness of the used model set design method. The two model sets have been evaluated in two separate simulation studies. In the first study a linear simulation of the aircraft is performed. In this study a number of faults including specific faults that occurred during the disastrous EL AL flight 1862 are correctly identified. In the second study, the isolation capabilities of the proposed method are demonstrated by means of simulation of the full nonlinear aircraft model.



## **Part II**

# **Fault-Tolerant Control and its Integration with Diagnosis**



# Persistency of Excitation for Subspace Predictive Control

---

Subspace predictive control is a control method that is characterized by the combination of a predictive control law with a subspace predictor. If the subspace predictor is updated on-line with new input-output data from the controlled system then subspace predictive control can be considered to be a data-driven adaptive control method. The property of data-driven adaptation is very suitable for fault-tolerant control. For the purpose of updating the predictor, the input-output data should be persistently exciting. In this chapter a method is presented that ensures persistency of excitation for subspace predictive control. This is done by adding a term to the cost function used by the predictive control law. This term is designed such that only the least excited directions of the input space are additionally excited. An advantage of the method is that the optimization problem that has to be solved for the predictive controller can still be solved by using quadratic programming. The proposed excitation method is evaluated in simulation on a linear parameter-varying model. The simulation results show both the ability of subspace predictive control to function as an adaptive control method and how the proposed excitation method is used to persistently excite the system.

---

## 5.1 Introduction

Subspace identification is a technique that can be used for identification of state-space models from input-output data. This technique has drawn considerable interest in the last two decades (Van Overschee and De Moor 1996; Verhaegen and Dewilde 1992), especially for linear time-invariant systems. A reason for this is the efficient way in which models are identified for systems of high order and with multiple inputs and outputs. Subspace identification can be used to form a subspace predictor for prediction of future outputs from past input-output data and a future input-sequence. This subspace predictor can be computed without

realization of the actual state-space models, which significantly reduces computational requirements. Favoreel and De Moor (1999) have combined the subspace predictor with model predictive control (Maciejowski 2002), resulting in a control algorithm that has been given the name subspace predictive control (SPC). In SPC, the output predicted by the subspace predictor is part of the cost function of the predictive controller. As a result of the subspace predictor being generated completely from input-output data, SPC is a data-driven control algorithm.

SPC has since its establishment by Favoreel and De Moor (1999) been modified to fit application-specific requirements by different researchers (Woodley et al. 2001; Kadali et al. 2003). For example, Kadali et al. (2003) have used a predictive control objective function with incremental inputs instead of the regular inputs. Furthermore, the same authors have added constraint handling and a feedforward term to the basic SPC algorithm by Favoreel and De Moor (1999). However, the implementation of Kadali et al. (2003) did not take advantage of the ability of SPC to adapt to changes in the system. If SPC is implemented such that the subspace predictor is updated on-line based on new input-output data, it can be considered to be an adaptive controller. Woodley et al. (2001) have implemented SPC in an adaptive manner and have also extended it to include  $\mathcal{H}_\infty$  cost functions. Furthermore, they demonstrated the adaptive implementation of SPC on a simple single-input single-output model. A suggestion made by Woodley et al. (2001) is that in the near future falling cost of computation would also make the algorithm suitable for more complex situations.

An important requirement for obtaining a subspace predictor that can accurately predict the system outputs, is that the available input-output data contains sufficient information on the system. This requirement can be met by ensuring that the input signals are persistently exciting the system. In this chapter, the notion of persistency of excitation (PE) is directly linked to non-singularity of a data matrix that contains input signals stacked in a particular order. This matrix will be explained later. The requirement for PE can conflict with the control objective, e.g. in the steady-state case. In such a case the control objective can be met by using constant input signals, which are not persistently exciting the system. If the input-output data in this case is used to update the subspace predictor, this can lead to a drastic degradation of the performance of the predictor. Therefore, in order to ensure PE, even in the steady-state case, the system should be excited more than strictly necessary for control. Instead of using a randomly generated excitation signal, such additional excitation can be derived by minimizing a control-oriented measure of model mismatch, as is done for example by Forssell and Ljung (2000); Bombois et al. (2006) for prediction error identification.

In the framework of simultaneous predictive control and system identification, methods for excitation have been proposed by formulating additional constraints for the optimization problem related to the predictive control law (Shouche et al. 1998; Aggelogiannaki and Sarimveis 2006). The drawback of these methods is that the additional constraints are non-convex. The result of this is that a non-convex optimization problem should be solved each time step to compute the control input. Standard constrained generalized predictive control problems re-

quire solving a quadratic program, which is a convex optimization problem (Maciejowski 2002). Since solving non-convex problems is significantly more involved than solving convex problems, the additional non-convex constraints are not desirable from a computational point of view.

The contribution of this chapter is twofold. First, this chapter shows that SPC can be used as a data-driven adaptive control method if the subspace predictor is updated on-line each time new input-output data becomes available. The property of SPC to adapt to changing system conditions is very desirable for fault-tolerant control, because it allows for adaptation to unanticipated fault conditions. Second, an excitation method is proposed for the SPC algorithm. This excitation method only considers the input directions that are least excited. These directions are then additionally excited. In this aspect the proposed excitation method is related to the methods described by Forssell and Ljung (2000); Bombois et al. (2006). The excitation method is integrated with SPC by adding an extra term to the cost function of the predictive control problem. The additional term allows the optimization problem to remain convex and quadratic. This problem can therefore still be solved by using quadratic programming. The two contributions of this chapter are demonstrated in simulation by means of a linear parameter-varying (LPV) model of a mass-spring-damper (MSD) system. This model has been introduced in Chapter 3.

The SPC algorithm presented in this chapter is derived for the deterministic case, which means that it is assumed that there is no process noise and measurement noise. This choice is motivated by one of the goals of this chapter, which is to display the effectiveness of the proposed excitation method. Adding process and measurement noise to the system also excites the system. Since this would make the actual contribution of the excitation method less visible, it is omitted. A derivation of the SPC algorithm in a stochastic setting, i.e. with process and measurement noise, can be found in Chapter 6.

The organization of this chapter is as follows. In Section 5.2, SPC theory is described for the deterministic case. Next, in Section 5.3 the proposed method for ensuring PE is explained. Section 5.4 contains the simulation results. Finally, concluding remarks are provided in Section 5.5.

## 5.2 Subspace Predictive Control

The SPC algorithm proposed by Favoreel and De Moor (1999) elegantly combines two theories, namely subspace identification and generalized predictive control. This combination results in a controller that does not require a model of the system in advance. Instead, based on input-output data, a subspace predictor can be identified that is used by the predictive controller. Because SPC does not require an explicit model of the system, it can be referred to as a “model-free” control method (Favoreel et al. 1999). However, this name is arguable because the subspace predictor can be considered to be a kind of model (Woodley et al. 2001).

In the following, the subspace predictor is derived first. Next, this predictor is integrated with a predictive control law.

### 5.2.1 Subspace Predictor

The general problem considered in linear subspace identification is to find system matrices  $A$ ,  $B$ ,  $C$ , and  $D$ , given measurements of the inputs  $u_k \in \mathbb{R}^m$  and outputs  $y_k \in \mathbb{R}^l$  of a linear time-invariant state-space system described by

$$x_{k+1} = Ax_k + Bu_k, \quad (5.1)$$

$$y_k = Cx_k + Du_k, \quad (5.2)$$

where  $x_k \in \mathbb{R}^n$  is the state of the system. Matrix input-output relations are commonly used in subspace identification. These relations can be obtained by a recursive substitution of (5.1)-(5.2) and they are given by

$$Y_p = \mathcal{O}_j X_p + \mathcal{T}_j U_p, \quad (5.3)$$

$$Y_f = \mathcal{O}_j X_f + \mathcal{T}_j U_f. \quad (5.4)$$

Let the measurements of the inputs and outputs  $u_k$  and  $y_k$  be given for  $k \in \{0, 1, \dots, 2M + j - 2\}$ , then the Hankel matrices for the output are constructed as

$$\begin{aligned} Y_p &= \begin{bmatrix} y_0 & y_1 & \cdots & y_{j-1} \\ y_1 & y_2 & \cdots & y_j \\ \vdots & \vdots & \ddots & \vdots \\ y_{M-1} & y_M & \cdots & y_{M+j-2} \end{bmatrix}, \\ Y_f &= \begin{bmatrix} y_M & y_{M+1} & \cdots & y_{M+j-1} \\ y_{M+1} & y_{M+2} & \cdots & y_{M+j} \\ \vdots & \vdots & \ddots & \vdots \\ y_{2M-1} & y_{2M} & \cdots & y_{2M+j-2} \end{bmatrix}, \end{aligned} \quad (5.5)$$

and the Hankel matrices for the input ( $U_p$  and  $U_f$ ) are constructed in a similar fashion. The subscripts  $p$  and  $f$  denote “past” and “future”, respectively. The matrices  $X_p$  and  $X_f$  are defined as

$$X_p = [x_0 \ x_1 \ \cdots \ x_{j-1}], \quad (5.6)$$

$$X_f = [x_M \ x_{M+1} \ \cdots \ x_{M+j-1}]. \quad (5.7)$$

The parameter  $M$ , which denotes the number of block rows of the Hankel matrices is typically chosen much smaller than the number of columns  $j$ . The observability

matrix  $\mathcal{O}_j$  and the block Toeplitz matrix  $\mathcal{T}_j$  are given by

$$\mathcal{O}_j = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{j-1} \end{bmatrix}, \quad \mathcal{T}_j = \begin{bmatrix} D & 0 & \dots & 0 \\ CB & D & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{j-2}B & CA^{j-3}B & \dots & D \end{bmatrix}. \quad (5.8)$$

A relation between  $X_f$ ,  $X_p$ , and  $U_p$  can be formulated as

$$X_f = A^j X_p + \mathcal{K}_j U_p, \quad (5.9)$$

where  $\mathcal{K}_j = [A^{j-1}B \ A^{j-2}B \ \dots \ B]$ . The objective of the subspace predictor is to provide a prediction of future outputs (i.e.  $Y_f$ ) given the past inputs and outputs (i.e.  $Y_p$  and  $U_p$ ) and a future control input sequence (i.e.  $U_f$ ). Such a prediction can be obtained by manipulation of the matrix relations (5.3), (5.4), and (5.9). From (5.3) the following relation for  $X_p$  can be derived

$$X_p = \mathcal{O}_j^\dagger (Y_p - \mathcal{T}_j U_p), \quad (5.10)$$

where  $\mathcal{O}_j^\dagger$  denotes the pseudo-inverse of  $\mathcal{O}_j$ . Substitution of (5.10) into (5.9) results in

$$X_f = [A^j \mathcal{O}_j^\dagger \quad \mathcal{K}_j - A^j \mathcal{O}_j^\dagger \mathcal{T}_j] \begin{bmatrix} Y_p \\ U_p \end{bmatrix}. \quad (5.11)$$

Next, substitution of (5.11) into (5.4) results in the desired relation

$$Y_f = \mathcal{O}_j [A^j \mathcal{O}_j^\dagger \quad \mathcal{K}_j - A^j \mathcal{O}_j^\dagger \mathcal{T}_j] \begin{bmatrix} Y_p \\ U_p \end{bmatrix} + \mathcal{T}_j U_f = L_p V_p + L_f U_f, \quad (5.12)$$

where  $V_p = [Y_p^T \ U_p^T]^T$  and  $L_p$  and  $L_f$  are the predictor matrices. Given the data matrices  $V_p$ ,  $Y_f$ , and  $U_f$ , these predictor matrices can be estimated by solving the following least squares problem

$$\min_{L_p, L_f} \left\| Y_f - [L_p \ L_f] \begin{bmatrix} V_p \\ U_f \end{bmatrix} \right\|_F^2, \quad (5.13)$$

where  $\|\cdot\|_F$  denotes the Frobenius norm of a matrix. The least squares problem defined in (5.13) can be solved efficiently by computing the RQ-decomposition (Van Overschee and De Moor 1996)

$$\begin{bmatrix} V_p \\ U_f \\ Y_f \end{bmatrix} = \overbrace{\begin{bmatrix} R_{11} & 0 & 0 \\ R_{21} & R_{22} & 0 \\ R_{31} & R_{32} & R_{33} \end{bmatrix}}^R \begin{bmatrix} Q_1^T \\ Q_2^T \\ Q_3^T \end{bmatrix}, \quad (5.14)$$

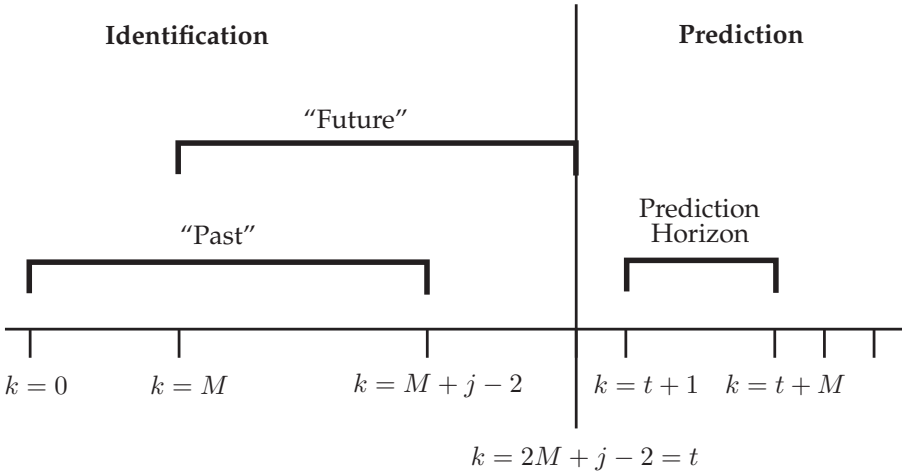
and subsequently computing

$$L = [R_{31} \ R_{32}] \begin{bmatrix} R_{11} & 0 \\ R_{21} & R_{22} \end{bmatrix}^{-1} = [L_p \ L_f], \quad (5.15)$$

where  $L_p \in \mathbb{R}^{Ml \times (m+l)M}$  and  $L_f \in \mathbb{R}^{Ml \times Mm}$  are the predictor matrices. These matrices can subsequently be used to form the subspace predictor

$$\hat{y}_f = L_p v_p + L_f u_f, \quad (5.16)$$

where  $v_p$  denotes  $[y_p^T \ u_p^T]^T$  with  $y_p = [y_{t-M+1}^T \ y_{t-M+2}^T \ \cdots \ y_t^T]^T$ ,  $u_p = [u_{t-M+1}^T \ u_{t-M+2}^T \ \cdots \ u_t^T]^T$ , and  $t$  denotes the current time step. The future input vector is defined as  $u_f = [u_{t+1}^T \ u_{t+2}^T \ \cdots \ u_{t+M}^T]^T$ . Note that (5.16) makes use of the fact that matrix relation (5.12) also holds for the individual vectors of  $V_p$ ,  $Y_f$ , and  $U_f$ . In order to further clarify the derivation of the subspace predictor, the time line related to it is shown in Figure 5.1. In this figure a distinction is made between an identification part and a prediction part, which takes place in the actual future. The identification part consist of two time intervals, one which is named “Past” and one which is named “Future”. The data from these two intervals is used to create the “past” and “future” Hankel matrices (see (5.5)). Note that although the name of the “Future” time interval in the identification part might indicate otherwise, it contains data from the past. In the identification part a subspace predictor is derived, which is used to make a prediction in the actual future. The time interval over which the prediction is made is indicated in the right part of Figure 5.1 by the prediction horizon.



**Figure 5.1:** Time line for the subspace predictor.

New input-output data becomes available at each time step. This new data should be used to update the predictor matrices  $L_p$  and  $L_f$ . This can be done by computing a new RQ-decomposition at each time step using the new input-output



data. Subsequently the updated predictor matrices can be derived by using (5.15) and the new RQ-decomposition. In this way the predictor is based on a batch of data from a fixed-size time window that lags behind one time step with respect to the actual time. Because of its operating principle this updating scheme is referred to as a receding horizon scheme (Woodley et al. 2001).

### 5.2.2 Subspace Predictor Integrated with a Predictive Control Law

The predictive control problem can be formulated as follows. Given a future reference output  $r_f = [r_{t+1} \ r_{t+2} \ \dots \ r_{t+N_p}]$  and a prediction of the outputs  $\hat{y}_f = [\hat{y}_{t+1} \ \hat{y}_{t+2} \ \dots \ \hat{y}_{t+N_p}]$ , find an input sequence  $u_f$  such that the following quadratic cost function is minimized

$$\begin{aligned} J &= \sum_{k=1}^{N_p} (\hat{y}_{t+k} - r_{t+k})^T Q_c (\hat{y}_{t+k} - r_{t+k}) + \sum_{k=1}^{N_c} u_{t+k}^T R_c u_{t+k}, \\ &= (\hat{y}_f - r_f)^T Q_a (\hat{y}_f - r_f) + u_f^T R_a u_f, \end{aligned} \quad (5.17)$$

where  $N_p$  is the prediction horizon,  $N_c$  is the control horizon,  $Q_c \in \mathbb{R}^{l \times l}$ , and  $R_c \in \mathbb{R}^{m \times m}$  are the positive semi-definite weighting matrices for the tracking error and the input effort, respectively. The matrices  $Q_a \in \mathbb{R}^{N_p l \times N_p l}$  and  $R_a \in \mathbb{R}^{N_c m \times N_c m}$  are formed from  $Q_c$  and  $R_c$  as follows

$$Q_a = \begin{bmatrix} Q_c & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & Q_c \end{bmatrix}, \quad R_a = \begin{bmatrix} R_c & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & R_c \end{bmatrix}. \quad (5.18)$$

The cost function used by Favoreel and De Moor (1999) is equal to (5.17). However, this cost function does not penalize the rate with which the input signal changes. Since this is desirable for practical systems such as aircraft, the objective function has been augmented with an additional term. This term consists of the incremental inputs  $\Delta u_f$ , where  $\Delta = (1 - z^{-1})$  and  $z^{-1}$  is the back-shift operator of one time step. The resulting cost function is of the form

$$J = (\hat{y}_f - r_f)^T Q_a (\hat{y}_f - r_f) + u_f^T R_a u_f + \Delta u_f^T R_a^\Delta \Delta u_f, \quad (5.19)$$

where  $R_a^\Delta$  has matrices  $R_c^\Delta$  on its diagonal and is constructed in a similar manner as  $R_a$ . The main idea of SPC is to use the subspace predictor derived in (5.16) to compute  $\hat{y}_f$  rather than to use a mathematical model as is common in model predictive control (MPC) (Maciejowski 2002). Before the subspace predictor from (5.16) can be used in (5.19), it has to be modified such that the control horizon is

properly incorporated. This can be done in the following way

$$\hat{y}_f = L_p v_p + L_f \overbrace{\begin{bmatrix} I_m & 0 & \cdots & 0 \\ 0 & I_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I_m \\ 0 & \cdots & 0 & I_m \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & I_m \end{bmatrix}}^E u_f, \quad (5.20)$$

where  $I_m \in \mathbb{R}^{m \times m}$  denotes an identity matrix. Next, the incremental input  $\Delta u_f$  has to be formulated as a function of  $u_f$  to ensure that  $u_f$  is the only optimization variable in  $J$ . This can be done as follows

$$\Delta u_f = \overbrace{\begin{bmatrix} I_m & 0 & 0 & \cdots & 0 \\ -I_m & I_m & 0 & & 0 \\ 0 & -I_m & I_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -I_m & I_m \end{bmatrix}}^{S_\Delta} u_f - \overbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & I_m \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}}^{S_v} v_p. \quad (5.21)$$

Now, (5.20) and (5.21) can be substituted into (5.19), which results in the following relation

$$\begin{aligned} J = & \underline{v_p^T L_p^T Q_a L_p v_p} + v_p^T L_p^T Q_a L_f E u_f - \underline{v_p^T L_p^T Q_a r_f} + u_f^T E^T L_f^T Q_a L_p v_p \\ & + u_f^T E^T L_f^T Q_a L_f E u_f - u_f^T E^T L_f^T Q_a r_f - \underline{r_f^T Q_a L_p v_p} - r_f^T Q_a L_f E u_f \\ & + \underline{r_f^T Q_a r_f} + u_f^T R_a u_f + u_f^T S_\Delta^T R_a^\Delta S_\Delta u_f - u_f^T S_\Delta^T R_a^\Delta S_v v_p \\ & - \underline{v_p^T S_v^T R_a^\Delta S_\Delta u_f} + \underline{v_p^T S_v^T R_a^\Delta S_v v_p}, \end{aligned} \quad (5.22)$$

where the underlined terms do not depend on  $u_f$  and can therefore be discarded from the cost function. The remaining terms can be written in a compact form as follows

$$J(u_f) = u_f^T H u_f + c^T u_f, \quad (5.23)$$

where

$$H = E^T L_f^T Q_a L_f E + S_\Delta^T R_a^\Delta S_\Delta + R_a, \quad (5.24)$$

$$c^T = 2 (v_p^T L_p^T Q_a L_f E - r_f^T Q_a L_f E - v_p^T S_v^T R_a^\Delta S_\Delta). \quad (5.25)$$

The SPC algorithm can subsequently be derived by minimizing  $J(u_f)$  with respect

to  $u_f$ . This can be done by computing the value of  $u_f$  for which it holds that

$$\frac{\partial J(u_f)}{\partial u_f} = u_f^T (H^T + H) + c^T = 0. \quad (5.26)$$

This computation results in the control law

$$u_f = -c(H^T + H)^{-1}. \quad (5.27)$$

The control law formulated in (5.27) does not take into account any constraints on the inputs, input rates, and outputs that exist in practical systems. Therefore, this control law must be modified such that it does take these constraints explicitly into account. The constraints in question are given by the following set of inequalities

$$\Delta U_{\min} \leq \Delta u_f \leq \Delta U_{\max}, \quad (5.28)$$

$$U_{\min} \leq u_f \leq U_{\max}, \quad (5.29)$$

$$Y_{\min} \leq \hat{y}_f \leq Y_{\max}, \quad (5.30)$$

where  $U_{\min} = [u_{\min}^T \cdots u_{\min}^T]^T$ ,  $\Delta U_{\min} = [\Delta u_{\min}^T \cdots \Delta u_{\min}^T]^T$ ,  $Y_{\min} = [y_{\min}^T \cdots y_{\min}^T]^T$ , and the same notation also holds for the parameters with subscript max. Again, it must be ensured that  $u_f$  is the only optimization variable to appear in these constraints. This can be done by substituting (5.20) and (5.21) into the inequality constraints, which then results in

$$\Delta U_{\min} + S_v v_p \leq S^\Delta u_f \leq \Delta U_{\max} + S_v v_p, \quad (5.31)$$

$$U_{\min} \leq u_f \leq U_{\max}, \quad (5.32)$$

$$Y_{\min} - L_p v_p \leq L_f E u_f \leq Y_{\max} - L_p v_p. \quad (5.33)$$

Now, the inequality constraints can be grouped into one inequality constraint of the form

$$A_{\text{ineq}} u_f \leq b_{\text{ineq}}, \quad (5.34)$$

with

$$A_{\text{ineq}} = \begin{bmatrix} I_{N_{cm}} & -I_{N_{cm}} & S_\Delta^T & -S_\Delta^T (L_f E)^T & -(L_f E)^T \end{bmatrix}^T, \quad (5.35)$$

$$b_{\text{ineq}} = \begin{bmatrix} U_{\max}^T & -U_{\min}^T & (\Delta U_{\max} + S_v v_p)^T & (-\Delta U_{\min} - S_v v_p)^T & (Y_{\max} - L_p v_p)^T & (-Y_{\min} + L_p v_p)^T \end{bmatrix}^T. \quad (5.36)$$

Next, the predictive control law can be formulated as a solution for the following quadratic programming (QP) problem at each time step

$$\begin{aligned} \min_{u_f} \quad & J(u_f) \\ \text{s.t.} \quad & A_{\text{ineq}} u_f \leq b_{\text{ineq}}. \end{aligned} \quad (5.37)$$

Solving a QP problem with linear constraints is a convex problem provided that matrix  $H$ , which appears in the cost function  $J(u_f)$ , is positive semi-definite. This is the case for  $J(u_f)$ . Several efficient methods exist for solving convex QP problems (Maciejowski 2002). Note that at each time step only the first input vector from  $u_f$ , i.e.  $u_{t+1}$ , is used for control because of the receding horizon operating principle of the predictive controller. This principle has been illustrated earlier in Figure 1.4 on page 14.

### 5.3 Persistency of Excitation

The control law defined by (5.37) can result in a constant control signal (i.e.  $\Delta u_f = 0$ ), for example during steady-state situations. The result of this is that the matrix to be inverted in (5.15) becomes singular. Therefore, a method is proposed that prevents this by ensuring PE. A formal definition for PE is the following

**Definition 5.1** (Verhaegen and Verdult 2007) *The sequence  $u_k$ ,  $k = 0, 1, 2, \dots, N_k$  is persistently exciting of order  $h$  if and only if there exists an integer  $N_k$  such that the matrix*

$$U_{0,h,N_k} = \begin{bmatrix} u_0 & u_1 & \dots & u_{N_k-1} \\ u_1 & u_2 & \dots & u_{N_k} \\ \vdots & \vdots & & \vdots \\ u_h & u_{h+1} & \dots & u_{N_k+h-1} \end{bmatrix} \quad (5.38)$$

*has full rank  $h$ .*

According to this definition, PE can be achieved by manipulating the input such that the Hankel matrix containing the inputs ( $U_{0,h,N_k}$ ) over a time window does not become (close to) singular. Note that the matrix  $U_{0,h,N_k}$  is constructed in the same way as  $U_f$  and  $U_p$ . Since  $U_f$  is the data matrix that is most directly manipulated by the SPC algorithm, this matrix is used as the target matrix to be manipulated by the excitation algorithm. The corresponding part of  $U_f$  in the  $R$ -matrix defined in (5.14) is  $[R_{21} \ R_{22}]$ . So, by manipulating  $U_f$ , the part of the  $R$ -matrix that must be prevented from becoming singular is also manipulated. Note that by manipulating  $[R_{21} \ R_{22}]$ , matrix  $R_{11}$  is also (indirectly) manipulated.

Now that it is determined that  $[R_{21} \ R_{22}]$  is the part of the  $R$ -matrix that is to be manipulated, it should be determined *how* it must be manipulated. A sensible approach to do this, is to additionally excite the least excited directions of  $U_f$ , i.e. those directions in the input space that actually need to be excited. The least excited directions can be determined by performing a singular value decomposition (SVD) on  $[R_{21} \ R_{22}]$ . Computing an SVD of  $[R_{21} \ R_{22}]$  is equivalent to computing

an SVD of  $U_f$ . The SVD of  $[R_{21} \ R_{22}]$  has the following form

$$[R_{21} \ R_{22}] = [w_1 \ w_2 \ \cdots \ w_{Mm-1} \ w_{Mm}] \cdot \begin{bmatrix} s_1 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & s_2 & \ddots & \ddots & \vdots & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & s_{Mm-1} & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & s_{Mm} & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{Mm-1} \\ v_{Mm} \\ \vdots \\ v_j \end{bmatrix}^T, \quad (5.39)$$

where  $w_i \in \mathbb{R}^{Mm}$  and  $v_i \in \mathbb{R}^{1 \times j}$ . The least excited direction of  $[R_{21} \ R_{22}]$  (and hence  $U_f$ ) is  $w_{Mm}$  and the second least excited direction is  $w_{Mm-1}$ , etc.

### 5.3.1 Computing the SVD Using Inverse Iterations

Computing the complete SVD of matrix  $[R_{21} \ R_{22}]$  becomes a computationally burdensome task if the dimensions of this matrix are large. The need to compute such an SVD at each time step of the control algorithm would make it unsuitable for on-line implementation. Moreover, computation of the complete SVD is not necessary since only the least excited directions are sought after. Therefore, only these least excited directions are computed efficiently by using the inverse iteration algorithm (Chan 1984) that only computes that part of an SVD that corresponds to the smallest nonzero singular value. The inverse iteration algorithm is described as follows. Let  $w_{sv}$  and  $v_{sv}$  be vectors from the SVD of a matrix  $\Theta$  that correspond to the smallest singular value  $\sigma_{sv}$ . In the case of (5.39), this would mean that  $w_{sv} = w_{Mm}$ ,  $v_{sv} = v_{Mm}^T$ , and  $\sigma_{sv} = s_{Mm}$ . The inverse iteration algorithm starts with an initial guess for  $w_{sv}$  at  $k = 0$ , followed by an iteration that should be performed until convergence:

1.  $\Theta \tilde{v}_{sv}(k+1) = w_{sv}(k)$ .
2.  $v_{sv}(k+1) = \tilde{v}_{sv}(k+1) / \|\tilde{v}_{sv}(k+1)\|_2$ .
3.  $\Theta^T \tilde{w}_{sv}(k+1) = v_{sv}(k+1)$ .
4.  $w_{sv}(k+1) = \tilde{w}_{sv}(k+1) / \|\tilde{w}_{sv}(k+1)\|_2$ .

After convergence of this iteration, the smallest singular value can be computed as  $\sigma_{sv} = 1 / \|\tilde{v}_{sv}\|_2$ . In order to run the inverse iteration algorithm the pseudo-inverse of matrix  $\Theta$  must be computed since it is required in steps 1 and 3. For example, step 1 of the iteration is solved by  $\tilde{v}_{sv}(k+1) = \Theta^\dagger w_{sv}(k)$ . It suffices to compute the pseudo-inverse only once, since  $(\Theta^T)^\dagger = (\Theta^\dagger)^T$ .

The described inverse iteration sequence computes only one direction that is least excited. It can be desirable to compute more than one direction. This can

also be done very efficiently using inverse iterations. For this purpose the influence of the least excited direction is removed to compute the second least excited direction. Since not  $\Theta$  itself, but  $\Theta^\dagger$ , is used in the inverse iteration algorithm, the influence of the least excited direction is removed directly from  $\Theta^\dagger$  in the following way

$$\tilde{\Theta}^\dagger = \Theta^\dagger - v_{sv} w_{sv}^T / \sigma_{sv}. \quad (5.40)$$

Next, the second least excited direction can be computed by applying the inverse iteration algorithm using  $\tilde{\Theta}^\dagger$ . This second sequence of iterations would then result in  $w_{Mm-1}$ ,  $v_{Mm-1}$ , and  $s_{Mm-1}$  from (5.39). The same procedure can be repeated several times to obtain more least excited directions.

### 5.3.2 Excitation Condition Integrated with the SPC Cost Function

Once one or more non-persistently excited directions are obtained, the cost function used in the optimization problem defined in (5.37) should be modified such that the system is additionally excited in the non-persistently excited directions. For this problem the following optimization is introduced

$$\min_{u_f} \left\| \rho \begin{bmatrix} \overbrace{w_{Mm} \\ w_{Mm-1} \\ \vdots \\ w_{Mm-N_{PE}+1}}^{\mathcal{P}} \end{bmatrix} - \begin{bmatrix} \overbrace{I \\ I \\ \vdots \\ I}^{\mathcal{I}} \end{bmatrix} u_f \right\|_2^2, \quad (5.41)$$

where  $\rho$  denotes the excitation level and  $N_{PE}$  denotes the number of least excited directions that should be additionally excited. The objective of the optimization problem posed in (5.41) is to get  $u_f$  as close to the non-persistently excited directions as possible. Since by definition of the SVD, the directions  $w_i$  are normalized vectors, the norm of vector  $[w_{Mm}^T \ w_{Mm-1}^T \ \cdots \ w_{Mm-N_{PE}+1}^T]^T$ , which is  $\sqrt{N_{PE}}$ , might make  $\mathcal{P}$  much smaller than  $\mathcal{I}$ . Therefore, the parameter  $\rho$  is introduced to ensure that  $\mathcal{P}$  and  $\mathcal{I}$  have the same order of magnitude. Furthermore,  $\rho$  can be used to determine the excitation level. The higher  $\rho$  is chosen, the higher the excitation level.

Evaluation of the cost function from (5.41) results in

$$J_{exc}(u_f) = u_f^T \mathcal{I}^T \mathcal{I} u_f - 2\rho \mathcal{P}^T \mathcal{I} u_f. \quad (5.42)$$

This function can then be added to the cost function from (5.37). Before this is done, a multiplication factor  $\tau \geq 0$  is introduced that weighs  $J_{exc}(u_f)$  before it is added. This factor  $\tau$  allows  $\tau J_{exc}(u_f)$  to be in the same order of magnitude as the other terms in the cost function. It can be considered to be the same kind of tuning parameter as  $Q_a$ ,  $R_a$ , and  $R_a^\Delta$ , relative to which it should be chosen. After

addition of  $\tau J_{\text{exc}}(u_f)$ , the modified  $H$  and  $c$  matrices (defined in (5.24) and (5.25)) become

$$H_{\text{exc}} = H + \tau \mathcal{I}^T \mathcal{I}, \quad (5.43)$$

$$c_{\text{exc}}^T = c^T - 2\tau \mathcal{P}^T \mathcal{I}, \quad (5.44)$$

and  $A_{\text{ineq}}$  and  $b_{\text{ineq}}$  remain unaltered. Note that  $H_{\text{exc}}$  is still positive semi-definite since it consists of the sum of two positive semi-definite matrices. It is therefore apparent that the resulting optimization problem can still be solved by convex quadratic programming. This is an important advantage of the proposed excitation condition since it allows an efficient computational implementation.

## 5.4 Simulation Results

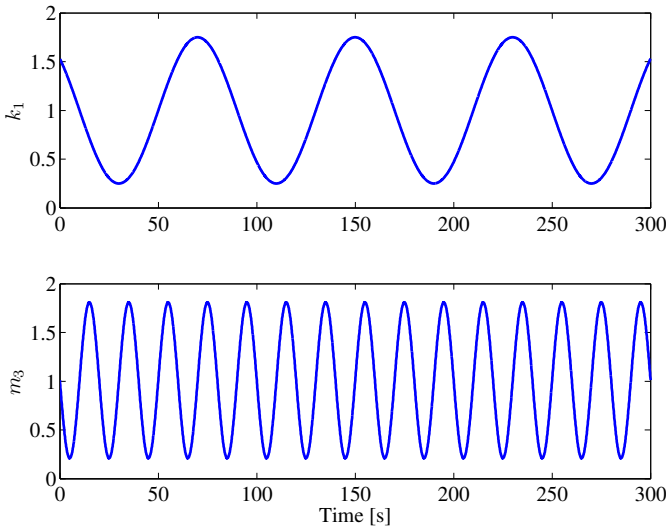
The goal of this section is twofold. First, the ability of SPC to adapt to a varying system is demonstrated. Second, the proposed method for ensuring PE is evaluated. The two goals are addressed separately using two different simulations. The reason for this is that the PE condition is best demonstrated in a steady-state simulation, while the simulation used to demonstrate the adaptation ability of SPC does not contain any steady-state situations. Both simulations use the LPV model of the MSD system that has been introduced in Chapter 3.

### 5.4.1 SPC as an Adaptive Controller

The LPV-MSD model that has been introduced in Chapter 3 consists of four varying parameters, namely the parameters  $k_1$  and  $m_3$ , and two fault parameters  $f_1$  and  $f_2$ . In this section, only  $k_1$  and  $m_3$  are used as varying parameters. The fault parameters  $f_1$  and  $f_2$  are both chosen to have a constant value of 1, which corresponds to the nominal (non-faulty) value. In order to assess the ability of SPC to adapt to changing parameters, a simulation of the LPV-MSD model has been performed. In this simulation, the parameters  $k_1$  and  $m_3$  vary according to the signal trajectories shown in Figure 5.2. In addition to these parameter variations, a more severe change of the model dynamics is injected into the simulation at  $T = 200$  s. This change of the model dynamics involves a time delay of 0.25 s in both input channels and is inspired from Fekri et al. (2006b), in which it was also used to assess the quality of an adaptive controller. The time delay can be interpreted as a fault. The objective of the SPC algorithm in the simulation is to track a constant reference signal for the variables  $x_1$  and  $x_3$ , which are measured by  $y_1$  and  $y_3$  (see Figure 3.4 on page 65 for a graphical representation of these variables). The SPC algorithm is simulated with a frequency of 10 Hz, which means that the subspace predictor is also updated with this frequency. The relevant tuning parameters for the SPC algorithm are chosen as:  $M = N_p = 20$  and  $N_c = 5$ . The value for  $M$  is chosen according the rule that it should be chosen two or three times larger than the expected system order (Van Overschee and De Moor 1996). For comparison

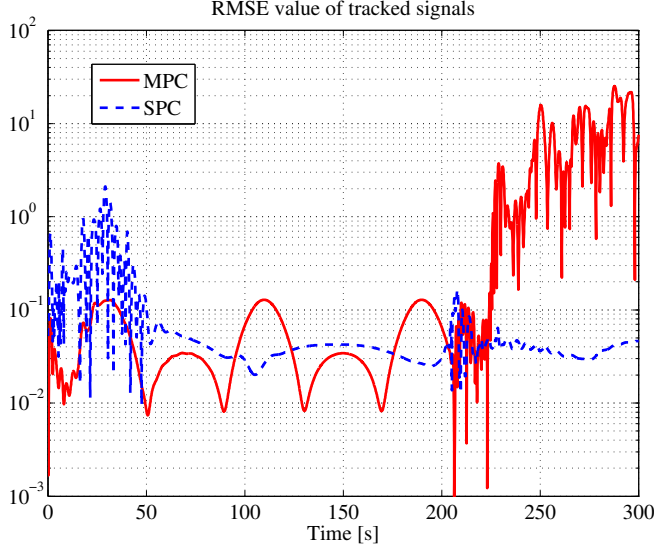
purposes, a conventional MPC algorithm is implemented with the same control objective as the SPC algorithm. This MPC algorithm is based on a model of the system with parameter values  $k_1 = 1$  and  $m_3 = 1$ . Note that these two values correspond exactly to the centers of the intervals in which they vary. Furthermore, the control parameters for the MPC algorithm are chosen similar to the parameters of the SPC algorithm.

The result of the simulation is shown in Figure 5.3. In this figure the root mean square error (RMSE) values for the SPC algorithm and MPC algorithm are depicted. These values are computed for the two tracked signals, i.e.  $x_1$  and  $x_3$ . It can be seen in the figure that the RMSE value for the SPC algorithm is rather large at the start of the simulation. The reason for this is that the SPC algorithm requires data to adjust to the controlled system. In the time interval from 50 s to 200 s, the SPC algorithm has settled and a good comparison can be made with the MPC algorithm. An important observation that can be made is that the SPC algorithm manages to maintain a relatively steady RMSE in this interval as opposed to the MPC algorithm. This is not surprising since the MPC algorithm is designed for the operating point with  $k_1 = 1$  and  $m_3 = 1$ . So, when the simulated system moves away from this operating point, the performance of the MPC algorithm tends to deteriorate as can be seen in the figure. A much larger deterioration of the performance of the MPC algorithm can be recognized after the injection of the delay in the two input signals at  $T = 200$  s. The SPC algorithm, on the other hand, manages to maintain a steady performance after a short transient period, in which data is gathered to adapt to the changed system conditions. This result clearly shows the ability of SPC to adapt to changes in the controlled system.



**Figure 5.2:** Simulated values for  $k_1$  and  $m_3$ .





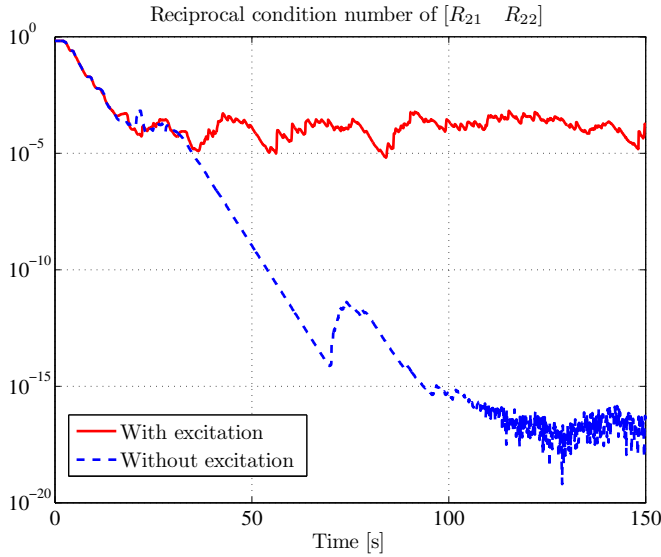
**Figure 5.3:** RMSE values of the tracked signals using MPC and SPC.

#### 5.4.2 Persistency of Excitation for SPC

In this section again the LPV-MSD model is simulated. However, in this simulation no parameters are varied. The values for the varying parameters are chosen constant as  $k_1 = 1$  and  $m_3 = 1$ . This is done to display how the proposed excitation method can be used to persistently excite the system in steady-state situations, which is where it is needed most. The control objective in this simulation is, similarly to the previous section, to track a constant reference signal for the variables  $x_1$  and  $x_3$ . The described simulation is performed for the cases with and without additional excitation. The tuning parameters related to the SPC algorithm without excitation are chosen similarly to the previous section. In case of additional excitation, the two least excited directions are computed and additionally excited. Tuning of the parameters  $\rho$  and  $\tau$  is performed empirically and is very much dependent on the parameters of the controlled system. The parameter  $\rho$  is chosen such that the vector containing the least excited directions and the vector containing the input signals (see (5.41)) have a norm that is in the same order of magnitude. The parameter  $\tau$  is chosen relative to the other tuning parameters in the predictive control cost function, i.e.  $Q_a$ ,  $R_a$ , and  $R_a^\Delta$ , such that a desired level of excitation is achieved.

In order to evaluate how the proposed method affects the excitation of the system, the reciprocal condition number of  $[R_{21} \ R_{22}]$  is analyzed. For the two cases, this condition number is depicted in Figure 5.4. A low reciprocal condition number indicates a (nearly) singular matrix and therefore a low excitation level. In Figure 5.4 it can be clearly seen that the condition number for the case without additional excitation becomes very small. More specifically, it approaches

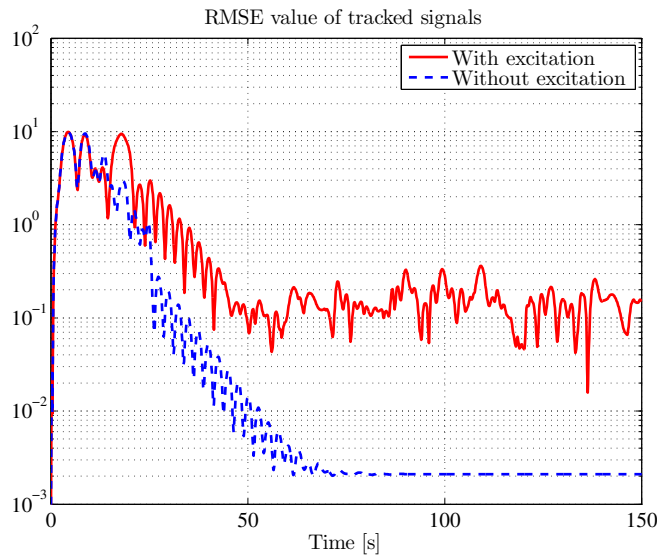
the smallest number that can be represented on the machine used for the simulation. This indicates that the considered matrix becomes singular. On the other hand, the condition number for the case with excitation remains at a steady level of approximately  $10^{-4}$  throughout the whole simulation. The price to pay for this excitation is a lower control performance. This is illustrated in Figure 5.5. In this figure, the RMSE values of the tracked signals are shown for the cases with and without additional excitation. It can be seen that the RMSE for SPC with excitation is larger than for SPC without excitation. However, it should be noted that the RMSE for SPC with excitation is in the order of magnitude of  $10^{-1}$ , which can still be considered acceptable.



**Figure 5.4:** Comparison of the reciprocal condition number of  $[R_{21} \ R_{22}]$  for the cases with and without additional excitation.

## 5.5 Conclusions

In this chapter subspace predictive control has been introduced. This control method combines a predictive control law with a subspace predictor. An on-line update of the predictor using new input-output data gives this control method the ability to adapt to changing system conditions, which is very desirable for fault-tolerant control. An issue concerned with updating of the subspace predictor is that the input-output data should be persistently exciting. For this purpose, a method has been proposed to ensure persistency of excitation. The notion of persistency of excitation is related to non-singularity of a Hankel matrix containing input signals. In order to ensure persistency of excitation an additional term



**Figure 5.5:** RMSE values of the tracked signals for the cases with and without additional excitation.

is added to the cost function used by the subspace predictive controller. This additional term ensures that the system is excited in the least excited directions of the input space. A big advantage of the proposed method is that the optimization problem that is required to be solved each time step can be solved by using quadratic programming. This is in contrast to methods proposed in literature that require a non-convex problem to be solved. Two simulation examples are presented that both use a linear parameter-varying model of a mass-spring-damper system. One simulation shows the ability of subspace predictive control to adapt to changing system conditions, while the other simulation shows the effectiveness of the proposed excitation method. Although persistency of excitation comes at the cost of a slightly degraded control performance, it is shown that a satisfactory trade-off can be made between persistency of excitation and control performance.



# Subspace Predictive Control Applied to Fault-Tolerant Control

---

**T**his chapter presents a fault-tolerant control system based on subspace predictive control. In this control method the mathematical model used by conventional predictive controllers to predict the future output is replaced by a subspace predictor. Because the subspace predictor is continuously updated in a closed-loop setting based on new input-output data, it can naturally adapt the controller after a fault has occurred. This property is very useful for fault-tolerant control since faults might be unanticipated. A novel feature of the presented subspace predictive control algorithm is that the predictor is recursively updated in a computationally efficient way. A multiple-model fault isolation scheme is used to distinguish between different anticipated faults. For anticipated faults the control system is specifically configured such that it can accommodate these faults more quickly. For unanticipated faults a more general setting of the control system is used to allow full adaptation. The proposed fault-tolerant control system is evaluated on the Boeing 747 benchmark model.

---

## 6.1 Introduction

In Chapter 5 the basic subspace predictive control (SPC) algorithm has been introduced and it has been extended with a condition that ensures persistency of excitation. In this chapter other extensions are made to the SPC algorithm, which include the derivation of the subspace predictor in a stochastic closed-loop setting and the recursive update of this predictor. In previous papers in which SPC has been used (Favoreel and De Moor 1999; Woodley et al. 2001; Kadali et al. 2003) as well as in Chapter 5, the subspace predictor has been derived using open-loop subspace identification techniques. However, when the SPC algorithm is active, the data gathered to update the predictor inherently is closed-loop data. It has been proven that using closed-loop data from a stochastic system for subspace identification results in a biased predictor (Ljung and McKelvey 1996). Therefore,

a number of different methods have appeared in literature to deal with this issue (Ljung and McKelvey 1996; Favoreel et al. 1999; Jansson 2005). Most of these methods require explicit knowledge of the controller or are based on (overly) stringent assumptions that limit their applicability. Recently, a practically applicable closed-loop subspace identification technique that does not require explicit knowledge of the controller has been developed by Chiuso (2007). Based on this technique a subspace predictor under closed-loop conditions can be derived (Dong et al. 2008), which is also used in this chapter.

Another novel feature of the SPC algorithm presented in this chapter is the way in which the subspace predictor is updated in a recursive manner. This updating scheme differs from others that are based on the “receding horizon” principle, such as, for example, the scheme proposed by Woodley et al. (2001). In the “receding horizon” updating scheme the predictor is based on input-output data from a fixed time window lagging behind the current time step. In the recursive updating scheme new data is appended to the old data, which is discounted with an exponential forgetting factor. This scheme has the advantage that it can be implemented in a computationally efficient manner by using Givens rotations (Golub and Van Loan 1996).

The implementation of SPC as an adaptive controller makes it very suitable for fault-tolerant control (FTC) of aircraft. Most FTC systems deal with faults by using pre-designed or parameter dependent controllers depending on the type of fault that has occurred (Hajiyev and Caliskan 2003). These systems require that the faults either be known in advance or be modeled by a variation of specific parameters (Song et al. 2002; Shin and Belcastro 2006; Belkharraz and Sobel 2007). In this way control designs can be made for each anticipated fault. Besides the fact that this approach can be very involved if many faults are anticipated, unanticipated faults or faults that cannot be modeled by parameter changes such as severe structural damage can occur. An advantage of SPC is that it can adapt on-line to this type of faults. This property is the result of the subspace predictor that is continuously updated using new input-output data. The main contribution of this chapter is to display the usefulness of SPC for realistic FTC problems. The developed SPC-based FTC system is applied to the Boeing 747 benchmark model. Simulations are performed with this model, in which the objective is to fly a pre-defined flight trajectory even after the occurrence of a number of critical faults. The considered fault conditions are stuck control surfaces and the fault condition of the aircraft during the disaster with EL AL flight 1862, that crashed into an apartment building in Amsterdam in 1992. This disaster is also referred to as the “Bijlmerramp” (NRC 2007).

Most aircraft flying today have control laws that are designed using classical single-loop control methods. These methods are preferable over multivariable control methods from a clearance point of view (Fielding et al. 2002). However, single-loop control methods are likely to display a degraded performance in case of faults that cause heavy cross-couplings between flight modes. These cross-couplings are usually the result of loss of symmetry of the aircraft after faults. Multivariable control methods can cope better with these cross-couplings

because they simultaneously achieve several control objectives. Multivariable control methods are therefore to be preferred over single-loop control methods from an FTC point of view (Bodson and Groszkiewicz 1997; Kale and Chipperfield 2005). This is one of the reasons that research into multivariable flight control recently has attracted considerable interest. From this perspective the FTC application of SPC, which is also a multivariable control method, is well motivated.

This chapter is organized as follows. First, the architecture of the FTC system is explained in Section 6.2. Subsequently, the closed-loop SPC algorithm is described in Section 6.3. In Section 6.4 the mechanism that (re-)configures the SPC-based FTC system is explained. The simulation results of this system applied to the Boeing 747 model are given in Section 6.5. Section 6.6 explains how the proposed FTC is implemented in a real-time simulation environment. Finally, concluding remarks are provided in Section 6.7.

## 6.2 Architecture of the Fault-Tolerant Control System

The architecture of the SPC-based FTC system consists of two control loops. The task of the outer control loop is to provide reference signals for the manipulated variables to be tracked by the inner loop. The manipulated variables are roll angle  $\phi$ , pitch angle  $\theta$ , and true airspeed  $V_{TAS}$ , each of which is a function of one of three controlled variables. These controlled variables are the altitude  $h$ , the yaw angle  $\psi$ , and the true airspeed  $V_{TAS}$ , respectively. A desired flight trajectory can be generated by choosing appropriate reference signals for the controlled variables. The architecture of the SPC-based FTC system is depicted in Figure 6.1. In this figure it can be seen that, besides the two control loops, a fault isolation system is present. Both the control loops and the fault isolation system are explained in more detail in the following.

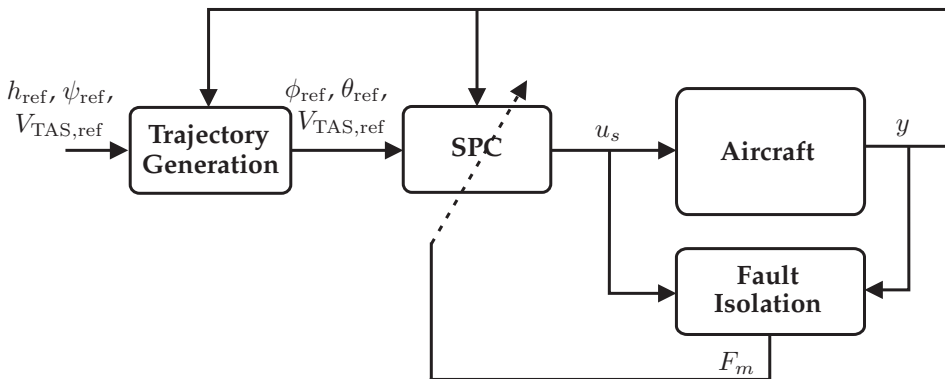


Figure 6.1: Architecture of the SPC-based FTC system.

### 6.2.1 Control Loops

The outer loop is implemented by means of a straightforward proportional integral derivative (PID) scheme. In order to track a desired altitude  $h_{\text{ref}}$ , a pitch angle command is generated as follows

$$\theta_{\text{ref}} = P_{\theta}(h - h_{\text{ref}}) + I_{\theta} \int (h - h_{\text{ref}})dt + D_{\theta} \frac{d(h - h_{\text{ref}})}{dt}, \quad (6.1)$$

where  $P_{\theta}$ ,  $I_{\theta}$ , and  $D_{\theta}$  are design parameters that determine the behavior of the outer loop. The desired yaw angle  $\psi_{\text{ref}}$  is tracked by issuing a roll angle command to the inner loop. This command is generated as follows

$$\phi_{\text{ref}} = P_{\phi}(\psi - \psi_{\text{ref}}) + I_{\phi} \int (\psi - \psi_{\text{ref}})dt + D_{\phi} \frac{d(\psi - \psi_{\text{ref}})}{dt}, \quad (6.2)$$

where  $P_{\phi}$ ,  $I_{\phi}$ , and  $D_{\phi}$  are the design parameters. An anti-windup scheme is implemented for both (6.1) and (6.2) to prevent the integrators from continuing to integrate in case of saturated control signals. The command for true airspeed is generated in the outer loop by directly issuing the true airspeed command to the inner loop. The inner loop is implemented using SPC, which is explained in detail in Section 6.3.

### 6.2.2 Fault Isolation

When SPC is used for FTC, in principle no fault information is required because SPC has the ability to adapt to changed system conditions. However, this adaptation process can take some time. In case of anticipated faults the adaptation can be expedited by using prior knowledge of the fault. This prior knowledge includes information as to which controls should be used to accommodate the anticipated fault. The requirement for the fault isolation scheme used in this chapter is therefore to obtain this information by determining which controls cannot be used anymore due to faults. This requirement is more easily achieved than the requirements for fault detection and isolation (FDI) systems commonly used for FTC.

An important requirement for FDI systems commonly used for FTC is that the faults should be estimated with a certain accuracy, since they are directly used by the FTC system (Song et al. 2002; Pachter and Huang 2003; Shin and Belcastro 2006). If these faults are not estimated accurately enough, poor performance of the FTC system may result. There also exist methods that explicitly take uncertainty of the FDI information into account, such as for example the methods developed by Kanev (2004). A requirement for the application of these methods is that the uncertainty of the FDI information must be known. Obtaining this uncertainty, however, is not a straightforward task. Therefore, the SPC algorithm uses a different philosophy to deal with fault model uncertainty. This philosophy is to let the



controller adapt to a changing system using available input-output data. In this way, no fault model is used and also no fault model uncertainty is required.

Fault isolation is implemented by means of multiple-model estimation. A multiple-model system consists of a model set that contains local models, each corresponding to a specific condition of the system. In an FDI setting, the local models usually represent different fault conditions of the monitored system (Zhang and Li 1998). Besides fault models, the model set also contains the nominal fault-free model of the system. When the system is free of faults, the model corresponding to the nominal case has maximum activation, which corresponds to a model weight of one, and all other models in the model set have a model weight of zero (minimum activation). In case of a fault, one or more of the local models corresponding to faults have model weights larger than zero.

The model set used for fault isolation is derived using the model set design method based on orthogonal decompositions, which has been described in Chapter 3. Since the local models in this model set are valid in a limited region around the operating point at which they have been derived, they are used accordingly. This means that fault isolation is performed only near this operating point in the simulations.

## 6.3 Closed-Loop Subspace Predictive Control

The SPC algorithm (Favoreel and De Moor 1999) elegantly combines a subspace predictor with a generalized predictive control law. When the subspace predictor is updated recursively, SPC has the ability to adapt to unanticipated conditions. In this section, it is first explained how the subspace predictor is derived in a closed-loop setting and how it can be updated recursively, then it is explained how the predictor is integrated with a predictive controller.

### 6.3.1 Closed-Loop Subspace Predictor

Contrary to previous work in which SPC was used (Favoreel and De Moor 1999; Woodley et al. 2001; Kadali et al. 2003), in this chapter the subspace predictor is derived using closed-loop identification techniques. In this previous work, open-loop identification techniques were used under closed-loop conditions. This results in a biased predictor due to correlation between inputs and measurement noise (Ljung and McKelvey 1996). Favoreel et al. (1999) have described an SPC method in which the subspace predictor is based on a closed-loop identification technique, but this technique is based on explicit controller knowledge and also assumes that the controller is time-invariant. This assumption prohibits the use of SPC as an adaptive controller. Therefore, the subspace predictor is derived using the closed-loop identification techniques developed by Chiuso (2007), which do not have the aforementioned limitations. Dong et al. (2008) give a complete explanation of how these identification techniques can be used to derive a subspace

predictor that can be integrated with a predictive control law. In this section, only the basic steps are treated.

### Derivation of the subspace predictor

The model considered for deriving the subspace predictor is a state-space model in innovation form and has the following form

$$x_{k+1} = Ax_k + Bu_k + Ke_k, \quad (6.3)$$

$$y_k = Cx_k + e_k, \quad (6.4)$$

where  $x_k \in \mathbb{R}^n$  is the state of the system,  $u_k \in \mathbb{R}^m$  is the input of the system,  $y_k \in \mathbb{R}^l$  is the output of the system, and  $e_k$  is assumed to be a zero-mean white noise sequence. The matrices  $A$ ,  $B$ ,  $C$ , and  $K$  are the state-space matrices that describe the system. The model described by (6.3)-(6.4) can also be written as

$$x_{k+1} = \Phi x_k + Bu_k + Ky_k, \quad (6.5)$$

where  $\Phi = A - KC$ . Subspace identification is based on relations between matrices that are systematically filled with input-output data. Two of such data matrices that are required for the derivation of the subspace predictor are created as follows

$$Y_k = [y_k \ y_{k+1} \ \cdots \ y_{k+j-1}], \quad (6.6)$$

$$Z_{[k-p,k)} = \begin{bmatrix} u_{k-p} & u_{k-p+1} & \cdots & u_{k-p+j-1} \\ y_{k-p} & y_{k-p+1} & \cdots & y_{k-p+j-1} \\ u_{k-p+1} & u_{k-p+2} & \cdots & u_{k-p+j} \\ y_{k-p+1} & y_{k-p+3} & \cdots & y_{k-p+j} \\ \vdots & \vdots & \cdots & \vdots \\ u_{k-1} & u_k & \cdots & u_{k+j-2} \\ y_{k-1} & y_k & \cdots & y_{k+j-2} \end{bmatrix}, \quad (6.7)$$

where  $p$  denotes the “past” time horizon, the subscript  $[k-p, k)$  denotes the range of the time indices of the entries in the first column of  $Z_{[k-p,k)}$ , and  $j$  denotes the number of columns that is used to create the data matrix  $Z_{[k-p,k)}$ . Usually it holds that  $j \gg p$ . Let  $f$  denote the “future” time horizon, then the following matrix relation can be derived (Chiuso 2007; Dong et al. 2008)

$$\begin{bmatrix} Y_k \\ Y_{k+1} \\ \vdots \\ Y_{k+f-1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ C[B \ K] & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ C\Phi^{f-2}[B \ K] & \cdots & C[B \ K] & 0 \end{bmatrix} Z_{[k,k+f)} + \begin{bmatrix} E_k \\ E_{k+1} \\ \vdots \\ E_{k+f-1} \end{bmatrix} + \begin{bmatrix} C\Phi^{s-1}[B \ K] & C\Phi^{s-2}[B \ K] & \cdots & \cdots & \cdots & C[B \ K] \\ 0 & C\Phi^{s-1}[B \ K] & \cdots & \cdots & \cdots & C\Phi[B \ K] \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & C\Phi^{s-1}[B \ K] & \cdots & C\Phi^{f-1}[B \ K] \end{bmatrix} Z_{[k-p,k)}, \quad (6.8)$$

where  $E_{k+i}$  and  $Y_{k+i}$ ,  $\forall i \in \{0, 1, \dots, f-1\}$ , are defined in a similar manner as  $Y_k$  in (6.6). Note that an important property of (6.8) is that the first block row does not depend on “future” inputs, i.e.  $u_{k+i}$ ,  $\forall i \in \{0, 1, \dots, f-1\}$ . It is this property that allows for an unbiased estimate of the system matrices. In order to estimate the predictor, it suffices to only consider the first block row, which can be written in the compact form

$$Y_k = \Xi_0 Z_{[k-p, k]} + E_k. \quad (6.9)$$

Subsequently,  $\Xi_0$  can be estimated by solving the least squares problem

$$\hat{\Xi}_0 = \arg \min_{\Xi_0} \|Y_k - \Xi_0 Z_{[k-p, k]}\|_F^2, \quad (6.10)$$

where  $\|\cdot\|_F$  denotes the Frobenius norm. The least squares problem defined in (6.10) can be solved, similarly to (5.13) on page 101, by performing an RQ-decomposition

$$\begin{bmatrix} Z_{[k-p, k]} \\ Y_k \end{bmatrix} = \overbrace{\begin{bmatrix} R_{11} & 0 \\ R_{21} & R_{22} \end{bmatrix}}^R \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix}, \quad (6.11)$$

from which the estimate  $\hat{\Xi}_0$  can be computed as

$$\hat{\Xi}_0 = R_{21} R_{11}^{-1}. \quad (6.12)$$

Let  $t$  denote the current time step, then based on the estimate  $\hat{\Xi}_0$ , a subspace predictor of the following form can be derived

$$\begin{bmatrix} \hat{y}_{t+1} \\ \hat{y}_{t+2} \\ \vdots \\ \hat{y}_{t+f-1} \end{bmatrix} = \overbrace{\begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \vdots \\ \Gamma_{f-1} \end{bmatrix}}^{\Gamma_r} \overbrace{\begin{bmatrix} u_{t-p} \\ y_{t-p} \\ \vdots \\ u_{t-1} \\ y_{t-1} \end{bmatrix}}^{w_p} + \overbrace{\begin{bmatrix} \Lambda_1 & 0 & \cdots & 0 \\ \Lambda_2 & \Lambda_1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ \Lambda_{f-1} & \Lambda_{f-2} & \cdots & \Lambda_1 \end{bmatrix}}^{\Lambda_r} \begin{bmatrix} u_t \\ u_{t+1} \\ \vdots \\ u_{t+f-2} \end{bmatrix}, \quad (6.13)$$

where  $\Gamma_r$  and  $\Lambda_r$  are the desired subspace predictor matrices and the parameters  $\Gamma_i$  and  $\Lambda_i$  can be constructed from  $\hat{\Xi}_0$  as

$$\Gamma_i = \hat{\Xi}_i + \sum_{j=0}^{i-1} \hat{C} \hat{\Phi}^{i-j-1} \hat{K} \Gamma_j, \quad (6.14)$$

$$\Lambda_i = \hat{C} \hat{\Phi}^{i-1} \hat{B} + \sum_{j=1}^{i-1} \hat{C} \hat{\Phi}^{i-j-1} \hat{K} \Lambda_j, \quad (6.15)$$

with  $\Gamma_0 = \hat{\Xi}_0$  and  $\Lambda_1 = \hat{C} \hat{B}$ . The parameters  $\hat{\Xi}_i$ ,  $\forall i \in \{1, \dots, f-1\}$  can be

constructed from  $\hat{\Xi}_0$  by using the relation

$$\begin{bmatrix} \hat{C}\hat{\Phi}^{s-1}[\hat{B} \ \hat{K}] & \hat{C}\hat{\Phi}^{s-2}[\hat{B} \ \hat{K}] & \cdots & \cdots & \cdots & \hat{C}[\hat{B} \ \hat{K}] \\ 0 & \hat{C}\hat{\Phi}^{s-1}[\hat{B} \ \hat{K}] & \cdots & \cdots & \cdots & \hat{C}\hat{\Phi}[\hat{B} \ \hat{K}] \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \hat{C}\hat{\Phi}^{s-1}[\hat{B} \ \hat{K}] & \cdots & \hat{C}\hat{\Phi}^{f-1}[\hat{B} \ \hat{K}] \end{bmatrix} = \begin{bmatrix} \hat{\Xi}_0 \\ \hat{\Xi}_1 \\ \vdots \\ \hat{\Xi}_{f-1} \end{bmatrix}, \quad (6.16)$$

where the matrix on the left-hand side of (6.16) is an estimate of the corresponding matrix from (6.8).

### Recursive implementation of the $R$ -update

For the construction of the data matrices  $Y_k$  and  $Z_{[k-p,k]}$  explained in the previous section it was assumed that input-output data was present from time steps:  $k-p, k-p+1, \dots, k+j-1$ . For an adaptive implementation of the subspace predictor, the predictor matrices should be recomputed again each time new data becomes present, i.e. at each new time step. In case of the receding horizon updating scheme, this would mean that new data matrices  $Y_{k+1}$  and  $Z_{[k-p+1,k+1]}$  must be generated using data from time steps:  $k-p+1, k-p+2, \dots, k+j$ . Subsequently, a new estimate for the predictor matrices could be obtained by computing the RQ-decomposition from (6.11) based on the new data matrices. However, computing such an RQ-decomposition at each time step can become computationally expensive for large data matrices. This computation can be prevented by using Cholesky updating and downdating of the  $R$ -matrix (Woodley et al. 2001). The principle of this method is that old data is removed in the downdating step and new data is included in the updating step. These two steps combined require much less computational effort than computing the whole RQ-decomposition. A drawback of using Cholesky updating and downdating is that matrix  $RR^T$  is required to be positive definite at any time. However, this cannot be guaranteed. Therefore, a recursive updating scheme of the  $R$ -matrix is used, which is similar to the one developed by Lovera et al. (2000). This recursive updating scheme differs from the “receding horizon” scheme in the fact that it does not use a fixed window of data. Instead, new data is appended to the old  $R$ -matrix, after it is discounted with an exponential forgetting factor. The recursive updating scheme is explained in the following.

Let the upper left and bottom left block matrix of  $R$  at time step  $t-1$  ( $R(t-1)$ ) be denoted by  $R_{11}(t-1)$  and  $R_{21}(t-1)$ , respectively. If new data becomes available at time step  $t$ , a new vector  $[w_p^T \ y_t^T]^T$  can be created. This vector can be used to update matrix  $R(t-1)$ . The updating step consists of firstly appending  $[w_p^T \ y_t^T]^T$  to  $[R_{11}(t-1)^T \ R_{21}(t-1)^T]^T$ . Subsequently, by applying a sequence of orthogonal Givens rotations (Golub and Van Loan 1996), the matrix is made lower triangular, i.e. updated. This sequence of manipulations is described in the

following equation

$$\left[ \begin{array}{c|c} \sqrt{\lambda}R_{11}(t-1) & w_p \\ \hline \sqrt{\lambda}R_{21}(t-1) & y_t \end{array} \right] \Omega = \left[ \begin{array}{c|c} R_{11}(t) & 0 \\ \hline R_{21}(t) & \tilde{y}_t \end{array} \right], \quad (6.17)$$

where  $\Omega$  denotes the sequence of orthogonal transformations and  $R_{11}(t)$  (which is lower triangular) and  $R_{21}(t)$  are the matrices from which an updated  $\hat{\Xi}_0$  can be computed according to (6.12). A more detailed explanation of  $\Omega$  is given in Appendix C. Note that  $R_{33}$  is not considered in the updating process because it does not influence the computation of  $R_{11}(t)$  and  $R_{21}(t)$ . Also, in (6.17) a forgetting factor  $\lambda \in [0, 1]$  is implemented to discount old data. The smaller the value of  $\lambda$  that is chosen, the more old data is discounted.

### 6.3.2 Closed-Loop Subspace Predictor Integrated with a Predictive Control Law

The predictive control problem has been formulated in Section 5.2.2. In that section the following cost function has been derived

$$J = (\hat{y}_f - r_f)^T Q_a (\hat{y}_f - r_f) + u_f^T R_a u_f + \Delta u_f^T R_a^\Delta \Delta u_f. \quad (6.18)$$

This cost function requires a prediction of the future output, i.e.  $\hat{y}_f$ . The subspace predictor derived in (6.13) can be used for this purpose. In order to include a control horizon, the subspace predictor is modified as follows

$$\hat{y}_f = \Gamma_r w_p + \Lambda_r \overbrace{\begin{bmatrix} I_m & 0 & \cdots & 0 \\ 0 & I_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I_m \\ 0 & \cdots & 0 & I_m \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & I_m \end{bmatrix}}^E u_f, \quad (6.19)$$

where the matrix  $E$  ensures that the input remains constant after the control horizon  $N_c$ . Next,  $\Delta u_f$  can be written as a function of the optimization variable  $u_f$

$$\Delta u_f = \overbrace{\begin{bmatrix} I_m & 0 & 0 & \cdots & 0 \\ -I_m & I_m & 0 & & 0 \\ 0 & -I_m & I_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -I_m & I_m \end{bmatrix}}^{S_\Delta} u_f - \overbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & I_m & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \end{bmatrix}}^{S_w} w_p. \quad (6.20)$$

When relations (6.19) and (6.20) are substituted into (6.18) and the terms that do not depend on  $u_f$  are discarded, the following cost function results

$$\begin{aligned} J(u_f) &= u_f^T (E^T \Lambda_r^T Q_a \Lambda_r E + S_\Delta^T R_a^\Delta S_\Delta + R_a) u_f \\ &\quad + 2 (w_p^T \Gamma_r^T Q_a \Lambda_r E - r^T Q_a \Lambda_r E - w_p^T S_w^T R_a^\Delta S_\Delta) u_f. \end{aligned} \quad (6.21)$$

Constraints should be placed on  $u_f$ ,  $\Delta u_f$ , and  $\hat{y}_f$  according to the physical limitations of the aircraft. These constraints can be formulated as follows

$$U_{\min} \leq u_f \leq U_{\max}, \quad (6.22)$$

$$\Delta U_{\min} \leq \Delta u_f \leq \Delta U_{\max}, \quad (6.23)$$

$$Y_{\min} \leq \hat{y}_f \leq Y_{\max}, \quad (6.24)$$

where  $U_{\min} = [u_{\min}^T \cdots u_{\min}^T]^T$ ,  $\Delta U_{\min} = [\Delta u_{\min}^T \cdots \Delta u_{\min}^T]^T$ ,  $Y_{\min} = [y_{\min}^T \cdots y_{\min}^T]^T$ , and the same notation also holds for the parameters with subscript max. Since the considered optimization variable is  $u_f$ , relations (6.19) and (6.20) are substituted into constraints (6.22)-(6.24). This substitution results in the inequality constraint

$$A_{\text{ineq}} u_f \leq b_{\text{ineq}}, \quad (6.25)$$

with

$$A_{\text{ineq}} = \begin{bmatrix} I_{N_c m} & -I_{N_c m} & S_\Delta^T & -S_\Delta^T (\Lambda_r E)^T & -(\Lambda_r E)^T \end{bmatrix}^T, \quad (6.26)$$

$$\begin{aligned} b_{\text{ineq}} &= \begin{bmatrix} U_{\max}^T - U_{\min}^T (\Delta U_{\max} + S_w w_p)^T & (-\Delta U_{\min} - S_w w_p)^T \\ (Y_{\max} - \Gamma_r w_p)^T & (-Y_{\min} + \Gamma_r w_p)^T \end{bmatrix}^T. \end{aligned} \quad (6.27)$$

The predictive control law can now be formulated as a solution of the following quadratic programming (QP) problem at each time step

$$\begin{aligned} \min_{u_f} \quad & J(u_f) \\ \text{s.t.} \quad & A_{\text{ineq}} u_f \leq b_{\text{ineq}}. \end{aligned} \quad (6.28)$$

Efficient solvers exist for this QP problem (Maciejowski 2002). Note that at each

time step only the first input vector from  $u_f$  is used for control.

The control law (6.28) is derived for linear time invariant systems of the form (6.3)-(6.4). However, in this chapter it is applied to a nonlinear aircraft model. This usage is justified since the nonlinear aircraft model can be approximated well by a linear parameter-varying (LPV) model (Marcos and Balas 2004), which has the same structure as (6.3)-(6.4) but with time-varying system matrices. The variation of the time-dependent parameters is relatively small most of the time. In this case SPC can easily adapt to the time-varying system. Only during fast variations of the time-dependent parameters with respect to the dynamics of the aircraft or during strong nonlinear behavior of the aircraft, SPC can be less accurate.

## 6.4 SPC (Re-)configuration

SPC is a control method that can adapt itself to the system for which it is used. In order to fully exploit these capabilities, preferably all relevant available inputs and outputs should be used to estimate the subspace predictor. Since the Boeing 747 model has 30 control inputs (see Appendix A) and even more outputs, a selection of these inputs and outputs must be made to minimize the computational burden of updating the subspace predictor. Therefore, the SPC-based FTC system is configured such that it uses different sets of control inputs for different fault conditions. For anticipated faults a specific set of inputs is chosen and for unanticipated faults a more general set is chosen. In this way, the changed dynamics in case of anticipated faults can be captured quicker than purely relying on adaptation of SPC. Both sets of control inputs are chosen such that sufficient control redundancy is available to perform “elementary manoeuvres” after the occurrence of a fault. By “elementary manoeuvres” three basic abilities of the aircraft are meant. These are: the ability to descend or ascend, the ability to change heading, and the ability to decelerate or accelerate.

The SPC-based FTC system is demonstrated for three fault conditions, all of which are also used as benchmark faults in GARTEUR AG-16 (Smaili et al. 2006). Two of these three fault conditions are an anticipated elevator lock-in-place and an anticipated rudder runaway. Lock-in-place is characterized by the freezing of a control surface at a certain position, regardless of the actuator commands. Runaway of a control surface is characterized as when the surface suddenly deflects to its maximum or minimum deflection position and locks at that position. These faults can have drastic consequences since they make further operation of the aircraft extremely difficult. The considered rudder runaway fault affects both the upper and lower rudder. The elevator lock-in-place fault affects all 4 elevator surfaces. The two faults are isolated using the multiple-model framework described in Chapter 2 based on model set II described in Section 4.3.4. The third fault condition is the condition of the aircraft during the disastrous “Bijlmerramp” scenario. For this fault condition it is not reasonable to assume that it can be anticipated because of the highly improbable faults that occurred during this disaster. Therefore this fault condition is treated as an unanticipated fault. The faults that occurred

on the aircraft during this disaster include loss of the engines and the pylons on the right wing of the aircraft. This loss caused a shift of the center of gravity of the aircraft, a total weight loss of 10.028 kg and damage to the right wing of the aircraft. This wing damage at its turn resulted in lift loss, increased drag, a yawing moment, and a pitching moment. On top of this, hydraulic systems 3 and 4 malfunctioned, which resulted in reduced or total loss of control authority of a number of control surfaces (Smaili and Mulder 2000).

In the nominal case, the previously mentioned manoeuvres can be performed using SPC with an input vector  $u_k$  consisting of only 4 inputs, which are listed in Table 6.1. Each input can, however, drive more than one of the controls of the Boeing 747 listed in Appendix A. This is because it is assumed that these controls are symmetrically actuated (or asymmetrically in case of the ailerons and spoilers). In Table 6.1 the numbers of the controls driven by the SPC inputs are shown between brackets. The control surfaces that are not directly driven by SPC are chosen constant and equal to a value that is valid for a trimmed situation at the beginning of the flight simulation. For an elevator lock-in-place fault, the SPC-based FTC system uses the stabilizer instead of the elevator surfaces for control of the longitudinal motion. For the rudder lock-in-place fault, the engine controls are subdivided into a control input that controls the left engines and one that controls the right engines such that differential engine thrust can be used when necessary. Furthermore, spoilers are used asymmetrically to increase the control authority in the lateral direction. A positive value of the SPC spoilers input results in a positive deflection of spoilers 5 to 8, while spoilers 13 to 16 remain at a zero deflection. A negative value of the SPC spoilers input results in a positive deflection of spoilers 13 to 16, while spoilers 5 to 8 remain at a zero deflection. For unanticipated faults a set of inputs is chosen with redundant control authority for both longitudinal and lateral dynamics. Note that for anticipated conditions, the input set can be chosen smaller. This has the additional benefit that SPC can be implemented in a more computationally efficient manner.

Besides the input vector  $u_k$ , the SPC-based FTC system also requires a number of measurements from the aircraft to be used in the output vector  $y_k$ . A selection is made from the many available measurements taking into consideration three issues. The first issue is the size of the output vector  $y_k$ , which should be chosen as small as possible to keep the computational requirements low. The second issue is concerned with the quality of the subspace predictor. For this purpose, the chosen outputs should capture the relevant dynamics of the system. Finally, the third issue is concerned with the manipulated variables. The control objective of the SPC-based FTC system is for the reference trajectory  $r_f$  to be tracked by the predicted output vector  $\hat{y}_f$  (see (6.18)). Therefore, the output vector  $y_k$  should include the measurements of the physical quantities to be manipulated. With the previous considerations in mind, 7 outputs are chosen, which are listed in Table 6.2 together with their noise levels. These realistic noise levels correspond to those of conventional aircraft sensors (Breeman 2006).

The SPC-based FTC system should be initialized such that it does not start identifying the system from scratch when a switch is made from nominal opera-



**Table 6.1:** SPC input allocation.

|                        |                                                                                                                                                    |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Nominal case           | Ailerons (1-4)<br>Elevators (17-20)<br>Rudders (22-23)<br>Engines (26-29)                                                                          |
| Elevator lock-in-place | Ailerons (1-4)<br>Stabilizer (21)<br>Rudders (22-23)<br>Engines (26-29)                                                                            |
| Rudder lock-in-place   | Ailerons (1-4)<br>Spoilers (5-8/13-16)<br>Elevators (17-20)<br>Engines left (26-27)<br>Engines right (28-29)                                       |
| Unanticipated faults   | Ailerons (1-4)<br>Spoilers (5-8/13-16)<br>Elevators (17-20)<br>Stabilizer (21)<br>Rudders (22-23)<br>Engines left (26-27)<br>Engines right (28-29) |

tion to an operation mode corresponding to a fault or when the simulation starts from  $T = 0$  s. Therefore, matrix  $R$  is initialized using input-output data obtained from simulation of the open-loop aircraft. In case of anticipated faults, open-loop data of the model with the anticipated fault is used to initialize the  $R$  matrix. And, in case of unanticipated faults, open-loop data of the *nominal* model is used to initialize the  $R$  matrix.

**Table 6.2:** Outputs used for SPC.

| Output          | Symbol    | Unit | Noise (standard dev.) |
|-----------------|-----------|------|-----------------------|
| roll angle      | $\phi$    | deg  | $1.73 \cdot 10^{-4}$  |
| pitch angle     | $\theta$  | deg  | $1.73 \cdot 10^{-4}$  |
| yaw angle       | $\psi$    | deg  | $1.73 \cdot 10^{-4}$  |
| true airspeed   | $V_{TAS}$ | m/s  | $1.00 \cdot 10^{-1}$  |
| angle of attack | $\alpha$  | deg  | $1.73 \cdot 10^{-3}$  |
| sideslip angle  | $\beta$   | deg  | $1.73 \cdot 10^{-3}$  |
| altitude        | $h$       | m    | $1.00 \cdot 10^{-1}$  |

## 6.5 Simulation Results

In this section the results of four simulations are presented. In all four simulations a flight scenario is flown consisting of an initial straight and level flight at an altitude of 980 m. During this first flight phase, the faults are inserted. Next, a second phase consisting of a heading change is initiated. The third and final flight phase of the trajectory consists of a descent to an altitude of 100 m. In the first simulation, the flight scenario is simulated without any faults. In the second, third, and fourth simulation, faults are injected during the first flight phase. In the second simulation a lock-in-place fault of the elevators is injected, in the third simulation a rudder runaway fault is injected, and in the fourth simulation the faults that occurred during the “Bijlmerramp” are injected.

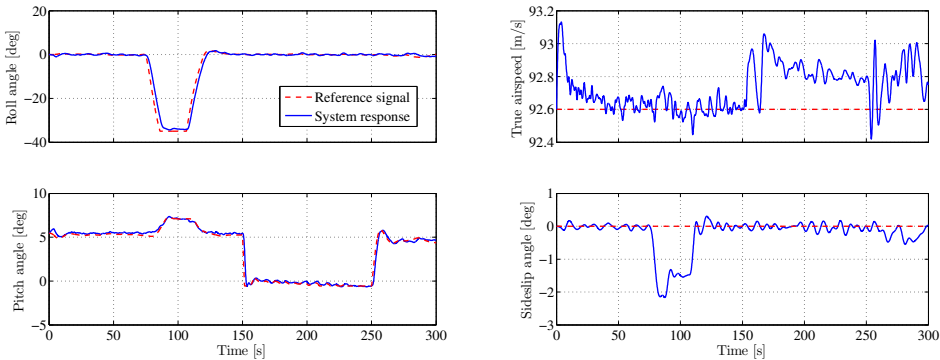
Before the actual simulation results are presented, the choices for the simulation settings and tuning parameters are described first. The aircraft model is simulated at a sampling frequency of 100 Hz. The fastest mode of the aircraft that has been observed from linearizations of the nonlinear aircraft model at different operating points is about 0.25 Hz. The sampling frequency of the SPC-based FTC system is 10 Hz, which is chosen sufficiently fast relative to the aircraft dynamics. The SPC parameters are chosen as:  $p = 20$ ,  $f = 20$ ,  $\lambda = 0.995$ ,  $N_p = f$ , and  $N_c = 5$ . The subspace predictor parameters  $p$  and  $f$  are chosen relative to the aircraft dynamics. The parameter  $\lambda$  is tuned such that the predictor is modified just enough at each time step to cope with the varying dynamics. The weighting matrices  $Q_a$ ,  $R_a$ , and  $R_a^\Delta$  are tuned relative to each other based on a combination of simulation experience and tuning rules as described in Chapter 7 of the book by Maciejowski (2002). In that chapter, it is for example argued that increasing  $R_a$  and  $R_a^\Delta$  relative to  $Q_a$  leads to reduced control activity. Although reduced control activity can be very desirable for a stable steady-state condition, it limits the ability of the controller to deal with disturbances, which can result in an unstable closed-loop system in extreme cases. Therefore, in tuning the weights, a trade-off has to be made between having little control activity and having the ability to deal with disturbances. Next, it is noted that the weighting matrices are tuned differently for the different settings described in Table 6.1. Furthermore, weighting matrix  $Q_a$  only contains nonzero entries on its diagonal for the entries that are manipulated by SPC, i.e.  $\phi$ ,  $\theta$ ,  $V_{TAS}$ , and  $\beta$ .

The tuning procedure for the outer loop parameters  $P_\theta$ ,  $I_\theta$ ,  $D_\theta$ ,  $P_\phi$ ,  $I_\phi$ , and  $D_\phi$  is based on simulation experience, similar to the weighting matrices. Parameter  $j$ , which determines the number of columns in the data matrices in (6.6) and (6.7) is chosen to have a value of 1000. This means that the data matrices contain  $1000/10 \text{ Hz} = 100 \text{ s}$  of data. Note that these large data matrices are created only once for each condition. Once an  $R$ -matrix is computed based on these data matrices, only the  $R$ -matrix is used and updated by the SPC algorithm. The  $R$ -matrix is generally much smaller than the data matrices since its dimensions do not depend on  $j$ . All simulations have been performed under closed-loop conditions with measurement noise levels as described in Table 6.2. Moreover, turbulence that is modeled using the Dryden spectra (Van Der Linden 1998) is added to the

simulated aircraft.

### 6.5.1 Trajectory Following for the Nominal Case

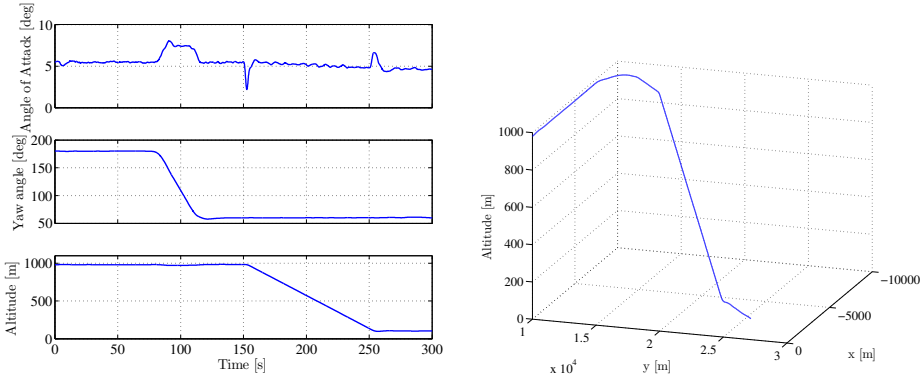
In this section, the simulation results for the nominal condition are presented. The flight trajectory starts with a straight and level flight at an altitude of 980 m, a true airspeed of 92.6 m/s, and a flap setting of 20 deg. This initial condition corresponds to the initial condition defined in GARTEUR AG-16. During the first flight phase the control objective is to maintain a constant altitude, yaw angle, velocity, and sideslip angle. Next, at  $T = 75$  s a change in yaw angle from 180 deg to 60 deg is initiated. Finally, at  $T = 150$  s a descent is initiated to an altitude of 100 m. This descent is performed with a fixed flight path angle  $\gamma$  of  $-5$  deg. In Figure 6.2 the references for the manipulated variables are represented by dashed lines. It can be seen that the reference signals are tracked very well, especially when the fact is considered that the SPC-based FTC system is completely data-driven. It can be seen that during the heading change maneuver, the sideslip angle is allowed to have a minimal tracking error, preventing large surface deflections. The flight trajectory is depicted in Figure 6.3 as well as the angle of attack, yaw angle, and the altitude. The actuator deflections and the engine commands are depicted in Figure 6.4. The engine commands are expressed in engine pressure ratio (EPR). It can be seen that the control signals are quite smooth and remain well within their operating limits, which is a result of the constraints on  $u_f$ .



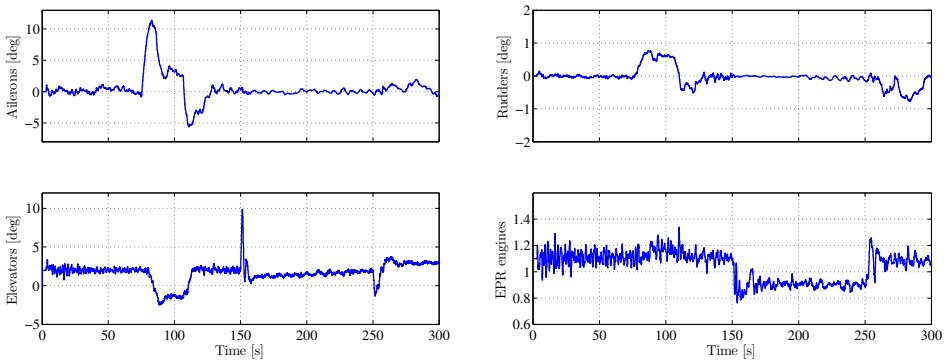
**Figure 6.2:** Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for the nominal condition. The dashed signals correspond to the control reference signals.

### 6.5.2 Trajectory Following for Elevator Lock-in-Place

In this section, the simulation results for elevator lock-in-place are presented. The simulation starts with the same initial condition as is described in the previous



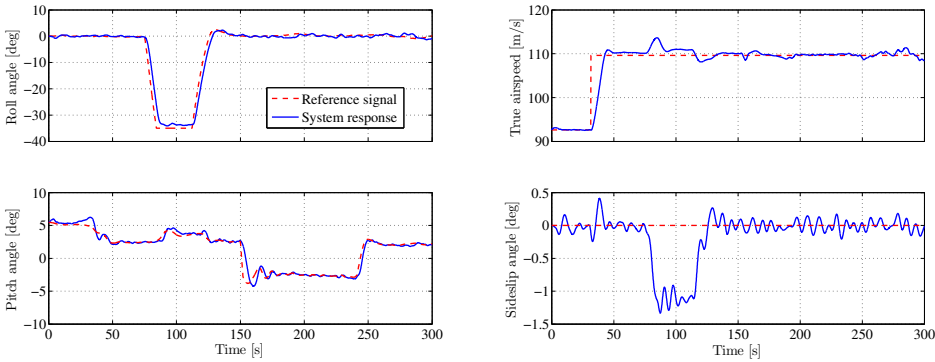
**Figure 6.3:** Angle of attack, yaw angle, altitude, and trajectory of the aircraft for the nominal condition.



**Figure 6.4:** Actuator deflections and engine commands for the nominal condition.

section for the nominal case. The elevator lock-in-place fault is injected at  $T = 18$  s at a deflection of 1.9 deg. The fault is correctly isolated at  $T = 28$  s. The relatively large isolation delay is a result of the fact that the elevator locks at a deflection position, which exactly suits the flight condition at that time. So, the faults can not be isolated until the aircraft is sufficiently excited by turbulence. It can be seen in Figure 6.5 that the reference signal for the true airspeed has been increased just after isolation of the fault. This has been done to increase the effectiveness of the stabilizer surface to allow sufficient control authority. Furthermore it can be seen that tracking of the reference signals is performed satisfactorily. Only during the descent, which is again performed with a fixed flight path angle of  $-5$  deg, the pitch angle command is tracked with a small error. In Figure 6.6, the angle of attack, yaw angle, and altitude are depicted together with the flight trajectory. For comparison purposes, the same trajectory is also flown using the autopilot from the GARTEUR AG-16 benchmark, the result of which is indicated by a grey signal in the figure showing the flight trajectory. It can be seen that the result of the

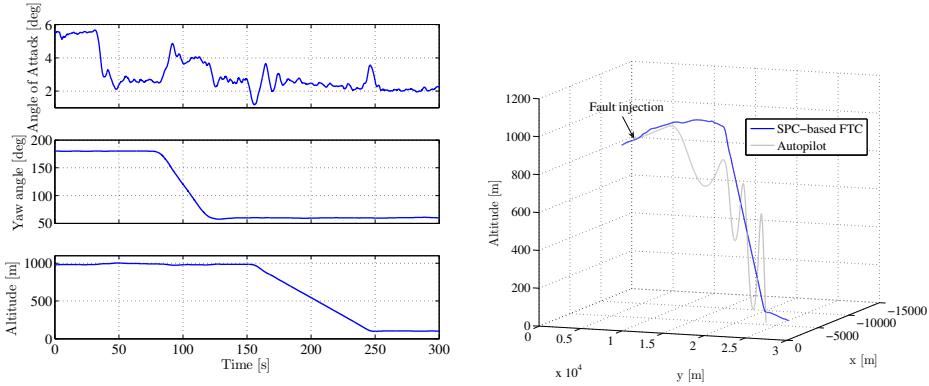
fault is a pitching moment which cannot be counteracted by the autopilot since it does not have control over the stabilizer. Therefore, when the autopilot is used, human pilot intervention is required to accommodate this fault. Since the elevator lock-in-place fault does not affect lateral motion, the heading change maneuver is still performed adequately by the autopilot. In Figure 6.7 the actuator deflections and engine commands of the SPC-based FTC system are shown. It can be seen that the elevator deflection remains constant after the fault is injected and that the stabilizer takes over after the fault is isolated. Note also that the rate of change of the stabilizer input is small when compared to the other surfaces. The reason for this is that the stabilizer surface has a maximum deflection rate of  $0.5 \text{ deg/s}$ , which is about 100 times smaller than the other surfaces. Generally, it can be concluded from these simulation results that the reaction on the fault is performed quickly and adequately as a result of the available prior knowledge being open-loop simulation data from a similar fault condition. This prior knowledge has significantly reduced adaptation time.



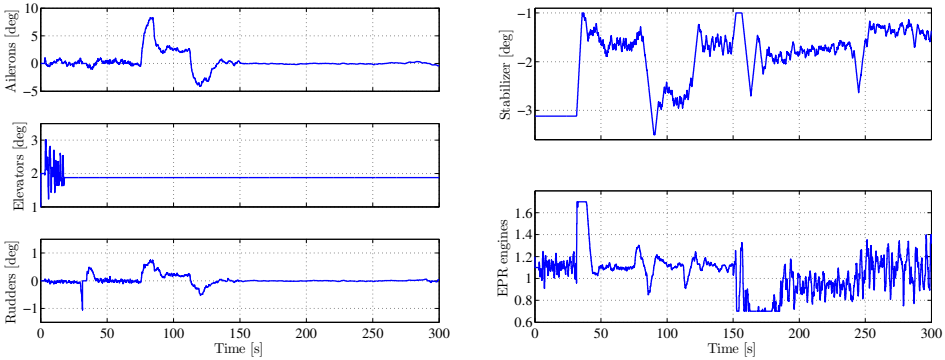
**Figure 6.5:** Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for elevator lock-in-place. The dashed signals correspond to the control reference signals.

### 6.5.3 Trajectory Following for Rudder Runaway

In this section, the simulation results for rudder runaway are presented. The rudder runaway fault is injected at  $T = 18 \text{ s}$ . After this, the upper and lower rudder surfaces start moving with a rate of  $50 \text{ deg/s}$  from their position at  $T = 18 \text{ s}$  to the maximum deflection position of  $25 \text{ deg}$ . The rudder runaway fault is isolated at  $T = 22 \text{ s}$ . It can be seen in Figure 6.8 that the aircraft starts to slip immediately after insertion of the fault and that the reference signals are not tracked very well just after the fault. This is because SPC needs some time to gather data for adapting to the faulty condition. After this has been done, the reference signals are tracked satisfactorily again, except for the sideslip angle. The reason for this is that it cannot be controlled completely towards zero due to the severity of the



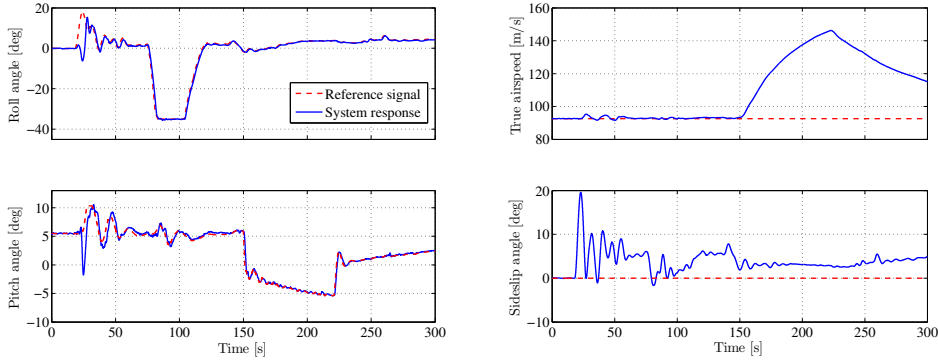
**Figure 6.6:** Angle of attack, yaw angle, altitude, and trajectory of the aircraft for elevator lock-in-place. In the trajectory plot, the gray line corresponds to the trajectory flown with the autopilot.



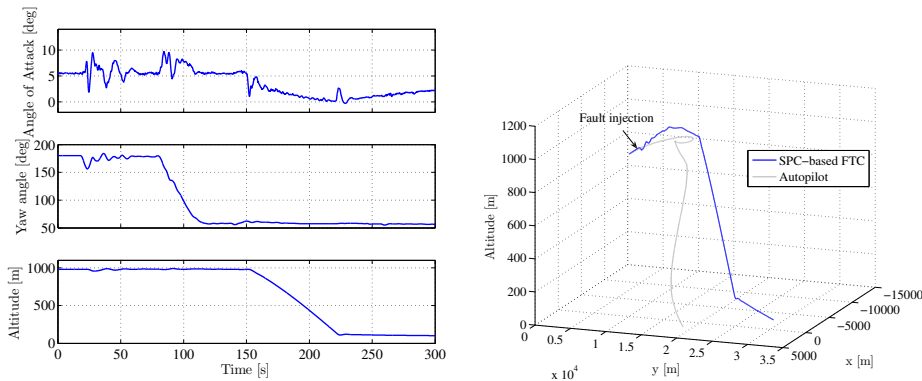
**Figure 6.7:** Actuator deflections and engine commands for elevator lock-in-place.

fault. At  $T = 75$  s the heading change is initiated. Subsequently, at  $T = 150$  s a descent to 100 m is initiated with a fixed flight path angle of  $-5$  deg. Note that the aircraft picks up speed in this descent. This is the result of the fact that the engines are required to provide differential thrust to counteract the yawing moment of the rudder runaway and can therefore not reduce thrust. In Figure 6.9 it can be seen that both the heading change and the descent manoeuvre are performed adequately. Furthermore, it can be observed that the autopilot is unable to counteract the yawing moment resulting from the rudder runaway fault, not even with a full deflection of the spoilers and ailerons. It is therefore clear that the human pilot must intervene to try to accommodate the fault. In Figure 6.10 it can be seen that after the fault some time is required before the control signals become smooth again, which is a result of the adaptation process. Also, it can be seen how the ailerons work together with the engines (providing differential thrust), and the spoilers to counteract the yawing moment resulting from the rudder runaway

fault. Next, it can be observed that in the time interval  $T = 150 - 300$  s the rudders have moved away from their maximum deflection position of 25 deg because the aircraft picks up speed resulting in a reduced blowdown limit, which means that the rudders are forced back towards their neutral position.



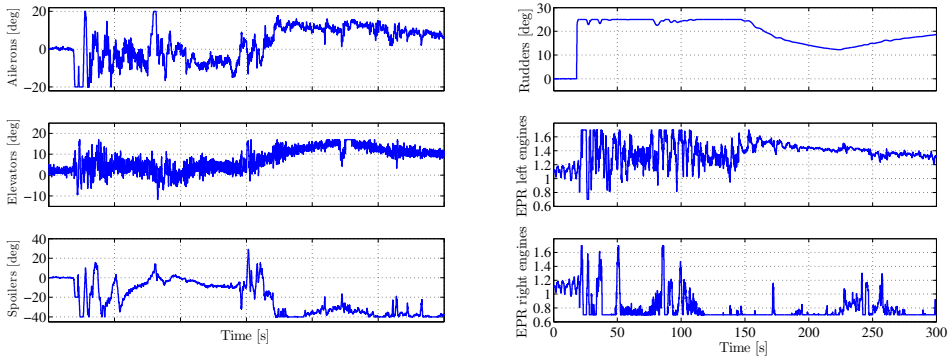
**Figure 6.8:** Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for rudder runaway. The dashed signals correspond to the control reference signals.



**Figure 6.9:** Angle of attack, yaw angle, altitude, and trajectory of the aircraft for rudder runaway. In the trajectory plot, the gray line corresponds to the trajectory flown with the autopilot.

#### 6.5.4 Trajectory Following for “Bijlmerramp” Condition

In this section, the simulation results for the “Bijlmerramp” fault condition are presented. The simulation setting in this section differs from the setting of the previous three simulations in the fact that it can accommodate unanticipated faults.



**Figure 6.10:** Actuator deflections and engine commands for rudder runaway.

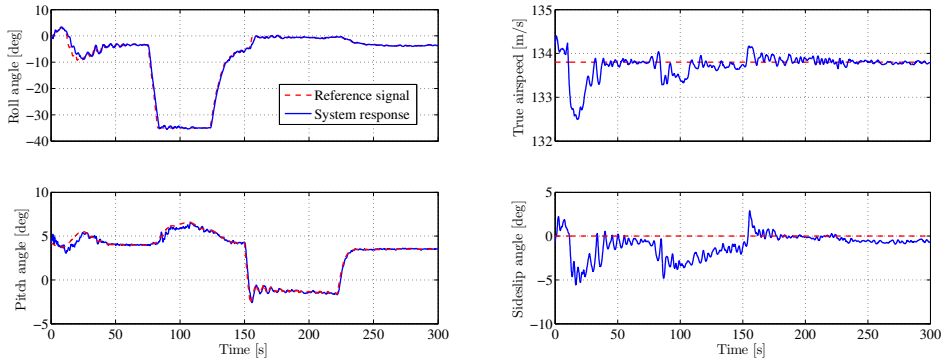
The setting for unanticipated faults continuously uses 7 inputs to control the aircraft, as is described in Table 6.1. Furthermore, no FDI is used for this setting. The simulation starts at an altitude of 980 m, a true airspeed of 133.8 m/s, and a flap setting of 1 deg according to the initial conditions defined in GARTEUR AG-16 for this specific fault. The fault is injected at  $T = 10$  s. Immediately after injection of the fault, the aircraft starts to roll and slip as can be seen in Figure 6.11. However, the SPC-based FTC system manages to quickly regain control and track the reference signals again after a period of about 15 s. In Figure 6.12 it can be seen that the trajectory can be flown safely even after occurrence of the very severe fault condition. Furthermore, it can be seen that the autopilot is not capable of safely flying the aircraft, since it crashes about 50 s after the injection of the fault. In Figure 6.13 the actuator deflections and the engine commands for the “Bijlmerramp” scenario are shown. It can be seen that the right engines immediately stop providing thrust after the fault is injected. Furthermore, it can be observed that the stabilizer is used in a limited range to prevent overly large altitude fluctuations due to the slow operation of this surface. An important conclusion that can be drawn from this simulation is that the SPC-based FTC system is able to adapt to an unanticipated condition, which severely changes the dynamics of the aircraft.

### 6.5.5 Discussion of the Simulation Results

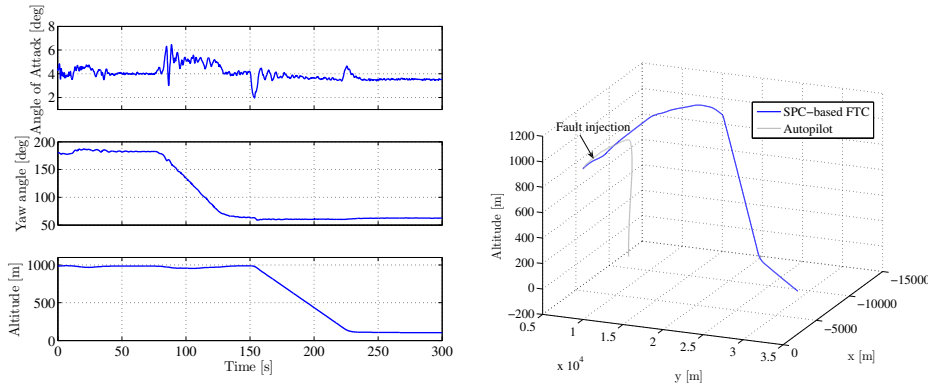
The presented simulation results show that by using the proposed methodology it is possible to design a controller for the nominal and faulty aircraft using only input-output data. This conclusion is remarkable, especially when the complexity of the aircraft model is considered. Two desirable properties of the proposed control design methodology are described in the following:

1. Modeling of the system to be controlled takes up a large part of the design process of model-based controllers. Since the proposed methodology pro-





**Figure 6.11:** Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for the “Bijlmerramp” fault condition. The dashed signals correspond to the control reference signals.

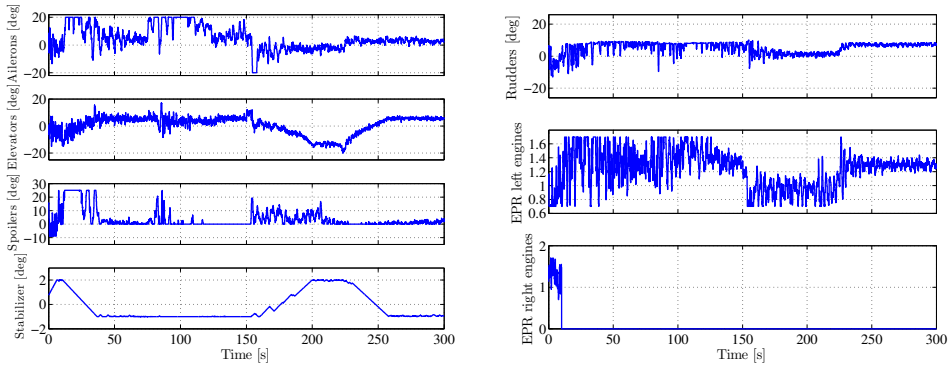


**Figure 6.12:** Angle of attack, yaw angle, altitude, and trajectory of the aircraft for the “Bijlmerramp” fault condition. In the trajectory plot, the gray line corresponds to the trajectory flown with the autopilot.

vides a framework to derive a controller using only input-output data, a significant amount of time can be saved in the design process.

2. For fault-tolerant control it is often required to have a model of the post-fault system. This requirement results in the impossibility of providing fault-tolerant control for all possible faults since not all possible faults can be anticipated. However, the proposed methodology has the ability to adapt on-line to unanticipated faults using input-output data. Therefore, it is a very suitable method for fault-tolerant control.

Opposing these desirable properties, there is an important issue of SPC that has not been addressed in this chapter. This issue is concerned with guaranteed stability of the SPC algorithm. Although from the many simulations that have



**Figure 6.13:** Actuator deflections and engine commands for the “Bijlmerramp” fault condition.

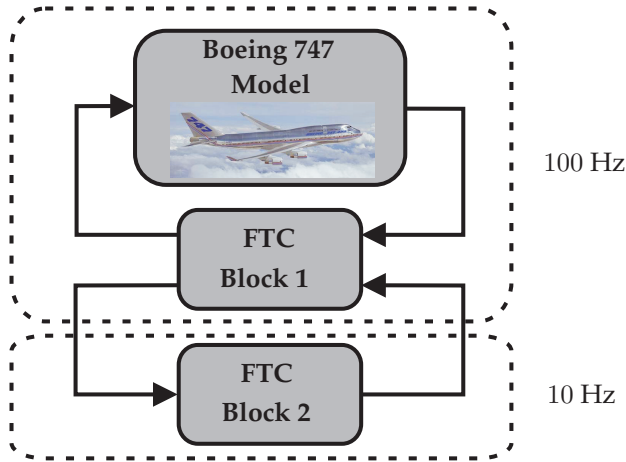
been performed there is no reason to believe that stability is an insuperable issue, a formal proof for stability is as of yet not available. It should be noted that SPC forms no exception in this aspect, since global stability proofs for a number of other adaptive controllers are also not available. An example of such an adaptive control scheme is robust multiple-model adaptive control (Fekri et al. 2006a), for which a proof for global asymptotic stability is also lacking.

## 6.6 Real-Time Implementation

The simulation results of the SPC-based FTC system presented in the previous section have been obtained using off-line simulations. An important property of a control method that is meant for real-time on-line implementation is its computational requirements. These requirements should not be too large such that they restrict a practical implementation for realistic systems. In order to demonstrate that the presented SPC-based FTC system does not have too restrictive computational requirements, an on-line version has been developed. This on-line version has been created in the scope of GARTEUR AG-16. In this project the participants have been invited to develop on-line FTC schemes for implementation on the SIMONA research flight simulator (SIMONA 2007). A real-time simulator environment has been developed specifically for this research simulator. This environment, which has been named Delft University Environment for Communication and Activation (DUECA) (Van Paassen et al. 2000), poses different requirements to the FTC system than the off-line simulation environment, which is MATLAB/Simulink.

An important requirement of the on-line simulation environment is that all computations required for the FTC system should be finished well within the sample time of the Boeing 747 benchmark model, which is 0.01 s. Since the computations required for the developed SPC-based FTC system are too heavy to be finished within 0.01 s, a multi-rate real-time architecture has been developed. This

architecture consists of 2 blocks that run at different operating frequencies. One block runs at the same sampling frequency as the aircraft model and one block runs at a sampling frequency of 10 Hz. A schematic diagram of the multi-rate architecture is shown in Figure 6.14. In Block 2 the time-consuming computations that cannot be finished within 0.01 s are performed. These computations include the update of the subspace predictor and the solver for the quadratic programming problem (6.28). Block 1 contains the less demanding computations, such as the computations required for the multiple-model FDI system. It should be noted that the sampling frequency of 10 Hz of Block 2 is chosen sufficiently fast relative to the dynamics of the Boeing 747 model.



**Figure 6.14:** Schematic diagram of the multi-rate real-time architecture.

The tuning parameters of the on-line SPC-based FTC system that determine the computational requirements are chosen as:  $N_p = 20$ ,  $N_c = 5$ ,  $p = 20$ ,  $f = 20$ ,  $m = 5$ , and  $l = 7$ . Furthermore, the maximum number of iterations of the solver for the quadratic programming problem has been set to 100 to ensure that the available computation time is never violated. The described parameter configuration results in an SPC-based FTC system that is fast enough to be run on the DUECA simulation environment using a computer with an AMD Athlon 64 X2 5600+ processor operating at 2.8 GHz and 4 Gb of RAM. It should be remarked, however, that it has not been possible to implement the setting for unanticipated faults sufficiently fast on this computer. Because for this setting it holds  $m = 7$ , *ceteris paribus*. Since the on-line results are similar to the off-line results, which have been previously presented, no on-line results are presented in this chapter. In conclusion, it is remarked that the on-line version of the SPC-based FTC system demonstrates that it is indeed possible to perform real-time data-driven adaptive control of a complex system such as the Boeing 747 benchmark model.

## 6.7 Conclusions

A reconfigurable fault-tolerant control system has been presented that is able to adapt on-line to faults. This system consists of a subspace predictor, derived in a closed-loop setting, combined with predictive control. The subspace predictor, which does not require knowledge of a mathematical model, is continuously updated on-line using new input-output data. It is this property that gives the proposed system its ability to adapt to faults. These faults may be either anticipated or unanticipated. In case of anticipated faults, prior knowledge of the faults allows the changed dynamics to be captured faster than purely relying on adaptation. A special setting for unanticipated faults has been designed that uses more control inputs than for anticipated faults to fully exploit the adaptation capabilities. The proposed fault-tolerant control system is evaluated in simulation on a detailed model of a Boeing 747. In the performed simulations, three fault conditions have been successfully accommodated. These fault conditions include an elevator lock-in-place, rudder runaway, and the “Bijlmerramp” fault condition. In the simulations it could be observed that the controller requires some time to adapt to the new fault situation. This is an inevitable consequence of the data-driven adaptation concept. However, it can be concluded from the performed simulations that the system allows to safely perform the required elementary maneuvers in both the nominal condition and the considered fault conditions.

# Conclusions and Recommendations

---

In this thesis two research objectives have been addressed: one that deals with the development of diagnosis methods using the multiple-model framework and one that deals with the development of an adaptive fault-tolerant controller. In this chapter the main conclusions of this research are provided. As with any research, new questions can be formulated based on the research presented in this thesis. These new research questions are formulated in this chapter as recommendations for further research.

---

## 7.1 Conclusions

### Convex Model Structure for Multiple-Model Estimation

The multiple-model (MM) framework is a suitable framework for fault diagnosis because of its extensive modeling capabilities. Each of the local models in the model set used by an MM system can have totally different dynamics. This is a very useful property for fault diagnosis since it allows many types of faults to be modeled. The mainstream methods for MM estimation use a hybrid model structure. An important drawback of this structure is that it requires that all model conditions that can occur, should be approximated by a local model in the model set. If this is not the case then the estimation performance of the MM system can deteriorate significantly. For fault diagnosis this issue is especially relevant since many fault conditions can also occur partially, for example a partial sensor fault. This would mean that ideally the model set should consist of infinitely many local models, each corresponding to a specific partial fault size. However, such a model set is not a computationally viable solution.

In Chapter 2 an alternative structure has been proposed for the hybrid MM system to overcome the aforementioned drawbacks. This structure is characterized by the fact that it explicitly interpolates between the local models. The resulting improved interpolation properties allow smaller model sets to be used for the proposed MM system. An additional advantage with respect to the hybrid MM

structure is that a transition probability matrix is not required. This matrix, which contains the transition probabilities from one local model to another local model, can be difficult to design in practice.

The estimation problem related to the proposed MM system is nonlinear. This is caused by the product between the state of the MM system and the model weights. Two filters have been proposed to solve the estimation problem: one filter estimates the state and the model weights in one step while the other filter uses two separate steps. These two filters are compared with a representative filter for the hybrid model structure. This representative filter is the interacting MM (IMM) filter, which has been chosen for its simplicity and widespread use in MM estimation. The comparison between the three filters is performed using two relevant Monte-Carlo simulation examples, one of which involves a target tracking problem and one of which involves a fault diagnosis problem. Both examples clearly show that the alternative MM structure is to be preferred over the hybrid MM structure when the actual system condition is not represented by a local model in the model set but by a weighted combination of these local models. Furthermore, it is illustrated that the alternative structure allows for the use of smaller model sets, while at the same time the estimation performance is not negatively affected.

### Model Set Design

If many different conditions are to be represented by a model set, then choosing which local models to include in the model set can prove to be a difficult task. Therefore, an important issue when using the MM framework is the structured design of model sets. This issue has attracted very little attention in literature. For most MM designs encountered in literature this issue is either resolved by considering only a limited number of system conditions to be modeled or by using heuristic model set design methods. Three methods for structured model set design are presented in Chapter 3, all of which are based on approximation of the parameter space of the varying parameters. Two of these methods assume that linear models can be obtained at any desired operating point of the system to be modeled. In this way a large model set can be generated that is subsequently reduced by these two methods. One of these two methods uses orthogonal decompositions (ODs) for reducing the large model set. The other method uses an algorithm that computes a convex polytope (CP), which has a limited number of vertices and contains all models in the large model set. These vertices are subsequently used to form the local models in the reduced model set. An important difference between the OD-based method and the CP-based method is that the local models obtained with the OD-based method are physically interpretable, while the local models obtained with the CP-based method are not. A third model set design method is presented that is based on the limit values (LV) of the system parameters. For this method the model equations are assumed to be known exactly, which is not the case for the other two methods.

The three proposed methods for structured model set design have been evaluated on a linear parameter-varying model of a mass-spring-damper system. It is

observed that the MM estimation algorithms based on each of the three methods have good state estimation performance. The assessment of the model weight estimation performance for the model sets created with the OD-based method and CP-based method cannot be performed directly. The reason for this is that the estimated model weights do not have a direct relation with the simulated varying model parameters. Therefore, a method has been proposed that reconstructs the varying model parameters from the estimated model weights. These reconstructed parameters are shown to correspond well to the simulated parameters for both model sets. The estimated model weights from the model set obtained with the LV-based method are directly related to the model parameters and therefore no reconstruction is necessary. The model parameters estimated with this method also correspond well to the simulated parameters. All three methods result in comparable estimation performances for the considered example, but differ in their properties. Therefore, it is concluded that the most important reason to choose one method over the other is the suitability of the method for a specific application.

### **Model Sets for Aircraft Fault Diagnosis**

The Boeing 747 benchmark model used throughout this thesis considers lock-in-place faults of control surfaces and the faults that occurred during the disastrous EL AL flight 1862 accident, which is known as the “Bijlmerramp”. In Chapter 4 two model sets are designed for this benchmark model using the OD-based method. The first model set is designed to model lock-in-place faults in two groups of control surfaces, a range of multiplicative sensor faults in one sensor, and two rare faults that occurred during the “Bijlmerramp” flight. The model set resulting from the OD-based method contains the two models corresponding to the two rare faults and it also contains models corresponding to the lock-in-place faults and the sensor fault. It is therefore concluded that this structurally designed model set compactly represents the considered faults. It should be noted, however, that it is not reasonable to assume that the faults that occurred during the disastrous flight can be anticipated. The purpose of the design of the first model set is therefore only to illustrate the versatility of the proposed model set design method. A second model set has been designed that considers lock-in-place faults in three groups of control surfaces that can indeed be reasonably anticipated. These faults are represented in the model set by local models that correspond to the condition at which the control surfaces are stuck at their maximum or minimum deflection position.

### **Persistency of Excitation for Subspace Predictive Control**

Subspace Predictive Control (SPC) is a control method that is characterized by the combination of a subspace predictor and a predictive control law. The subspace predictor, which is derived using input-output data of the system is used to predict the future output for the predictive controller. An important property of SPC is that it can adapt to system variations if it is implemented in an adaptive manner, i.e. if the subspace predictor is updated on-line with new input-output data. This updating scheme makes the adaptive controller a data-driven one. In Chapter 5,

SPC has been applied for control of a linear parameter-varying model of a mass spring damper system. It has been shown that SPC can indeed adapt to parameter variations of this model as opposed to a model predictive controller that uses a fixed model. An important issue in SPC is that the system must be persistently excited so as to ensure that the predictor contains sufficient model information. Therefore, in Chapter 5 a method has been proposed that additionally excites the system in the directions of the input space that are least excited. This method has the property that it allows the optimization problem related to the constrained predictive control law to remain convex and quadratic. In this way the optimization problem can still be solved by using quadratic programming, which can be efficiently implemented.

### **Subspace Predictive Control applied to Fault-Tolerant Control**

Most conventional fault-tolerant control (FTC) systems require faults to be anticipated. However, in practice faults may occur that cannot be anticipated. Therefore it is very convenient if the controller has the ability to adapt to the system after an unanticipated fault has occurred. SPC is a control method that has the ability to adapt on-line to a varying system, which therefore makes it very suitable for FTC. An SPC-based FTC system has been developed for the Boeing 747 benchmark model in Chapter 6. The faults that this system can handle may be either anticipated or unanticipated. Anticipated faults are isolated by using a scheme based on the MM framework. The model set used for this isolation scheme is described in Chapter 4. In case of anticipated faults, prior knowledge of the faults allows the changed dynamics to be captured faster than purely relying on adaptation. This prior knowledge includes information on which control inputs should be used to accommodate a specific fault. A special setting for unanticipated faults has also been designed that uses more control inputs than for anticipated faults to fully exploit the adaptation capabilities. The simulation results show that the SPC-based FTC system is able to successfully accommodate anticipated faults, such as elevator lock-in-place and rudder runaway. Furthermore, the developed FTC system is also shown in simulation to be able to adapt to the unanticipated fault conditions, which had occurred during the “Bijlmerramp” flight.

Besides the off-line version of the SPC-based FTC system an on-line version of this system has been developed. This system is fast enough to run on the real-time environment of the SIMONA aircraft simulator at the Aerospace Engineering department of the Delft University of Technology. With this on-line version of the SPC-based FTC system it is demonstrated that it is indeed possible to perform real-time data-driven adaptive control of a complex system such as the Boeing 747 benchmark model.



## 7.2 Recommendations for Further Research

In Chapter 2, the Convex Model Filter (CMF) has been introduced for the combined estimation of the system state and the model weights of an MM system. This filter applies a fixed forgetting factor in the model weight estimation step. This forgetting factor is used to discount old data. In practical situations the model weights can vary in various manners: slow in steady-state situations and abrupt when faults occur. These two types of variations ideally require two different values for the forgetting factor to achieve good estimation performance. So, when a fixed forgetting factor is chosen a trade-off should be made between tracking performance in steady-state situations and during transients. Therefore, it is recommended to integrate an adaptive forgetting factor in the CMF that can deal with both situations.

The two filtering methods proposed in Chapter 2 both provide approximate solutions to the original nonlinear problem of estimating the state and the model weights. It is worthwhile to investigate how the estimation performance of these filters can be improved by a nonlinear filter or an advanced approximating filter such as the particle filter. It is especially interesting to assess whether the additional computational cost involved with the suggested filters significantly pays off in additional performance for the considered estimation problem.

A requirement for good model weight estimation performance is that the system should be excited well enough. One way to do this is to sufficiently excite the system by practical matters such as performing maneuvers or including turbulence and noise. Another way, which has not been considered in this thesis, is to design excitation experiments specifically aimed at distinguishing between local models. This field of research has received little attention, while it is highly relevant for the MM framework. It is therefore recommended to perform further research in this field. Good starting points are the work by Blackmore and Williams (2006) and the book by Campbell and Nikoukhan (2004).

This thesis provides a number of contributions to SPC theory. One research item that remains open for further research is concerned with the global asymptotic stability of the closed-loop system when using SPC. Although this thesis shows that SPC is stable for the many simulations that have been performed, a formal proof is lacking. It should be noted, however, that even for established methods such as robust multiple-model adaptive control (Fekri et al. 2006a) a proof for global asymptotic stability is lacking. A good starting point in the search to a proof for global stability of the SPC algorithm would be to first consider the unconstrained case. In this way the nonlinear behavior as a result of constraints can be disregarded.

Another useful addition to SPC theory is to incorporate the uncertainty in the determination of the predictor matrices directly in the predictive control law. This would make the SPC algorithm more robust to predictor uncertainty. A good measure for this uncertainty is the covariance matrix of the least squares problem that should be solved to obtain the predictor matrices.

Chapter 5 has presented a method that ensures persistency of excitation for SPC. This method has been developed for the case in which the subspace predictor is derived under open-loop conditions. In Chapter 6, the subspace predictor has been derived under closed-loop conditions. An interesting item for further research is therefore to extend the proposed persistency of excitation method to closed-loop conditions.

In Chapter 6, the SPC-based FTC system has been applied to recover from three faults. For two of these faults, namely the elevator lock-in-place fault and the fault condition corresponding to the “Bijlmerramp”, the commanded true airspeed is set to a relatively high level by the outer loop. This has been done to increase the control authority of the remaining control surfaces. How this and the other outer loop commands should be modified as a function of faults has not been addressed in this thesis. It is, however, an important research topic that deserves further investigation. A possible solution is to estimate the post-fault flight envelope and to subsequently modify the outer loop commands such that the aircraft remains inside this flight envelope.

# A

## Flight Parameters and Controls of the Boeing 747 Aircraft

### A.1 Flight Parameters

In this appendix three different views of the Boeing 747 aircraft are used to visualize the relevant aircraft flight parameters. These flight parameters include the attitudes, which are defined with respect to a body-fixed and an earth-fixed reference frame. The body-fixed reference frame consists of three axes  $X_B$ ,  $Y_B$ , and  $Z_B$  that are oriented in a fixed manner with respect to the aircraft. This means that the body-fixed reference frame rotates in a similar fashion as the aircraft. The origin of this reference frame is the center of gravity of the aircraft. The earth-fixed reference frame is fixed with respect to the earth and it can therefore rotate with respect to the aircraft. The earth-fixed reference frame also has its origin in the center of gravity of the aircraft and consists of three axes:  $X_V$ ,  $Y_V$ , and  $Z_V$ . The axis  $Z_V$  is pointed in the direction of the gravity vector,  $X_V$  is pointed to the north and  $Y_V$  is pointed 90 degrees to the right of the  $X_V$  axis. This reference frame is also known as the vehicle carried normal earth reference frame (Mulder et al. 2006). In Figures A.1, A.2, and A.3 the aircraft flight parameters listed in Table A.1 are depicted with respect to the two defined reference frames.

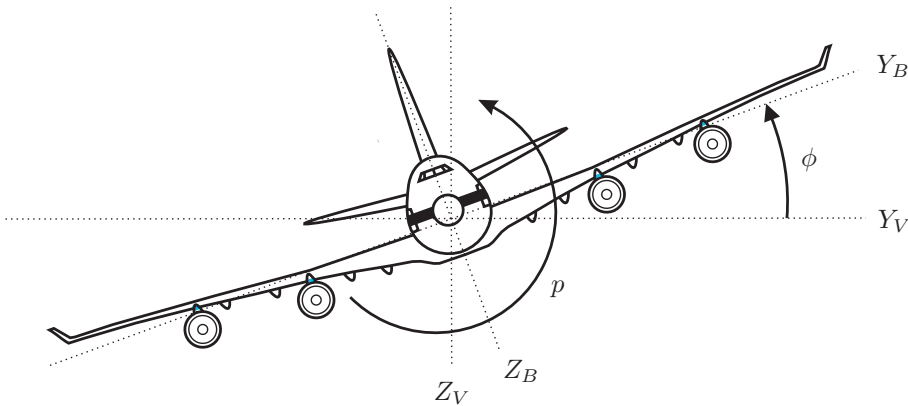


Figure A.1: Front view of the Boeing 747.

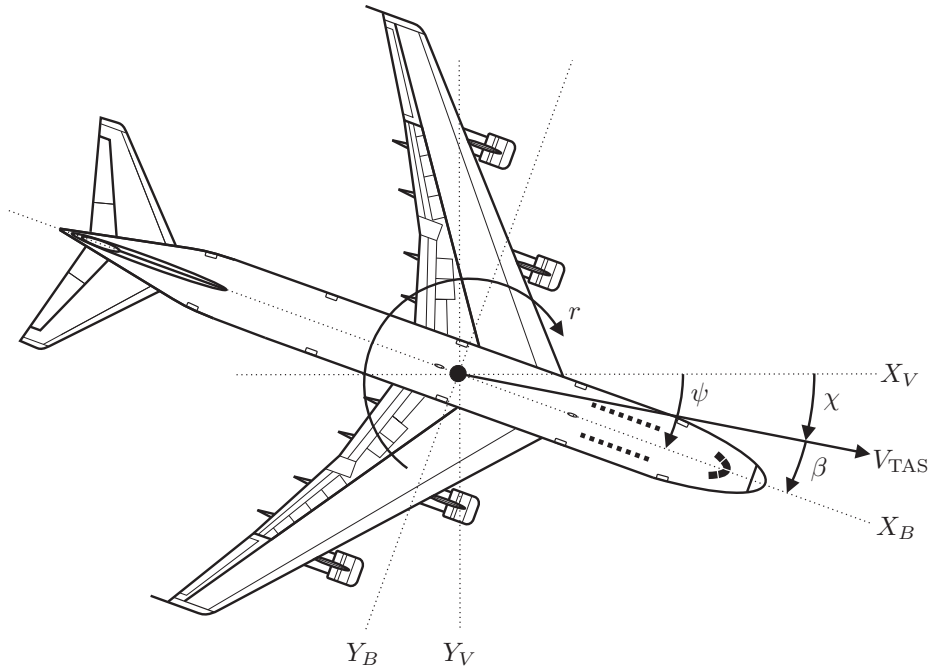


Figure A.2: Top view of the Boeing 747.

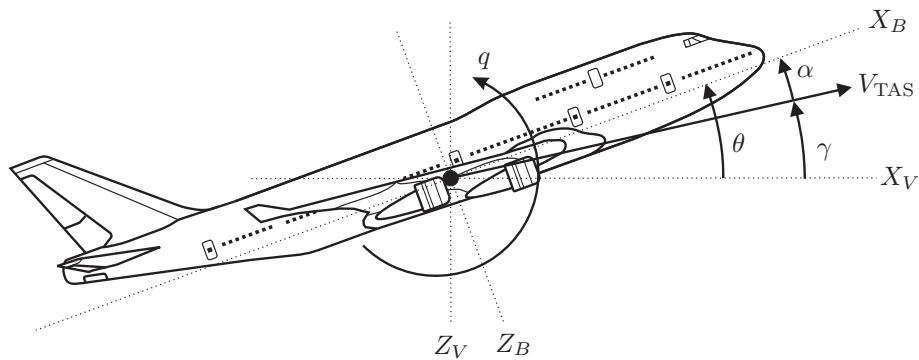


Figure A.3: Side view of the Boeing 747.

## A.2 Controls of the Boeing 747

The Boeing 747 has 29 different controls. These controls are listed in Table A.2 together with the mechanical limits and operation rates (Hanke and Nordwall (1970)). The locations of these controls on the aircraft are depicted in Figure A.4.

Table A.1: Aircraft flight parameters.

|           |                   |
|-----------|-------------------|
| $p$       | roll rate         |
| $q$       | pitch rate        |
| $r$       | yaw rate          |
| $V_{TAS}$ | true airspeed     |
| $\alpha$  | angle of attack   |
| $\beta$   | sideslip angle    |
| $\gamma$  | flight path angle |
| $\theta$  | pitch angle       |
| $\phi$    | roll angle        |
| $\psi$    | yaw angle         |
| $\chi$    | tracking angle    |

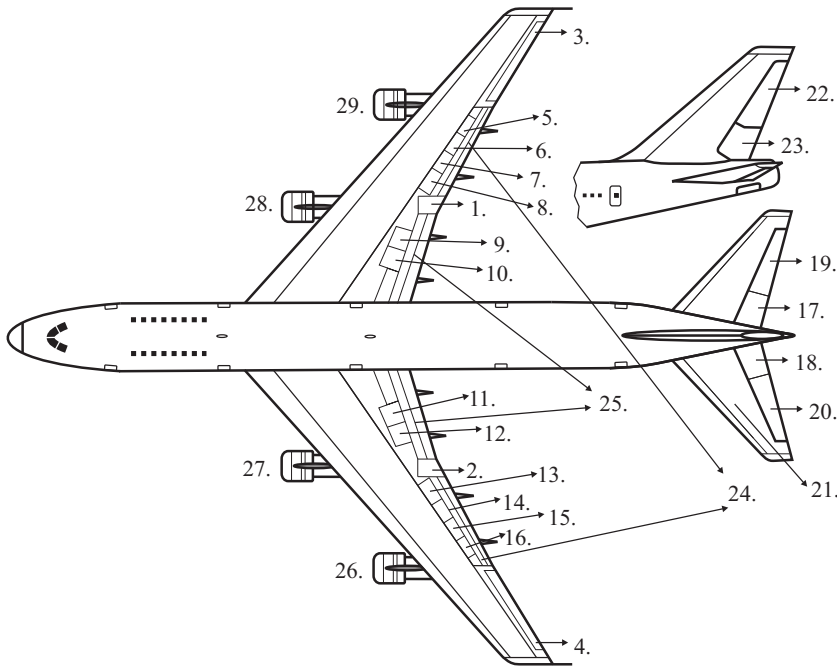


Figure A.4: Locations of controls on the Boeing 747.

Table A.2: Controls of the Boeing 747.

| Control no. | Description          | Mechanical limits | Operation rate    |
|-------------|----------------------|-------------------|-------------------|
| 1.          | Right inner aileron  | −20/20 deg        | 40/45 deg/s (↓/↑) |
| 2.          | Left inner aileron   | −20/20 deg        | 40/45 deg/s (↓/↑) |
| 3.          | Right outer aileron  | −25/25 deg        | 45/55 deg/s (↓/↑) |
| 4.          | Left outer aileron   | −25/25 deg        | 45/55 deg/s (↓/↑) |
| 5.          | Spoiler panel # 1    | 0/45 deg          | 75 deg/s          |
| 6.          | Spoiler panel # 2    | 0/45 deg          | 75 deg/s          |
| 7.          | Spoiler panel # 3    | 0/45 deg          | 75 deg/s          |
| 8.          | Spoiler panel # 4    | 0/45 deg          | 75 deg/s          |
| 9.          | Spoiler panel # 5    | 0/20 deg          | 75 deg/s          |
| 10.         | Spoiler panel # 6    | 0/20 deg          | 25 deg/s          |
| 11.         | Spoiler panel # 7    | 0/20 deg          | 25 deg/s          |
| 12.         | Spoiler panel # 8    | 0/20 deg          | 75 deg/s          |
| 13.         | Spoiler panel # 9    | 0/45 deg          | 75 deg/s          |
| 14.         | Spoiler panel # 10   | 0/45 deg          | 75 deg/s          |
| 15.         | Spoiler panel # 11   | 0/45 deg          | 75 deg/s          |
| 16.         | Spoiler panel # 12   | 0/45 deg          | 75 deg/s          |
| 17.         | Right inner elevator | −23/17 deg        | 37 deg/s          |
| 18.         | Left inner elevator  | −23/17 deg        | 37 deg/s          |
| 19.         | Right outer elevator | −23/17 deg        | 37 deg/s          |
| 20.         | Left outer elevator  | −23/17 deg        | 37 deg/s          |
| 21.         | Stabilizer           | −12/3 deg         | 0.2 to 0.5 deg/s  |
| 22.         | Upper rudder surface | −25/25 deg        | 50 deg/s          |
| 23.         | Lower rudder surface | −25/25 deg        | 50 deg/s          |
| 24.         | Outer flaps          | 0/25 deg          | 1.8 deg/s         |
| 25.         | Inner flaps          | 0/25 deg          | 1.8 deg/s         |
| 26.         | Thrust engine # 1    | 0.7/1.7 EPR       | -                 |
| 27.         | Thrust engine # 2    | 0.7/1.7 EPR       | -                 |
| 28.         | Thrust engine # 3    | 0.7/1.7 EPR       | -                 |
| 29.         | Thrust engine # 4    | 0.7/1.7 EPR       | -                 |

# B

## Discretization of a Multiple-Model System

This appendix describes how a multiple-model (MM) system can be discretized while preserving the linearity in the interpolations of the local models. Consider the continuous-time MM system given by

$$\dot{x}(t) = \sum_{i=1}^{N_m} \mu^{(i)}(t) [A^{(i)}x(t) + B^{(i)}u(t)] + w(t), \quad (\text{B.1})$$

$$y(t) = \sum_{i=1}^{N_m} \mu^{(i)}(t) [C^{(i)}x(t) + D^{(i)}u(t)] + v(t), \quad (\text{B.2})$$

where  $\mu^{(i)}(t)$  is the model weight corresponding to the  $i$ -th local model represented by the state space quadruple  $\{A^{(i)}, B^{(i)}, C^{(i)}, D^{(i)}\}$  and  $N_m$  is the number of local models in the model set. For application in an on-line environment a discrete-time version of this MM system is required. In the following the method is described that has been proposed by Hallouzi et al. (2006b) for discretization of an MM system. This method uses the Taylor expansion of the  $A$  and  $B$  matrices, which is given by

$$A_d = \sum_{k=0}^{\infty} \frac{A^k T^k}{k!}, \quad (\text{B.3})$$

$$B_d = \sum_{k=1}^{\infty} \frac{A^{k-1} B T^k}{k!}, \quad (\text{B.4})$$

where  $A_d$  and  $B_d$  are the discretized versions of the continuous-time system matrices  $A$  and  $B$ , respectively, and  $T$  is the sampling time. The  $C$  and  $D$  matrices are not changed by discretization and are therefore not considered. The matrices  $A_d$  and  $B_d$  can be approximated by truncating the Taylor expansion in (B.3)-(B.4). Such an approximation results in truncation errors which can be bounded in case of the  $A$  matrix as follows (Moler and Loan 2003)

$$|r_{i,j}| \leq \left( \frac{(\|A\|T)^{M+1}}{(M+1)!} \right) \left( \frac{1}{1 - \|A\|T/(M+2)} \right) \leq \epsilon, \quad (\text{B.5})$$

where  $M$  is the number of terms used for computing the Taylor expansion,  $\epsilon$  is a prescribed error tolerance and  $r_{i,j}$  represents the element in row  $i$  and column  $j$  of the remainder matrix  $R$ , which is defined as

$$R = \sum_{k=M+1}^{\infty} \frac{A^k T^k}{k!}. \quad (\text{B.6})$$

The matrix norm  $\|A\|$  used in (B.5) is defined as

$$\|A\| = \sum_{i,j=1}^n |a_{i,j}|, \quad (\text{B.7})$$

with  $A \in \mathbb{R}^{n \times n}$  and  $a_{i,j}$  represents the element of  $A$  in row  $i$  and column  $j$ . For matrix  $B$  a similar bound as (B.5) can be derived.

From (B.5) it is clear that  $\epsilon$  is proportional to the sample time  $T$ . So, a sufficiently small  $\epsilon$  can be achieved by using a first-order approximation ( $M=1$ ) in combination with a sufficiently small sample time  $T$ . Consider the continuous-time state equation (B.1), the first-order approximation of the discretized state equation is given by

$$x_{k+1} = Ix_k + \sum_{i=1}^{N_m} \mu_k^{(i)} [TA^{(i)}x_k + TB^{(i)}u_k] + w_k. \quad (\text{B.8})$$

If it holds that the model weights sum up to 1, which is the case if the local models correspond to physical models, then  $Ix_k$  can be substituted with  $\sum_{i=1}^{N_m} \mu_k^{(i)} x_k$  and (B.8) becomes

$$x_{k+1} = \sum_{i=1}^{N_m} \mu_k^{(i)} [\overbrace{(TA^{(i)} + I)}^{A_d^{(i)}} x_k + \overbrace{TB^{(i)}}^{B_d^{(i)}} u_k] + w_k. \quad (\text{B.9})$$

Now it can be seen that, for a first-order approximation and under the condition that the model weights sum up to 1, the discretized MM system is exactly the weighted sum of the individually first-order discretized local models. This property results in a very convenient discretization procedure for the continuous-time local models, because it preserves linearity of the interpolation of the local models.

The previously described method is also considered as a viable option to discretize LPV models by Tóth et al. (2008).



# C

## Recursive Update of Matrix $R$

This appendix describes how the  $R$ -matrix in an RQ-decomposition can be recursively updated. Let the matrix on the left-hand side of (6.17) on page 123 be represented by

$$\left[ \begin{array}{ccccc|c} \times & 0 & 0 & \cdots & 0 & \times \\ \times & \times & 0 & \cdots & 0 & \times \\ \vdots & \times & \times & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots \\ \times & \times & \times & \cdots & \times & \times \\ \hline \vdots & \vdots & \vdots & & \vdots & \vdots \\ \times & \times & \times & \cdots & \times & \times \end{array} \right] = \tilde{R}(t), \quad (\text{C.1})$$

where  $\times$  denotes the nonzero matrix entries. Updating the tall matrix from (C.1) can be performed by making it lower triangular. This can be done by performing a number of orthogonal transformations. The first transformation is performed by  $T_r$ , which rearranges the columns of this matrix such that the last column is placed between the first and second column

$$\tilde{R}(t) \xrightarrow{T_r} \left[ \begin{array}{cccccc|c} 1 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & \vdots & -1 & 0 & \cdots & 0 & 0 \\ 0 & \vdots & 0 & -1 & \ddots & \vdots & \vdots \\ \vdots & 0 & \vdots & \ddots & \ddots & 0 & 0 \\ \vdots & 0 & 0 & \cdots & 0 & -1 & \vdots \\ 0 & 1 & 0 & \cdots & \cdots & 0 & \times \end{array} \right] = \left[ \begin{array}{cccccc|c} \times & \boxed{\times} & 0 & 0 & \cdots & 0 & 0 \\ \times & \times & \boxed{\times} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \times & \boxed{\times} & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 & 0 \\ \times & \times & \times & \times & \cdots & \boxed{\times} & \vdots \\ \hline \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ \times & \times & \times & \times & \cdots & \times & \times \end{array} \right], \quad (\text{C.2})$$

where the entries that need to be annihilated are indicated by a box. An efficient way to annihilate matrix entries one at a time is to use Givens rotations (Golub and Van Loan 1996). Let  $[a \ b]$  be a given vector, with  $a$  and  $b$  being nonzero scalars. El-

ement  $b$  of this vector can then be annihilated with a Givens rotation  $G$  as follows

$$[a \ b] \overbrace{\begin{bmatrix} c & -s \\ s & c \end{bmatrix}}^{G^T} = [r \ 0], \quad (\text{C.3})$$

where  $r = \sqrt{a^2 + b^2}$ ,  $c = a/r$ ,  $s = b/r$  and  $G$  is an orthogonal matrix. Using the Givens rotation matrix  $G_1$ , the first entry of  $\tilde{R}(t)T_r$  can be annihilated as follows

$$\overbrace{\begin{bmatrix} \underline{\times} & \underline{\times} & 0 & 0 & \cdots & 0 \\ \underline{\times} & \underline{\times} & \times & 0 & \cdots & 0 \\ \vdots & \vdots & \times & \times & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ \underline{\times} & \underline{\times} & \times & \times & \cdots & \times \end{bmatrix}}^{\tilde{R}(t)T_r} \overbrace{\begin{bmatrix} c_1 & -s_1 & 0 & \cdots & 0 \\ s_1 & c_1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}}^{G_1} =$$

$$\begin{bmatrix} \times' & 0 & 0 & 0 & \cdots & 0 \\ \times' & \times' & \boxed{\times} & 0 & \cdots & 0 \\ \vdots & \vdots & \times & \boxed{\times} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ \underline{\times'} & \underline{\times'} & \times & \times & \cdots & \boxed{\times} \\ \hline \vdots & \vdots & \vdots & \vdots & & \vdots \\ \underline{\times'} & \underline{\times'} & \times & \times & \cdots & \times \end{bmatrix}, \quad (\text{C.4})$$

where  $c_1$  and  $s_1$  are computed using a Givens rotation applied to the two underlined entries of the leftmost matrix in (C.4), i.e.  $\tilde{R}(t)T_r$ . The resulting Givens rotation matrix  $G_1$  applies this Givens rotation only to the first two columns of  $\tilde{R}(t)T_r$ . In the right hand side of (C.4) it can be seen that one entry of  $\tilde{R}(t)T_r G_1$  has been annihilated and that the entries that are modified by  $G_1$  are indicated by a superscript  $'$ . The entries that need to be annihilated in further steps are again indicated by a box.

In the next step, the annihilation of the second entry can be performed as

$$\begin{array}{c}
 \overbrace{\left[ \begin{array}{ccccc|c}
 \times' & 0 & 0 & 0 & \dots & 0 \\
 \times' & \underline{\times'} & \underline{\times} & 0 & \dots & 0 \\
 \vdots & \vdots & \times & \times & \ddots & \vdots \\
 \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\
 \times' & \times' & \times & \times & \dots & \times
 \end{array} \right]}^{\tilde{R}(t)T_r G_1}
 \overbrace{\left[ \begin{array}{ccccc|c}
 1 & 0 & 0 & 0 & \dots & 0 \\
 0 & c_2 & -s_2 & 0 & \dots & 0 \\
 0 & s_2 & c_2 & 0 & \dots & 0 \\
 0 & 0 & 0 & 1 & \ddots & \vdots \\
 0 & \vdots & \vdots & \ddots & \ddots & 0 \\
 0 & 0 & 0 & \dots & 0 & 1
 \end{array} \right]}^{G_2}
 = \\
 \left[ \begin{array}{ccccc|c}
 \times' & 0 & 0 & 0 & \dots & 0 \\
 \times' & \times'' & 0 & 0 & \dots & 0 \\
 \vdots & \vdots & \times'' & \boxed{\times} & \ddots & \vdots \\
 \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\
 \times' & \times'' & \times'' & \times & \dots & \boxed{\times} \\
 \hline
 \vdots & \vdots & \vdots & \vdots & & \vdots \\
 \times' & \times'' & \times'' & \times & \dots & \times
 \end{array} \right], \quad (C.5)
 \end{array}$$

where, similarly to (C.4),  $c_2$  and  $s_2$  are computed using a Givens rotation applied to the two underlined entries of the leftmost matrix in (C.5), i.e.  $\tilde{R}(t)T_r G_1$ . The entries that are modified by  $G_2$ , i.e. all nonzero entries in the second and third column, are now indicated by a superscript  $''$  in the right-hand side of (C.5). The same sequence as described above, should be repeated until all entries to be annihilated, which are indicated by a box in (C.5), are indeed annihilated. In this way an orthogonal matrix  $\Omega = T_r G_1 G_2 \cdots G_{p(m+l)+1}$  can be obtained. This matrix can be used to obtain the updated matrix  $R(t) = \tilde{R}(t)\Omega$ .



# Bibliography

- Aggelogiannaki, E. and H. Sarimveis (2006). Multiobjective constrained MPC with simultaneous closed-loop identification. *International Journal of Adaptive Control and Signal Processing* **20**(4), 145–173.
- Alwi, H. and C. Edwards. (2008). Fault detection and fault-tolerant control of a civil aircraft using a sliding-mode-based scheme. *IEEE Transactions on Control Systems Technology*. DOI: 10.1109/TCST.2007.906311.
- Apkarian, P., P. Gahinet and G. Becker (1995). Self-scheduled  $\mathcal{H}_\infty$  control of linear parameter-varying systems: a design example. *Automatica* **31**(9), 1251–1261.
- Åström, K. J. and B. Wittenmark (1997). *Computer-Controlled Systems: Theory and Design*. Upper Saddle River, NJ, USA: Prentice Hall.
- Bar-Shalom, Y. and X. R. Li (1995). *Multitarget-Multisensor Tracking: Principles and Techniques*. Storrs, CT, USA: University of Connecticut Press.
- Bar-Shalom, Y., X. R. Li and T. Kirubarajan (2001). *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc.
- Bartys, M., R. Patton, M. Syfert, S. de las Heras and J. Quevedo (2006). Introduction to the DAMADICS actuator FDI benchmark study. *Control Engineering Practice* **14**(6), 577–596.
- Basseville, M. and V. Nikiforov (1993). *Detection of Abrupt Changes: Theory and Application*. Englewood Cliffs, NJ, USA: Prentice Hall.
- Beard, R. V. (1971). Failure accommodation in linear systems through self-reorganization. Technical report, Man Vehicle Laboratory, Cambridge, Massachusetts, USA. Rep. no. MVT-71-1.
- Belkharraz, A. I. and K. Sobel (2007). Simple adaptive control for aircraft control surface failures. *IEEE Transactions on Aerospace and Electronic Systems* **43**(2), 600–611.
- Blackmore, L. and B. Williams (2006). Finite horizon control design for optimal discrimination between several models. In *Proceedings of the IEEE Conference on Decision and Control*, San Diego, CA, USA (December).

- Blanke, M., M. Kinnaert, J. Lunze and M. Staroswiecki (2006). *Diagnosis and Fault-tolerant Control* (2<sup>nd</sup> ed.). Springer-Verlag Berlin Heidelberg.
- Blom, H. A. P. and Y. Bar-Shalom (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control* **33**(8), 780–783.
- Bodson, M. (2002). Evaluation of optimization methods for control allocation. *Journal of Guidance, Control, and Dynamics* **25**(4), 703–711.
- Bodson, M. and J. E. Groszkiewicz (1997). Multivariable adaptive algorithms for reconfigurable flight control. *IEEE Transactions on Control Systems Technology* **5**(2), 217–229.
- Bokor, J. and G. Balas (2004). Detection filter design for LPV systems - a geometric approach. *Automatica* **40**(3), 511–518.
- Bombois, X., G. Scorletti, M. Gevers, P. M. J. Van Den Hof and R. Hildebrand (2006). Least costly identification experiment for control. *Automatica* **42**(10), 1651–1662.
- Bos, R., X. Bombois and P. M. J. Van Den Hof (2005). Accelerating simulations of first principle models of complex industrial systems using quasi linear parameter varying models. In *Proceedings of the CDC-ECC, Seville, Spain* (December), pp. 4146–4151.
- Breeman, J. (2006). Quick start guide to AG 16 benchmark model. Technical report, NLR.
- Campbell, S. L. and R. Nikoukhah (2004). *Auxiliary signal design for failure detection*. Princeton, N. J.: Princeton University Press.
- Chan, T. F. (1984). Deflated decomposition of solutions of nearly singular systems. *SIAM Journal on Numerical Analysis* **21**(4), 738–754.
- Chan, T. F. and P. C. Hansen (1992). Some applications of the rank revealing QR factorization. *SIAM Journal on Scientific and Statistical Computing* **13**(3), 727–741.
- Chen, J. and R. J. Patton (1999). *Robust model-based fault diagnosis for dynamic systems*. Kluwer Academic Publishers.
- Chiuso, A. (2007). The role of vector autoregressive modeling in predictor-based subspace identification. *Automatica* **43**(6), 1034–1048.
- Chow, E. Y. and A. S. Willsky (1984). Analytical redundancy and the design of robust failure detection systems. *IEEE Transactions on Automatic Control* **29**(7), 603–614.
- Cieslak, J., D. Henry, A. Zolghadri and P. Goupil (2008). Development of an active fault-tolerant flight control strategy. *Journal of Guidance, Control, and Dynamics* **31**(1), 135–147.

- De Persis, C. and A. Isidori (2001). A geometric approach to nonlinear fault detection and isolation. *IEEE Transactions on Automatic Control* **46**(6), 853–865.
- Diao, Y. and K. M. Passino (2002). Intelligent fault-tolerant control using adaptive and learning methods. *Control Engineering Practice* **10**(8), 801–817.
- Dong, J., M. Verhaegen and E. Holweg (2008). Closed-loop subspace predictive control for fault tolerant MPC design. Seoul, Korea (July). *To be presented at the IFAC World Congress*.
- Doucet, A. and C. Andrieu (2001). Iterative algorithms for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing* **49**(6), 1216–1227.
- Doucet, A., N. J. Gordon and V. Krishnamurthy (2001). Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing* **49**(3), 613–624.
- Ducard, G. and H. P. Geering (2008). Efficient nonlinear actuator fault detection and isolation system for unmanned aerial vehicles. *Journal of Guidance, Control, and Dynamics* **31**(1), 225–237.
- Edwards, C., T. Lombaerts and H. Smaili (Eds.) (2008). *Fault Tolerant Control -A Benchmark Challenge*. Preliminary accepted by Springer-Verlag.
- Favoreel, W. (1999). *Subspace methods for identification and control of linear and bilinear systems*. Ph. D. thesis, Faculty of Engineering, K.U. Leuven, Belgium.
- Favoreel, W. and B. De Moor (1999). SPC: Subspace Predictive Control. In *Proceedings of the IFAC World Congress*, Beijing, China (July).
- Favoreel, W., B. De Moor, M. Gevers and P. Van Overschee (1999). Closed-loop model-free subspace-based LQG-design. In *Proceedings of the Mediterranean Conference on Control and Automation*, Haifa, Israel (June).
- Favoreel, W., B. De Moor and P. Van Overschee (1999). Model-free subspace-based LQG-design. In *Proceedings of the American Control Conference*, San Diego, CA, USA (June).
- Fekri, S., M. Athans and A. Pascoal (2006a). Issues, progress and new results in robust adaptive control. *International Journal of Adaptive Control and Signal Processing* **20**(10), 519–579.
- Fekri, S., M. Athans and A. Pascoal (2006b). A two-input two-output robust multiple model adaptive control (RMMAC) case study. In *Proceedings of the American Control Conference*, Minneapolis, MN, USA (June 14-16).
- Fielding, C., A. Varga, S. Bennani and M. Selier (Eds.) (2002). *Advanced Techniques for Clearance of Flight Control Laws*. Berlin Heidelberg: Springer.
- Fisher, K. A. and P. S. Maybeck (2002). Multiple model adaptive estimation with filter spawning. *IEEE Transactions on Aerospace and Electronic Systems* **38**(3), 755–768.

- Forssell, U. and L. Ljung (2000). Some results on optimal experiment design. *Automatica* **36**(5), 749–756.
- Frank, P. M. (1990). Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results. *Automatica* **26**(3), 459–474.
- Fujimori, A. and L. Ljung (2006). Model identification of linear parameter varying aircraft systems. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* **220**(4), 337–346.
- GARTEUR (2007). GARTEUR FM-AG16 final workshop. Available from Internet: [http://www.nlr.nl/documents/GARTEUR\\_AG16\\_Workshop](http://www.nlr.nl/documents/GARTEUR_AG16_Workshop). Last checked on December 12, 2007.
- Gáspár, P. and J. Bokor (2006). A fault-tolerant rollover prevention system based on an LPV method. *International Journal of Vehicle Design* **42**(3-4), 392–412.
- Gertler, J. (1997). Fault detection and isolation using parity relations. *Control Engineering Practice* **5**(5), 653–661.
- Gietelink, O. J., R. Hallouzi, J. Ploeg, B. De Schutter and M. Verhaegen (2007a). Cooperative driving using environment sensors and vehicle-to-vehicle communication. *Submitted to a Journal*.
- Gietelink, O. J., R. Hallouzi, B. De Schutter and M. Verhaegen (2007b). Fault management for automated longitudinal vehicle control. *Submitted to a Journal*.
- Golub, G. H. and C. F. Van Loan (1996). *Matrix Computations* (Third ed.). Baltimore, MD: The John Hopkins University Press.
- Gustafsson, F. (2000). *Adaptive Filtering and Change Detection*. John Wiley & Sons.
- Hajiyev, C. and F. Caliskan (2003). *Fault Diagnosis and Reconfiguration in Flight Control Systems*. Kluwer Academic Publishers.
- Hallouzi, R., V. Verdult, R. Babuška and M. Verhaegen (2005). Fault detection and identification of actuator faults using linear parameter varying models. In *Proceedings of the IFAC World Congress, Prague, Czech Rep.* (July).
- Hallouzi, R., V. Verdult, H. Hellendoorn, P.L.J. Morsink and J. Ploeg (2004a). Communication based longitudinal vehicle control using an extended Kalman filter. In *Proceedings of the IFAC Symposium on Advances in Automotive Control, Salerno, Italy* (April).
- Hallouzi, R., V. Verdult, H. Hellendoorn and J. Ploeg (2004b). Experimental evaluation of a co-operative driving set-up based on inter-vehicle communication. In *Proceedings of the IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon, Portugal* (July).



- Hallouzi, R. and M. Verhaegen (2007). Reconfigurable fault tolerant control of a Boeing 747 using subspace predictive control. In *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit*, Hilton Head, SC, USA (August).
- Hallouzi, R. and M. Verhaegen (2008a). Fault-tolerant subspace predictive control applied to a Boeing 747 model. *Journal of Guidance, Control, and Dynamics*. DOI: 10.2514/1.33256.
- Hallouzi, R. and M. Verhaegen (2008b). Persistency of excitation in subspace predictive control. Seoul, Korea (July). *To be presented at the IFAC World Congress*.
- Hallouzi, R., M. Verhaegen, R. Babuška and S. Kanev (2006a). Model weight and state estimation for multiple model systems applied to fault detection and identification. In *Proceedings of the IFAC Symposium on System Identification*, Newcastle, Australia (March).
- Hallouzi, R., M. Verhaegen and S. Kanev (2006b). Model weight estimation for FDI using convex fault models. In *Proceedings of the IFAC SAFEPROCESS*, Beijing, China (August).
- Hallouzi, R., M. Verhaegen and S. Kanev (2008). Multiple model estimation: a convex model formulation. *International Journal of Adaptive Control and Signal Processing*. DOI: 10.1002/acs.1034.
- Hanke, C. R. and D. R. Nordwall (1970). The simulation of a jumbo jet transport aircraft volume II: modeling data. Technical report, The Boeing Company.
- Härkegård, O. (2004). Dynamic control allocation using constrained quadratic programming. *Journal of Guidance, Control, and Dynamics* **27**(6), 1028–1034.
- Isermann, R. (1984). Process fault detection based on modeling and estimation methods - a survey. *Automatica* **20**(4), 387–404.
- Isermann, R. (2006). *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer-Verlag Berlin Heidelberg.
- Isidori, A. (1995). *Nonlinear Control Systems* (3<sup>rd</sup> ed.). Springer-Verlag London.
- Jansson, M. (2005). A new subspace identification method for open and closed loop data. In *Proceedings of the IFAC World Congress*, Prague, Czech Republic (July).
- Jiang, J. (2005). Fault-tolerant control systems - an introductory overview. *Acta Automatica Sinica* **31**(1), 161–174.
- Jilkov, V. P. and X. R. Li (2004). Online Bayesian estimation of transition probabilities for Markovian jump systems. *IEEE Transactions on Signal Processing* **52**(6), 1620–1630.

- Johnston, L. A. and V. Krishnamurthy (2001). An improvement to the interacting multiple model (IMM) algorithm. *IEEE Transactions on Signal Processing* **49**(12), 2909–2923.
- Jones, H. L. (1973). *Failure detection in linear systems*. Ph. D. thesis, Dept. of Aeronautics and Astronautics, MIT, Cambridge, Massachusetts, USA.
- Kadali, R., B. Huang and A. Rossiter (2003). A data driven subspace approach to predictive controller design. *Control Engineering Practice* **11**(3), 261–278.
- Kale, M. M. and A. J. Chipperfield (2005). Stabilized MPC formulations for robust reconfigurable flight control. *Control Engineering Practice* **13**(6), 771–788.
- Kanev, S. (2004). *Robust Fault-Tolerant Control*. Ph. D. thesis, University of Twente.
- Kanev, S. (2006). Polytopic model set generation for fault detection and diagnosis. Technical Report 06-016, Delft University of Technology, Delft Center for Systems and Control.
- Kanev, S. and M. Verhaegen (2000). Controller reconfiguration for non-linear systems. *Control Engineering Practice* **8**(11), 1223–1235.
- Kanev, S. and M. Verhaegen (2005). Two-stage Kalman filtering via structured square-root. *Journal of Communications in Informations and Systems* **5**(1), 143–168.
- Keviczky, T. and G. J. Balas (2006). Software-enabled receding horizon control for autonomous unmanned aerial vehicle guidance. *Journal of Guidance, Control, and Dynamics* **29**(3), 680–694.
- Kinnaert, M. (2003). Fault diagnosis based on analytical models for linear and nonlinear systems - a tutorial. In *IFAC Proceedings of SAFEPROCESS*, Washington, DC, USA (June).
- Kothare, M. V., V. Balakrishnan and M. Morari (1996). Robust constrained model predictive control using linear matrix inequalities. *Automatica* **32**(10), 1361–1379.
- Li, X. R. and V. P. Jilkov (2005). Survey of maneuvering target tracking. Part V: multiple-model methods. *IEEE Transactions on Aerospace and Electronic Systems* **41**(4), 1255–1321.
- Li, X. R., Z. Zhao and X-B. Li (2005). General model-set design methods for multiple-model approach. *IEEE Transactions on Automatic Control* **50**(9), 1260–1276.
- Liao, F., J. L. Wang, E. K. Poh and D. Li (2005). Fault-tolerant robust automatic landing control design. *Journal of guidance, control and dynamics* **28**(5), 854–871.
- Liao, F., J. L. Wang and G. H. Yang (2002). Reliable robust flight tracking control: an LMI approach. *IEEE Transactions on Control Systems Technology* **10**(1), 76–89.

- Ljung, L. (1979). Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. *IEEE Transactions on Automatic Control* **24**(1), 36–50.
- Ljung, L. and T. McKelvey (1996). Subspace identification from closed loop data. *Signal Processing* **52**(2), 209–215.
- Logothetis, A. and V. Krishnamurthy (1999). Expectation maximization algorithms for MAP estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing* **47**(8), 2139–2156.
- Lombaerts, T. J. J., Q. P. Chu, J. A. Mulder and D. A. Joosten (2007). Real time damaged aircraft model identification for reconfiguring flight control. In *Proceedings of the AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Hilton Head, SC, USA (August).
- Lovera, M., T. Gustafsson and M. Verhaegen (2000). Recursive subspace identification of linear and non-linear Wiener state-space models. *Automatica* **36**, 1639–1650.
- Maciejowski, J. M. (2002). *Predictive Control with Constraints*. Prentice Hall.
- Maciejowski, J. M. and C. N. Jones (2003). MPC fault-tolerant flight control case study: flight 1862. In *Proceedings of IFAC SAFEPROCESS*, Washington D.C., USA (June).
- Marcos, A. and G. J. Balas (2003). A Boeing 747-100/200 aircraft fault tolerant and fault diagnostic benchmark. Technical Report AEM-UoM-2003-1, University of Minnesota.
- Marcos, A. and G. J. Balas (2004). Development of linear-parameter-varying models for aircraft. *Journal of Guidance, Control and Dynamics* **27**(2), 218–228.
- Marcos, A., S. Ganguli and G. J. Balas (2005). Application of  $\mathcal{H}_\infty$  fault detection and isolation to a transport aircraft. *Control Engineering Practice* **13**(1), 105–119.
- Maybeck, P. S. (1999). Multiple model adaptive algorithms for detecting and compensating sensor and actuator/surface failures in aircraft flight control systems. *International Journal of Robust and Nonlinear Control* **9**(14), 1051–1070.
- Moler, C. and C. Van Loan (2003). Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review* **45**(1), 3–49.
- Mulder, J. A., W. H. J. J. van Staveren, J. C. van der Vaart and E. de Weerd (2006). Flight dynamics - lecture notes. Delft University of Technology.
- Murray-Smith, R. and T. A. Johansen (1997). *Multiple model approaches to modelling and control*. London, UK: Taylor and Francis.
- Narendra, K. S., O. A. Driollet, M. Feiler and K. George (2003). Adaptive control using multiple models, switching and tuning. *International Journal of Adaptive Control and Signal Processing* **17**(2), 87–102.

- Nelson, A. T. (2000). *Nonlinear estimation and modeling of noisy time-series by dual Kalman filtering methods*. Ph. D. thesis, Oregon Graduate Institute of Science and Technology.
- Nelson, A. T. and E. A. Wan (1998). A two-observation Kalman framework for maximum-likelihood modeling of noisy time series. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, Anchorage, AK, USA.
- Ni, L. and C. R. Fuller (2003). Control reconfiguration based on hierarchical fault detection and identification for unmanned underwater vehicles. *Journal of Vibration and Control* **9**, 735–748.
- Nichols, R. A., R. T. Reichert and W. J. Rugh (1993). Gain scheduling for  $H$ -infinity controllers: a flight control example. *IEEE Transactions on Control Systems Technology* **1**(2), 69–79.
- Niemann, H. and J. Stoustrup (2005). Passive fault tolerant control of a double inverted pendulum - case study. *Control Engineering Practice* **13**(8), 1047–1059.
- Nijmeijer, H. (1982). Observability of autonomous discrete-time nonlinear systems: a geometric approach. *International Journal of Control* **36**(5), 867–874.
- NRC (2007). Profiel: Bijlmerramp. Available from Internet: <http://www.nrc.nl/W2/Lab/Profiel/Bijlmerramp/>. Last checked on December 28, 2007.
- NTSB (1999). Aircraft accident report - US Air flight 427. Technical Report NTSB-AAR-99-01, National Transportation Safety Board.
- NTSB (2001). Aircraft accident report - United Airlines flight 585. Technical Report NTSB-AAR-01-01, National Transportation Safety Board.
- Oosterom, M. (2005). *Soft computing methods in flight control system design*. Ph. D. thesis, Delft University of Technology.
- Oosterom, M., R. Babuška and H. B. Verbruggen (2002). Soft computing applications in aircraft sensor management and flight control law reconfiguration. *IEEE Transactions on Systems, Man and Cybernetics* **32**(2), 125–139.
- Pachter, M. and Y-S. Huang (2003). Fault tolerant flight control. *Journal of Guidance, Control, and Dynamics* **26**(1), 151–160.
- Prakash, J., S. Narasimhan and S. C. Patwardhan (2005). Integrating model based fault diagnosis with model predictive control. *Industrial & Engineering Chemistry Research* **44**(12), 4344–4360.
- Rodrigues, M., D. Theilliol, M. Adam-Medina and D. Sauter (2008). A fault detection and isolation scheme for industrial systems based on multiple operating models. *Control Engineering Practice* **16**(2), 225–239.

- Rodrigues, M., D. Theilliol and D. Sauter (2005). Design of an active fault tolerant control and polytopic unknown input observer for systems described by a multi-model representation. In *Proceedings of the IEEE CDC-ECC*, Seville, Spain (December).
- Ru, J. and X. R. Li (2003). Interacting multiple model algorithm with maximum likelihood estimation for FDI. In *Proceedings of the IEEE International Symposium on Intelligent Control*, Houston, TX, USA (October).
- Ru, J. and X. R. Li (2008). Variable-structure multiple-model approach to fault detection, identification, and estimation. *To appear in IEEE Transactions on Control Systems Technology*.
- Seattle Times (2007). Safety at issue: the 737. Available from Internet: <http://seattletimes.nwsources.com/news/local/737/>. Last checked on September 27, 2007.
- Seguchi, H. and T. Ohtsuka (2003). Nonlinear receding horizon control of an underactuated hovercraft. *International Journal of Robust and Nonlinear Control* **13**(3-4), 381–398.
- Setnes, M. and R. Babuška (2001). Rule base reduction: some comments on the use of orthogonal transforms. *IEEE Transactions on Systems, Man and Cybernetics-Part C* **31**(2), 199–206.
- Shamma, J. S. and M. Athans (1991). Guaranteed properties of gain scheduled control for linear parameter varying plants. *Automatica* **27**(3), 559–564.
- Shin, J-Y. and C. M. Belcastro (2006). Performance analysis on fault tolerant control system. *IEEE Transactions on Control Systems Technology* **14**(5), 920–925.
- Shorten, R., R. Murray-Smith, R. Bjørgan and H. Gollee (1999). On the interpretation of local models in blended multiple model structures. *International Journal of Control* **72**(7/8), 620–628.
- Shouche, M., H. Genceli, P. Vuthandam and M. Nikolaou (1998). Simultaneous constrained model predictive control and identification of DARX processes. *Automatica* **34**(12), 1521–1530.
- Silva, P. M., V. M. Becerra, I. Khoo and J. M. F. Calado (2007). Hybrid fault detection and isolation in an heater bank of an HVAC system. In *Proceedings of the Mediterranean Conference on Control and Automation*, Athens, Greece (July).
- Simani, S., C. Fantuzzi and R. J. Patton (2003). *Model-based fault diagnosis in dynamic systems using identification techniques*. Advances in Industrial Control. Springer-Verlag London Ltd.
- Simon, D. and T. L. Chia (2002). Kalman filtering with state equality constraints. *IEEE Transactions on Aerospace and Electronic Systems* **31**(1), 128–136.

- Simon, D. and D. L. Simon (2006). Kalman filtering with inequality constraints for turbofan engine health estimation. *IEE Proceedings - Control Theory and Applications* **153**(3), 371–378.
- SIMONA (2007). TU Delft - SIMONA research simulator. Available from Internet: <http://www.simona.tudelft.nl/>. Last checked on October 8, 2007.
- Smaili, M. H. (1999). Flightlab 747 benchmark for advanced flight control engineering v4.03. Technical report, Delft University of Technology.
- Smaili, M. H., J. Breeman, T. J. J. Lombaerts and D. A. Joosten (2006). A simulation benchmark for integrated fault tolerant flight control evaluation. In *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit*, Keystone, CO, USA (August).
- Smaili, M. H. and J. A. Mulder (2000). Flight data reconstruction and simulation of the 1992 Amsterdam Bijlmermeer airplane accident. In *AIAA Modelling and Simulation Technologies Conference and Exhibit*, Denver, Colorado USA (August).
- Song, Y., G. Campa, M. Napolitano, B. Seanor and M. G. Perhinschi (2002). On-line parameter estimation techniques comparison within a fault tolerant flight control system. *Journal of Guidance, Control, and Dynamics* **25**(3), 528–537.
- Stilwell, D. J. (2001). State-space interpolation for a gain-scheduled autopilot. *Journal of Guidance, Control, and Dynamics* **24**(3), 460–465.
- Stoustrup, J. and V. D. Blondel (2004). Fault tolerant control: A simultaneous stabilization result. *IEEE Transactions on Automatic Control* **49**(4), 305–310.
- Szászi, I., A. Marcos, G. J. Balas and J. Bokor (2005). Linear parameter-varying detection filter design for a Boeing 747-100/200 aircraft. *Journal of Guidance, Control, and Dynamics* **28**(3), 461–470.
- Tóth, R., F. Felici, P. S. C. Heuberger and P. M. J. Van den Hof (2008). LPV state-space discretization. Seoul, Korea (July). *To be presented at the IFAC World Congress*.
- Uppal, F. J. and R. J. Patton (2005). Neuro-fuzzy uncertainty de-coupling: a multiple-model paradigm for fault detection and isolation. *International Journal of Adaptive Control and Signal Processing* **19**(4), 281–304.
- Van Der Linden, C. A. A. M. (1998). *DASMAT - Delft University Aircraft Simulation Model and Analysis Tool*. Delft University Press.
- Van Overschee, P. and B. De Moor (1996). *Subspace identification for linear systems: theory, implementation, applications*. Kluwer Academic Publishers.
- Van Paassen, M. M., O. Stroosma and J. Delatour (2000). DUECA - data-driven activation in distributed real-time computation. In *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit*, Denver, CO, USA (August).



- Varga, A. (2007). Fault detection and isolation of actuator failures for a large transport aircraft. In *Proceedings of the CEAS European Air and Space Conference*, Berlin, Germany (September).
- Venkatasubramanian, V., R. Rengaswamy, K. Yin and S. N. Kavuri (2003). Review of process fault diagnosis - Part I: quantitative model-based methods. *Computers and Chemical Engineering* **27**(3), 293–311.
- Verdult, V., L. Ljung and M. Verhaegen (2002). Identification of composite local linear state-space models using a projected gradient search. *International Journal of Control* **75**(16/17), 1385–1398.
- Verdult, V., J. A. K. Suykens, J. Boets, I. Goethals and B. de Moor (2004). Least squares support vector machines for kernel CCA in nonlinear state-space identification. In *Proceedings of the International Symposium on Mathematical Theory of Networks and Systems*, Leuven, Belgium (July).
- Verhaegen, M. and P. Dewilde (1992). Subspace identification, part I: The output-error state space model identification class of algorithms. *International Journal of Control* **56**(5), 1187–1210.
- Verhaegen, M., R. Hallouzi and S. Kanev (2006). Fault detection and identification using multiple models. In *Proceedings of the Portuguese Conference on Automatic Control (CONTROLO)*, Lisbon, Portugal (September).
- Verhaegen, M. and V. Verdult (2007). *Filtering and System Identification: An Introduction*. Cambridge University Press.
- Verhaegen, M. H. (1989). Round-off error propagation in four generally-applicable, recursive, least-squares estimation schemes. *Automatica* **25**(3), 437–444.
- Verma, V., G. Gordon, R. Simmons and S. Thrun (2004). Real-time fault diagnosis. *IEEE Robotics and Automation Magazine* **11**(2), 56–66.
- Wan, E. A. and A. T. Nelson (2001). *Dual Extended Kalman Filter Methods*. John Wiley & Sons.
- Wang, F. and V. Balakrishnan (2002). Improved stability analysis and gain-scheduled controller synthesis for parameter-dependent systems. *IEEE Transactions on Automatic Control* **47**(5), 720–734.
- Wang, X., B. Huang and T. Chen (2007). Data-driven predictive control for solid oxide fuel cells. *Journal of Process Control* **17**(2), 103–114.
- Woodley, B. R., J. P. How and R. L. Kosut (2001). Subspace based direct adaptive  $\mathcal{H}_\infty$  control. *International Journal of Adaptive Control and Signal Processing* **15**, 535–561.
- Yen, G. G. and L-W. Ho (2003). Online multiple-model-based fault diagnosis and accommodation. *IEEE Transactions on Industrial Electronics* **50**(2), 296–312.

- Yen, J. and L. Wang (1999). Simplifying fuzzy rule-based models using orthogonal transformation methods. *IEEE Transactions on Systems, Man and Cybernetics-Part B* **29**(1), 13–24.
- Zhang, D., Z. Wang and S. Hu (2007). Robust satisfactory fault-tolerant control of uncertain linear discrete-time systems: an LMI approach. *International Journal of Systems Science* **38**(2), 151–165.
- Zhang, X., T. Parisini and M. M. Polycarpou (2005). Sensor bias fault isolation in a class of nonlinear systems. *IEEE Transactions on Automatic Control* **50**(3), 370–376.
- Zhang, Y. and J. Jiang (2001). Integrated active fault-tolerant control using IMM approach. *IEEE Transactions on Aerospace and Electronic Systems* **37**(4), 1221–1235.
- Zhang, Y. and J. Jiang (2006). Issues on integration of fault diagnosis and reconfigurable control in active fault-tolerant control systems. In *Proceedings of the IFAC SAFEPROCESS*, Beijing, China (August).
- Zhang, Y. and X. Rong Li (1998). Detection and diagnosis of sensor and actuator failures using IMM estimator. *IEEE Transactions on Aerospace and Electronic Systems* **34**(4), 1293–1313.



# List of Abbreviations

|         |                                                          |
|---------|----------------------------------------------------------|
| AG-16   | Action Group 16                                          |
| CDF     | Cumulative Distribution Function                         |
| CM      | Convex Model                                             |
| CMF     | Convex Model Filter                                      |
| CP      | Convex Polytope                                          |
| EKF     | Extended Kalman Filter                                   |
| FD      | Fault Diagnosis                                          |
| FDI     | Fault Detection and Isolation                            |
| FTC     | Fault-Tolerant Control                                   |
| GARTEUR | Group for Aeronautical Research and Technology in EUROpe |
| IMM     | Interacting Multiple-Model                               |
| JMLS    | Jump Markov Linear System                                |
| LPV     | Linear Parameter-Varying                                 |
| LV      | Limit Values                                             |
| MAP     | Maximum a Posteriori                                     |
| MM      | Multiple-Model                                           |
| MMSE    | Minimum Mean Square Error                                |
| MPC     | Model Predictive Control                                 |
| MSD     | Mass-Spring-Damper                                       |
| OD      | Orthogonal Decomposition                                 |
| PDF     | Probability Density Function                             |
| PE      | Persistency of Excitation                                |
| QP      | Quadratic Programming                                    |
| RMSE    | Root Mean Square Error                                   |
| RRQR    | Rank-Revealing QR (decomposition)                        |

|      |                               |
|------|-------------------------------|
| SNR  | Signal to Noise Ratio         |
| SPC  | Subspace Predictive Control   |
| SRCF | Square Root Covariance Filter |
| SVD  | Singular Value Decomposition  |
| VAF  | Variance Accounted For        |

# Summary

## Multiple-Model Based Diagnosis for Adaptive Fault-Tolerant Control

Redouane Hallouzi

Safety and reliability of control systems is becoming increasingly important in present-day society. This is caused by the large presence of control systems in many products that are used in everyday life. When faults occur in such control systems this has consequences for their operation. Especially in safety-critical systems such as, for example, aircraft, consequences of faults can be catastrophic. In order to maintain an acceptable level of performance of control systems in case of faults, these faults have to be diagnosed and accommodated. This thesis deals with the development of methods for fault diagnosis and the accommodation of faults in control systems.

Fault diagnosis is concerned with detection of faults, the determination of which components are affected by faults, which is referred to as fault isolation, and the determination of the size of the faults. A straightforward approach to diagnose faults is to use multiple redundant hardware components for the same purpose. This approach, which is based on the principle of hardware redundancy, allows faults to be diagnosed by comparing the operation of the redundant hardware components with each other. Drawbacks of this approach are that it increases the overall cost of control systems and that it adds to the weight and maintenance requirements. Analytical redundancy is an alternative to hardware redundancy that overcomes these drawbacks. Analytical redundancy is based on mathematical models of the system rather than redundant hardware. Faults are diagnosed by comparing the expected system behavior from the mathematical model with the actual system behavior. This type of diagnosis is referred to as model-based diagnosis.

In this thesis, the multiple-model (MM) framework is used for fault diagnosis because of its flexible structure that allows modeling of a wide variety of fault conditions. An MM system consists of a model set that contains local models corresponding to different operating conditions, which also include fault conditions. Faults are diagnosed by estimating the model weights, which correspond to the validity of the local models. The mainstream of estimation algorithms for the MM framework is based on a hybrid MM structure. This structure has the property

that there is no interaction between the different local models. This property limits the ability of estimation algorithms to interpolate between local models, which is very desirable because it allows the use of smaller model sets. Therefore, an MM structure has been proposed that explicitly interpolates between local models. MM estimation algorithms have been developed for this structure. These algorithms are compared with a widely used MM estimation algorithm that uses the hybrid model structure. From this comparison it has been concluded that the proposed MM structure indeed has better interpolation properties, which allow for the use of relatively small model sets.

An important issue when using the MM framework is the design of model sets given a set of operating conditions to be modeled. Very little attention has been paid in literature to methods that address the structured design of model sets. Therefore, three methods have been proposed for this purpose, all of which have the objective to compactly represent the parameter space of the parameters that vary as a result of varying operating conditions. Two of these methods depend on the availability of an initial model set that contains a large number of models sampled at different operating conditions to be modeled. The first method reduces this large model set by applying orthogonal decompositions. The second method uses a convex polytope with a limited number of vertices that contains all models in the large model set. The vertices of the convex polytope are used to form the local models in the reduced model set. A third model set design method is presented that is based on the limit values of the varying parameters. The three methods have been evaluated on a linear parameter-varying model that also includes faults. The three methods have shown a comparable good estimation performance for the considered model. Since the three methods result in model sets with different properties, the application requirements determine which method is most suitable.

The model set design method based on orthogonal decompositions has also been used for generating model sets for the purpose of fault diagnosis of a detailed nonlinear model of a Boeing 747 aircraft. The considered faults include “lock-in-place” of control surfaces. These faults are characterized by the control surfaces getting stuck at a fixed position. The model sets designed for this type of faults consist of local models corresponding to conditions at which the control surfaces are stuck at their maximum or minimum deflection. The correctness of these model sets is confirmed by the presence of similar model sets in literature for the same fault type. Furthermore, it has been shown in simulation that the model sets are accurate enough to isolate different “lock-in-place” faults of the Boeing 747 model.

Once information about a fault has been obtained it can be used for accommodation of the fault. Control systems with the ability to accommodate faults are referred to as fault-tolerant control (FTC) systems. Most FTC systems require detailed fault information, which limits their practical applicability, especially in case of unanticipated faults. Therefore, in this thesis a data-driven FTC system is developed that has the ability to adapt to changing system conditions including faults. The data-driven FTC system relies on subspace predictive control (SPC) for its adaptation to the system. SPC consists of a subspace predictor integrated with

a predictive control law. The subspace predictor is identified using input-output data of the system and is continuously updated on-line with new data when it becomes available. In this way, the subspace predictor can adapt to changing system conditions. An issue that is related to SPC is that it has to be ensured that the input-output data contains sufficient information about the controlled system. For this purpose a persistency of excitation condition has been developed that ensures that the system is additionally excited only in those directions of the input space that are least excited. An important advantage of this excitation condition is that it can be implemented in a computationally efficient manner.

Another extension to SPC presented in this thesis is the application of a closed-loop subspace predictor instead of an open-loop one. Furthermore, an efficient recursive updating scheme has been developed for this closed-loop subspace predictor. The FTC system based on SPC has been applied to the Boeing 747 aircraft model. In order to react more quickly to anticipated faults, the fault information obtained from an MM fault isolation scheme is used to switch between different settings of the FTC system. Results have been presented of three simulations, each of which with a different fault condition of the aircraft model. Two simulations have been performed with anticipated fault conditions: “lock-in-place” of the elevator surfaces and runaway of the rudder surfaces to their maximum deflection. The third simulation has been performed with an unanticipated fault condition modeled after the EL AL aircraft that crashed into an apartment building in Amsterdam in 1992. This disaster is known as the “Bijlmerramp”. The “Bijlmerramp” fault condition includes the separation of both engines on the right wing of the aircraft. All three fault conditions have been accommodated well by the developed FTC system. It has been shown for all three fault conditions that the aircraft has been able to perform the elementary maneuvers required for a safe landing.

In addition to an off-line version of the SPC-based FTC system, an on-line version has been implemented in a real-time simulation environment. With this on-line version it is demonstrated that it is indeed possible to perform real-time data-driven adaptive control of a complex system such as the Boeing 747 benchmark model.



# Samenvatting

## Multi-Model Gebaseerde Diagnose voor Adaptief Fouttolerant Regelen

Redouane Hallouzi

Veiligheid en betrouwbaarheid van regelsystemen worden steeds belangrijker in de huidige maatschappij. Dit komt door de grote aanwezigheid van regelsystemen in veel producten die dagelijks worden gebruikt. Als er fouten optreden in dergelijke regelsystemen, dan heeft dit gevolgen voor het functioneren ervan. Vooral in veiligheidskritische systemen, zoals vliegtuigen, kunnen de gevolgen van fouten catastrofaal zijn. Om een acceptabel prestatieniveau van regelsystemen te kunnen handhaven in geval van fouten, moet er een diagnose worden gemaakt van deze fouten en moet er op passende wijze worden ingegrepen. In dit proefschrift zijn methoden ontwikkeld voor foutdiagnose en het omgaan met fouten in regelsystemen.

Het doel van foutdiagnose is om fouten te detecteren, te bepalen welke componenten worden beïnvloed door fouten, ook bekend als foutisolatie en te bepalen wat de grootte is van fouten. Een simpele aanpak om fouten te kunnen diagnostiseren maakt gebruik van redundante hardware componenten met dezelfde functionaliteit. Deze aanpak, die gebaseerd is op het principe van hardware redundantie, zorgt ervoor dat fouten kunnen worden gediagnosticeerd door het vergelijken van de functionaliteit van de redundante hardware componenten. Nadelen van deze aanpak zijn de toename van de totale kosten van regelsystemen en de toename van het gewicht en de onderhoudsvereisten. Analytische redundantie is een alternatief voor hardware redundantie dat de genoemde nadelen kan ondergaan. Analytische redundantie is gebaseerd op wiskundige modellen in plaats van op redundante hardware. Fouten kunnen worden gediagnosticeerd door het vergelijken van het door een wiskundig model verwachte systeemgedrag met het feitelijke gedrag. Dit type diagnose wordt ook wel modelgebaseerde diagnose genoemd.

In dit proefschrift is het multi-model (MM) raamwerk gebruikt voor foutdiagnose vanwege de flexibele structuur die het toelaat om een breed scala aan foutcondities te modelleren. Een MM systeem bestaat uit een modelset die lokale modellen bevat. Deze lokale modellen komen overeen met verschillende systeemcondities waaronder foutcondities. Foutdiagnose vindt plaats door het schatten

van de modelgewichten, die betrekking hebben op de validiteit van lokale modellen. De meeste schattingsalgoritmen voor het MM raamwerk maken gebruik van een hybride modelstructuur. Deze structuur heeft de eigenschap dat er geen onderlinge interactie is tussen de verschillende lokale modellen. Deze eigenschap beperkt de mogelijkheid van de schattingsalgoritmen om te interpoleren tussen de lokale modellen, hetgeen zeer wenselijk is omdat dat het gebruik van kleinere modelsets toestaat. Daarom is een MM structuur voorgesteld die expliciet interpoleert tussen de lokale modellen. Voor deze structuur zijn schattingsalgoritmen ontwikkeld die vervolgens zijn vergeleken met een veelgebruikt MM schattingsalgoritme voor de hybride modelstructuur. Uit deze vergelijking is gebleken dat de voorgestelde MM structuur daadwerkelijk betere interpolatie eigenschappen heeft, die leiden tot het gebruik van kleinere modelsets.

Het ontwerpen van modelsets in het MM raamwerk, gegeven een set van systeemcondities die gemodelleerd moeten worden, is een belangrijk aandachtspunt. Tot nu toe is er weinig aandacht besteed in de literatuur aan gestructureerde ontwerpstechnieken voor modelsets. Daarom zijn er hiervoor drie technieken ontwikkeld met als doelstelling het compact representeren van de parameterruimte van de parameters die variëren als gevolg van variërende systeemcondities. Twee van deze technieken zijn gebaseerd op de beschikbaarheid van een initiële modelset die een groot aantal lokale modellen bevat, elk overeenkomend met verschillende systeemcondities die gemodelleerd dienen te worden. De eerste techniek maakt gebruik van orthogonale decomposities om de grote modelset te verkleinen. De tweede techniek maakt gebruik van een convexe polytoop met een beperkt aantal hoekpunten dat alle modellen uit de grote modelset omvat. De hoekpunten van de polytoop worden vervolgens gebruikt om de lokale modellen in de verkleinde modelset te vormen. Verder is er een derde techniek voor het ontwerpen van modelsets voorgesteld die gebruik maakt van de limietwaarden van de variërende parameters. De drie ontwerpstechnieken zijn met elkaar vergeleken op basis van een lineair parameter-variërend model waarin ook fouten gemodelleerd zijn. Alle drie technieken hebben geleid tot een vergelijkbare en goede schattingsprestatie. Omdat de eigenschappen van de modelsets verkregen met de drie verschillende technieken verschillend zijn, bepalen de eisen van de specifieke applicatie met betrekking tot deze eigenschappen welke methode het meest geschikt is.

De modelset ontwerpstechniek gebaseerd op orthogonale decomposities is ook gebruikt voor het genereren van modelsets voor foutdiagnose van een gedetailleerd niet-lineair model van een Boeing 747 vliegtuig. De beschouwde fouten omvatten "lock-in-place" fouten van stuurvlakken. Dit soort fouten wordt gekarakteriseerd door het vastzitten van de stuurvlakken. De modelsets ontworpen voor deze fouten bevatten lokale modellen die foutcondities representeren waarbij de betreffende stuurvlakken vast zitten op de maximale of minimale uitslag. De correctheid van deze modelsets wordt bevestigd door het feit dat "lock-in-place" fouten op dezelfde manier gemodelleerd zijn in de literatuur. Het is daarnaast aangetoond dat de ontworpen modelsets nauwkeurig genoeg zijn om verschillende "lock-in-place" fouten van het Boeing 747 simulatiemodel te isoleren.

Zodra een fout is gediagnosticeerd, kan de foutinformatie worden gebruikt om het regelsysteem aan te passen. Regelsystemen die op passende wijze met fou-



ten kunnen omgaan worden ook wel fouttolerante regelsystemen genoemd. De meeste fouttolerante regelsystemen vereisen gedetailleerde foutinformatie, wat de praktische bruikbaarheid beperkt. Dit is vooral het geval voor onverwachte fouten. Daarom is in dit proefschrift een datagedreven fouttolerant regelsysteem ontwikkeld dat zich kan aanpassen aan variërende systeemcondities waaronder fouten. Dit fouttolerante regelsysteem vertrouwt op *subspace predictive control* (SPC) voor de aanpassing aan variërende systeemcondities. SPC bestaat uit een *subspace* voorspeller die is geïntegreerd met een voorspellende regelaar. De *subspace* voorspeller is geïdentificeerd door gebruik te maken van ingang/uitgangsdata van het systeem. De voorspeller wordt continu *on-line* bijgewerkt met nieuwe ingang/uitgangsdata. Op deze manier kan de *subspace* voorspeller zich aanpassen aan variërende systeemcondities. Voor het gebruik van SPC is het noodzakelijk dat de ingang/uitgangsdata voldoende informatie over het geregelde systeem bevat. Daarom is er een excitatie conditie ontwikkeld die ervoor zorgt dat het systeem alleen wordt geëxciteerd in die richtingen van de ingangsruimte die het minst geëxciteerd zijn. Een belangrijk voordeel van de ontwikkelde excitatie conditie is dat deze op een efficiënte manier kan worden geïmplementeerd.

Een andere uitbreiding op SPC die beschreven is in dit proefschrift is het gebruik van een gesloten-lus *subspace* voorspeller in plaats van een open-lus voorspeller. Verder is er een efficiënt recursief algoritme ontwikkeld dat de gesloten-lus *subspace* voorspeller continu bijwerkt met nieuwe data. Het fouttolerante regelsysteem dat is gebaseerd op SPC is toegepast op het Boeing 747 vliegtuigmodel. Om sneller met verwachte fouten om te kunnen gaan, wordt foutinformatie, verkregen met behulp van MM foutisolatie, gebruikt om te schakelen tussen verschillende instellingen van het fouttolerante regelsysteem. Er zijn resultaten gepresenteerd van drie simulaties die elk met verschillende foutcondities uitgevoerd zijn. Twee simulaties zijn uitgevoerd met verwachte fouten: "lock-in-place" van de hoogteroeren en het weglopen naar de maximale uitslag van de richtingsroeren. De derde simulatie is uitgevoerd met een onverwachte fout die is gemodelleerd naar het EL AL vliegtuig dat zich in 1992 in een flat in Amsterdam boorde. Deze ramp staat ook wel bekend als de Bijlmerramp. De foutcondities van het Bijlmer-ramp vliegtuig omvatten het afvallen van de beide motoren aan de rechtervleugel. Uit de simulaties uitgevoerd met alle drie de beschouwde foutcondities is gebleken dat het fouttolerante regelsysteem in staat is geweest om zich aan te passen aan de opgetreden fouten. In alle drie de simulaties is het vliegtuig in staat geweest om elementaire manoeuvres uit te voeren die vereist zijn voor een veilige landing.

Naast een *off-line* versie van de fouttolerante regelaar is er ook een *on-line* versie ontwikkeld voor een *real-time* simulatieomgeving. Met deze *on-line* versie is aangetoond dat het mogelijk is om in *real-time* een databasebaseerde adaptieve fouttolerante regeling toe te passen op een complex systeem zoals het Boeing 747 vliegtuigmodel.



# Curriculum Vitae

Redouane Hallouzi was born on September 20, 1980 in Beni Touzine, Morocco. In 1992 he started his secondary education ("Gymnasium") at the Mondriaan Lyceum in Amsterdam, The Netherlands, and finished it in 1998. From 1998 to 2003 he studied Electrical Engineering at the Delft University of Technology, where he graduated cum laude at the Control Systems Engineering department. His graduation project was carried out at TNO Automotive in Delft, The Netherlands, and resulted in the M.Sc. thesis entitled "Communication Based Longitudinal Vehicle Control using Extended Kalman Filtering".

In January 2004, he started working on his Ph.D. project at the Delft Center for Systems and Control of the Delft University of Technology. The main focus of his Ph.D. project was on the development of fault diagnosis and fault-tolerant control algorithms. During his Ph.D. project he contributed to action group 16 of the European GARTEUR project on fault-tolerant control of aircraft. Furthermore, he took graduate courses at the Dutch Institute of Systems and Control (DISC), for which he received the DISC certificate.

In 2007, he was awarded a valorisation grant from the Dutch Technology Foundation (STW). He is currently using this grant to explore the possibilities for commercial application of fault diagnosis and fault-tolerant control methods.

