Retrospective Cost Adaptive Control with Concurrent Closed-Loop Identification

by

Frantisek M. Sobolic

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in The University of Michigan
2017

Doctoral Committee:

Professor Dennis S. Bernstein, Chair
Assistant Professor Kira L. Barton
Dr. Robert J. Fuentes, Raytheon Company
Associate Professor Anouck R. Girard
Professor Ilya Kolmanovsky

Frantisek M. Sobolic

fsobolic@umich.edu

ORCID iD: 0000-0003-4231-9055

For my family

ACKNOWLEDGEMENTS

Many friends, family, and colleagues have helped me through my time in graduate school. My advisor, Professor Dennis Bernstein, has provided me with guidance, enthusiasm, and encouragement throughout my time here at the University of Michigan. His abundant patience and understanding has made my research goals and life goal a reality. I would also like to thank my committee members, Professor Anouck Girard, Professor Ilya Kolmanovsky, Professor Kira Barton, and Dr. Robert Fuentes for their time in reviewing my work and providing invaluable insights. The University is a special place because of them and all the work they are doing to improve not only the research but students minds as well.

Life during graduate school would not be the same without the people I have met along the way. I would like to thank my friends and colleagues who I have had the privilege of working and studying with: Yousaf Rahman, Ray Yu, Antai Xie, Ankit Goel, Ahmad Ansari, Rohit Gupta, Sweewarman Balachandran, Jinwoo Seok, Julie Fogarty, Alex Walsh, Ryan Caverly, Professor James Forbes, Hyeongjun Park, Hicham Alkandry.

I would also like to thank the Raytheon Company for the support they have given me. A special thanks to Dr. James Fisher, Dr. Robert Fuentes, Dr. Andrew Douglas, Mike Unger, Dr. Brett Ridgely, James DeBoer, Dr. Joe Peterson, Dr. Don Croft, and Lee Conger for all their help, patience, and guidance.

Finally, I would like to thank my family for their abundant love and support throughout my life. To my parents, Frank and Mary Anne, words can not express how much your love means to me. To my brothers James and Adam and my sister Elena, thank you for your encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Retrospective cost adaptive control (RCAC) is a discrete-time direct adaptive control algorithm for stabilization, command following, and disturbance rejection. In recent years, the controller gains are solved for via recursive least squares (RLS). This approach works well on systems that are linear time invariant but for systems that are linear time varying, we propose extensions to the RLS solution via the inclusion of either a multiplicative or additive covariance update. A novel algorithm called the innovations-based sliding window-Q optimization (ISW-QO) is developed that optimizes the additive process noise covariance matrix Q via the retrospective innovations of the Kalman filter. For the parameter estimation problem, we prove that, under reasonable assumptions, the value of Q that minimizes the cumulative innovations also minimizes the cumulative state-estimate error. This algorithm is compared to the adaptive Kalman filter and results show that the ISW-QO algorithm works as well if not better than the adaptive Kalman filter.

Next, RCAC is known to work on systems given minimal modeling information. The information needed by RCAC is the leading numerator coefficient and any nonminimim-phase (NMP) zeros of the plant transfer function. This information, which is normally needed a priori, is key in the development of the filter, known as the target model within the retrospective performance variable. A novel approach to alleviate the need for prior modeling of both the leading coefficient of the plant transfer function as well as its NMP zeros is developed. The extension to the RCAC algorithm is the use of concurrent optimization of both the target model and the controller coefficients. Concurrent optimization of the target model and controller

coefficients is a quadratic optimization problem in the target model and controller coefficients separately. However, this optimization problem is not convex as a joint function of both variables, and therefore nonconvex optimization methods are needed. We take advantage of the biquadratic structure of the cost function by applying an alternative convex search algorithm and compare it to a nonlinear optimization routine.

Finally, insights within RCAC that include intercalated injection between the controller numerator and the denominator unveil the workings of RCAC fitting a specific closed-loop transfer function to the target model. We exploit this interpretation by investigating several closed-loop identification architectures in order to extract this information for use in the target model. For illustration, RCAC with concurrent closed-loop identification is applied to a system whose dynamics are highly nonlinear with time-varying NMP zeros.

CHAPTER 1

Introduction

The purpose of implementing a controller on a dynamic system is to improve, or enhance performance with the addition of sensors and actuators. The sensors provide measurements within the system that allow the controller to drive the actuators to reach or maintain a desired state. A general diagram of a control system is shown in Figure 1.1. Due to the information gained through sensed signals, the controller is designed to actuate the dynamic system in a desired manner. Thus, the cycle of sensing and actuating, as shown in Figure 1.1, represents a closed-loop feedback control system.

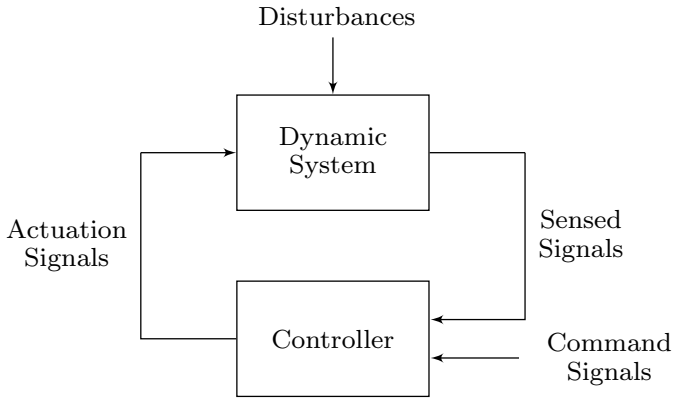


Figure 1.1: A generic diagram representing the control of a dynamic system.

Modern technologies such as aircraft, rockets, missiles, and spacecraft systems would not exist without feedback control. Methods to design controllers for these

technologies vary, but the objective is the same. A starting place for most control design is to identify and describe the system through understanding the governing equations of motion. Thus a large effort of the control design is spent investigating and understanding the system that is being controlled. Aircraft, rockets, and missiles, for example, undergo extensive wind tunnel experimentation in order to characterize the behavior during flight. Standard control techniques for these applications involve look-up tables that optimize the control design for a particular flight condition. The time and effort that is spent understanding and characterizing these dynamic systems is usually what drives the cost of the control design.

Many control designs are optimized for a certain system configuration. If the dynamic system were to change, for example adding an air-breathing engine to a missile that was originally a glider or changing a rocket's propellant which changes the mass properties, the control design may not work as well or may even cause the system to become unstable. An even worse case scenario is the dynamics changing in an unknown fashion while in operation. Examples of this include an aircraft that has aerodynamics that have changed due to icing on the wings or an actuator that becomes stuck. For some of these scenarios, robust control techniques are designed to deal with uncertainties so long as robust stabilization can be guaranteed for the prior uncertainty. A consequence of robust control techniques is a trade-off of performance for robustness and some of these uncertainties may not be known a priori. This motivates the desire to develop *adaptive* control techniques that are able to adjust to the uncertainties in the system.

Unlike robust control, adaptive control techniques seek to overcome the robustness/performance trade-off by adjusting the controller relative to the response of the system during closed-loop operation. Therefore, it is reasonable to think of an adaptive controller as a nonlinear controller that aims to overcome the robustness/performance trade-off due to uncertainty in the system dynamics. Figure 1.2

shows a generic diagram of an adaptive control technique. Based on the input and output of the dynamic system, the controller is tuned through an adaptive strategy to meet the desired performance objectives. The method for adjusting the controller in response to the changes in the system is what distinguishes one adaptive scheme from another.

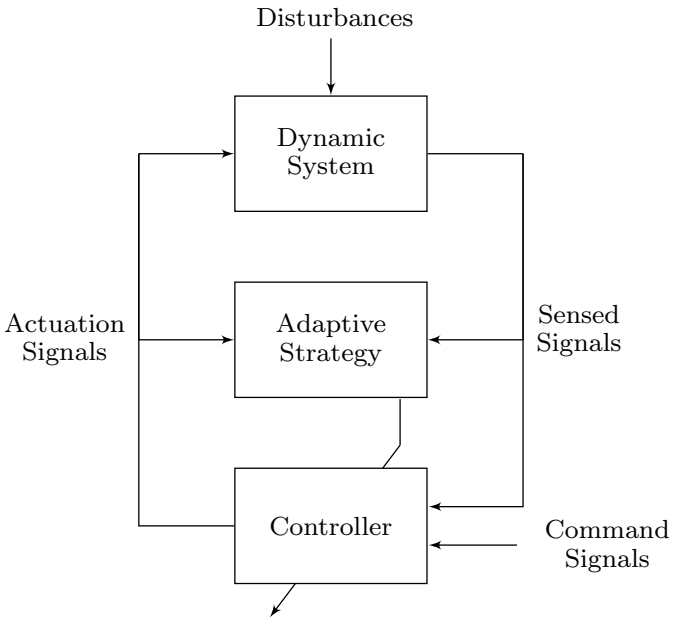


Figure 1.2: A generic diagram representing the adaptive control of a dynamic system.

The way an adaptive controller is implemented is broken down into two categories: *indirect* and *direct* adaptive control. In indirect adaptive control, the system parameters are estimated on-line and used to calculate the control parameters. The on-line estimated system is treated as if it is the true system in the calculation of the controller. In direct adaptive control, the estimated system is parameterized in terms of the desired controller. Therefore, the controller is estimated directly without any intermediate calculations. Throughout this dissertation, we focus on the direct adaptive controller and from here on out this is referred to as the adaptive controller.

The adaptive control literature focuses primarily on adaptive command following and adaptive disturbance rejection [1–5]. Model reference adaptive control (MRAC)

is one of the primary methods used for adaptive command following. The idea underlying MRAC is the design of a reference model that generates a desired trajectory which the system follows. Thus, the objective is to force an unknown system to follow the output of a known reference model [1, 2]. Adaptive disturbance rejection is a common objective among problems involving noise control and vibration suppression.

An underlying fundamental question in adaptive control is determining which modeling information is essential for achieving acceptable transient and asymptotic performance. Most adaptive control schemes are limited by certain assumptions that are made about the system. Some of the assumption include passivity, stably invertible or minimum phase, and matched uncertainties. These assumptions are made to prove convergence of the closed-loop system. The adaptive control algorithms presented in this dissertation do not assume any of these assumptions, but limited modeling information is required as discussed in Chapter 7.

In this dissertation, we consider a direct discrete-time adaptive control law. The discrete control is advantageous due to its ability to be implemented directly in embedded code without the need to perform an intermediate discretization step that potentially could cause a loss of phase margin. Specifically, we focus on the adaptive controller called retrospective cost adaptive control (RCAC). RCAC was developed in [6–8], and has been demonstrated in both simulation and laboratory experiments. The underlining concept of RCAC is the idea of re-optimizing the controller based on a retrospective view of the actual control inputs that went into the system and the actual performance that is represented within the *retrospective performance variable*. In other words, given the actual control and actual performance from the past, could a different controller have given better performance?

Optimization of the RCAC retrospective performance variable is one of the key features. This retrospective performance variable is a function of the performance variable and the control input. The control input is filtered through what we call the

target model G_{f}, whose design requires the limited modeling information from the system. The optimization step can be setup a number of different ways and more recently, in [8], a recursive least squares (RLS) algorithm is used. RLS has been shown to work well on linear unknown systems [7, 8] but suffers from lack of re-adaptation once the covariance matrix has decreased. In this dissertation, we explore the use of extending the RLS algorithm on systems where the dynamics are changing by introducing multiplicative and additive covariance matrix updates in Chapter 3.

RCAC has been shown to be effective on nonminimum-phase (NMP) systems. The NMP zero location (if any) is part of the modeling information that is needed in the RCAC target model G_{f} along with the relative degree, and the leading numerator coefficient of the system. In order to obtain this information from the system, we use closed-loop identification techniques to identify the system and extract the information needed for the target model. The use of concurrent identification and controller optimization allows the RCAC algorithm to obtain the most accurate target model G_{f} at each time-step.

The remainder of the introduction summarizes the contents within the chapters of this dissertation. Specifically, we outline the specific contributions of each chapter.

1.1 Relevant Publications

The following is a list of publications relevant to the research presented in this dissertation.

- F. Sobolic and D. S. Bernstein, "Semi-adaptive Control of a Nonlinear Missile Using Concurrent Closed-Loop Identification," *J. Guid. Dyn. Contr.* (submitted).

- F. Sobolic, K. F. Aljanaideh, and D. S. Bernstein, "Direct and Indirect Closed-Loop Architectures for Estimating Nonminimum-Phase Zeros," *Circ. Sys. Sig.*

Proc. (submitted)

- F. Sobolic and D. S. Bernstein, "Retrospective Cost Adaptive Control with Concurrent Closed-Loop Identification of Time-Varying Nonminimum-Phase Zeros," in *Proc. Conf. Dec. Contr.*, Las Vegas, NV, Dec 2016, pp. 371–376.

- F. Sobolic and D. S. Bernstein, "Kalman-Filter-Based System Identification via Retrospective Estimation of the Process Noise Covariance," in *Proc. Amer. Contr. Conf.*, Boston, MA, July 2016, pp. 4545–4550.

- F. Sobolic, A. Goel, and D. S. Bernstein, "Retrospective Cost Adaptive Control Using Concurrent Controller Optimization and Target Model Identification," in *Proc. Amer. Contr. Conf.*, Boston, MA, July 2016, pp. 3416–3421.

- F. Sobolic and D. S. Bernstein, "An Inner-Loop/Outer-Loop Architecture for an Adaptive Missile Autopilot," in *Proc. Amer. Contr. Conf.*, Chicago, IL, July 2015, pp. 850-855.

- F. Sobolic and D. S. Bernstein, "Aerodynamic-Free Adaptive Control of the NASA Generic Transport Model," in *Proc. AIAA Guid. Nav. Contr. Conf.*, Boston, MA., August 2013, AIAA-2013-4851.

1.2 Dissertation Outline

This dissertation is organized as follows.

Chapter 2 Summary

The results of Chapter 2 gives a brief overview of RCAC. The adaptive standard problem is introduced and the key details of RCAC are given. The controller gains in this chapter are found by using the recursive least squares algorithm. The capability of RCAC is then shown on an application to the NASA generic transport model (GTM) simulation. The GTM model has been used as an adaptive control testbed to

simulate uncertainties [9–11]. The uncertainties investigated in this chapter include a sudden increase in drag, a sudden decrease in lift, and both a time-varying increase in drag and decrease in lift which demonstrate icing on the lifting surfaces of the aircraft.

Chapter 3 Summary

In Chapter 3, we extend [12, 13] to the RCAC recursive update law. In particular, the use of multiplicative and additive covariance updates. Multiplicative covariance updates refer to the use of a forgetting factor which gives exponentially less weight to the older terms. By introducing a forgetting factor $\lambda < 1$, the filter is more sensitive to recent samples at the cost of being unstable. The filter places a discrete-time unstable pole at $1/\lambda > 1$, which implies that an increase in forgetting factor λ will lead to instabilities. To combat the constant instability, the variable forgetting factor $\lambda(k)$ becomes less than one when the new information differs from the old information. Therefore, the idea of the variable forgetting factor is to adjust $\lambda(k) < 1$ when new information is present or remain at 1.0 if the information is not changing.

The additive covariance update is based on the Kalman filter. The optimization of the controller coefficients are updated based on the A matrix equal to the identity matrix with the addition of a noise term that has a covariance of Q_{KF}. Specifically, we treat the gain updates as estimation of the coefficients of an input-output model. The dynamic system state components for the Kalman filter are thus the coefficients of the input-output model, and the dynamics matrix is the identity.

Finally, these techniques are applied to a nonlinear longitudinal missile model with an inner-loop/outer-loop adaptive missile autopilot. The constant and variable forgetting factor and the Kalman filter technique are compared against the baseline three-loop autopilot.

Chapter 4 Summary

The Kalman filter is a stochastically optimal observer that utilizes statistical in-

formation about the measurement and process noise. In particular, the measurement and process noise are assumed to be white Gaussian processes with zero mean and covariances R and Q, respectively. Although R is relatively easy to characterize in practice, it is often challenging to estimate Q. Consequently, there is an extensive amount of literature on adaptive Kalman filtering, where the innovations signal is used to construct an estimate of Q [14–17].

In this chapter, we develop an alternative approach to estimating Q and apply this technique to parameter estimation. First, we show that the process noise covariance Q that minimizes the cumulative innovations does not (in general) correspond to the value of Q that, when used in the Kalman filter, minimizes the cumulative state-estimate error. However, for parameter estimation, we show that under certain assumptions the value of Q that minimizes the cumulative state-estimate error does in fact minimize the cumulative innovations. This insight provides the basis for a novel approach to on-line estimation of the process noise covariance for parameter estimation problems.

The estimation technique uses a past window of data to rerun the Kalman filter along with an optimization routine that retrospectively determines the value of Q that minimizes the cumulative innovations at the present time. Based on the previous analysis, this estimate is assumed to minimize the state-estimate error. Since the optimization does not lend itself to analytical gradients, we use gradient-free optimization techniques.

This technique is applied to the system identification problem, specifically, estimation of the coefficients of an input-output model. The dynamical system state components for the Kalman filter are thus the coefficients of the input-output model, and the dynamics matrix is the identity. The ultimate objective is to use this technique to identify linear systems whose coefficients are time-varying.

Adaptive Kalman filter techniques have been developed for this sort of problem

in [14–16, 18, 19], where the covariances R and Q are determined in real-time. These methods are also innovations-based and are not derived in the sense of a minimization optimization problem of the innovation itself. Rather, they are based on averaging inside a moving estimation window. Another method commonly used for covariance identification is the maximum likelihood estimation (MLE). This method assumes that the noise is characterized by a Gaussian model and that the best fit is determined by minimizing the error for the given noise structure [19]. The type of parameter uncertainty investigated in this chapter is only constrained in the structure of Q.

Chapter 5 Summary

Although the overarching motivation for using feedback control versus open-loop control is the ability to overcome uncertainty, feedback control depends on a model of the plant in order to operate reliably and without risking instability. Assuming an exact and complete model, LQG can be used to stabilize all MIMO plants with optimal H_2 performance regardless of plant order, open-loop pole and zero locations, and channel coupling. In practice, however, uncertainty may be unavoidable due to complex, unknown, or unpredictably changing physics. To overcome model uncertainty, robust control techniques can be used to guarantee stability and performance, albeit at the expense of performance. Adaptive control can be viewed as a form of robust control, wherein the control law adjusts itself to the plant during operation, thereby circumventing the performance sacrifice inherent in robust control.

A fundamental goal of feedback control is to achieve maximal closed-loop performance in the presence of prior model uncertainty. In the case of adaptive control, closed-loop performance must account for transient performance as the controller adjusts itself to the actual plant characteristics. For example, universal adaptive control laws [20] can adapt to uncertainty in the sign of the leading coefficient of the plant transfer function, although the transient response may be large.

Nonminimum-phase (NMP) zeros also present a challenge to adaptive control; for

example, the control laws in [1–3] assume that the plant is minimum phase. Adaptive control of NMP plants is considered in [8, 21–24]. In [8, 24] knowledge of the NMP zeros is embedded in the target model G_{f}, which is used to filter the past data in order to retrospectively optimize the controller coefficients.

The goal of this chapter is to extend retrospective cost adaptive control (RCAC) as presented in [8, 24] to alleviate the need for prior modeling of both the sign of the leading coefficient of the plant transfer function as well as its NMP zeros. The key element of this extension is concurrent optimization of both the target model and the controller coefficients. In particular, we show that concurrent optimization facilitates the application of RCAC with less prior modeling information than is assumed in [8, 24]. In particular, the number and location of the NMP zeros need not be known aside from the parity of the number of positive NMP zeros.

Concurrent optimization of the target model and controller coefficients is a quadratic optimization problem in the target model and controller coefficients separately. However, this optimization problem is not convex as a joint function of both variables, and therefore nonconvex optimization methods are needed. In this chapter, we address this problem in two different ways. First we take advantage of the biquadratic structure of the cost function by applying an alternate convex search algorithm [25]. Related techniques for biconvex and bilinear optimization are given in [26–29]. For comparison, the Matlab fminsearch routine is used to simultaneously optimize the controller and target model coefficients.

Chapter 6 Summary

The starting point for this chapter is the survey paper [30], which analyzes various architectures for closed-loop identification. That work emphasizes the practical importance of the problem and demonstrates the richness of the subject in terms of the diverse architectures that can be employed. As a complement to [30], the contribution of this chapter is a detailed numerical study that compares multiple closed-loop iden-

tification architectures in terms of their accuracy in estimating nonminimum-phase (NMP) zeros.

This chapter thus considers six architectures for identifying a plant operating in closed loop, namely, direct closed-loop (DCL) identification, standard auxiliary direct closed-loop (ADCL/S) identification, intercalated auxiliary direct closed-loop (ADCL/I) identification, indirect closed-loop (ICL) identification, standard auxiliary indirect closed-loop (AICL/S) identification, intercalated auxiliary indirect closed-loop (AICL/I) identification.

The goal of this chapter is to assess the advantages and disadvantages of these six architectures in estimating the nonminimum-phase (NMP) zeros of a plant operating in closed loop. NMP zeros are one of the most challenging aspects of feedback control in terms of limiting achievable performance [31]. Consequently, after constructing a model of the open- or closed-loop plant, the metric used to assess the estimation accuracy is defined to be the accuracy of the estimate of the NMP zero. This objective is motivated by retrospective cost adaptive control [32–36], which requires knowledge of NMP plant zeros.

Chapter 7 Summary

The contents within chapter 7 goes beyond [12, 13, 37] by taking advantage of recent developments described in [36]. In particular, it is shown in [36] that the retrospective cost function used by RCAC is based on the residual of a fit between a specific closed-loop transfer function (called the *intercalated transfer function*) and the user-specified target model. The intercalated transfer function arises due to the way in which the virtual controller perturbation, due to the adaptation, is injected into the closed-loop system. Since plant zeros are invariant under feedback, it immediately becomes clear why plant nonminimum-phase (NMP) zeros must be reproduced in the target model: if these zeros are not included in the target model, RCAC may cancel them.

In this chapter, we explore the use of the adaptive controller on a longitudinal missile model. For the 3DOF missile, it is possible to compute the NMP zeros as a function of angle of attack and Mach number at each linearized flight condition. Although this could be done, the estimates of the NMP zeros depend on the available model and thus maybe erroneous due to model uncertainty. Therefore, we take an alternative approach, where the NMP zeros of the missile are estimated on-line and the estimates are used to update the target model at each time step. The contribution of this chapter is thus a detailed study of the feasibility of semi-adaptive control, where RCAC is used along with concurrent estimation of the plant NMP zeros in order to update the target model. This approach is demonstrated using a fully nonlinear simulation of the 3DOF missile.

Finally, we present conclusions in Chapter 8.

CHAPTER 2

The Retrospective Cost Adaptive Control (RCAC) Algorithm

2.1 Introduction

Retrospective cost adaptive control (RCAC) is a direct, digital adaptive control algorithm. RCAC can be applied to stable or unstable systems, minimum phase or non-minimum phase systems, linear or nonlinear systems, on stabilization, command following or disturbance rejection problems. In this chapter, we discuss the basics of RCAC and derive a recursive solution to the adaptive gains. To demonstrate the capability of the algorithm, RCAC is applied on the NASA generic transport model (GTM) where slow or abrupt uncertainties in the model are introduced.

2.2 The Adaptive Standard Problem

Consider the discrete-time system

$$x(k+1) = Ax(k) + Bu(k) + D_1 w(k), \tag{2.1}$$

$$y(k) = Cx(k) + D_2 w(k), \tag{2.2}$$

$$z(k) = E_1 x(k) + E_0 w(k), \tag{2.3}$$

where $x(k) \in \mathbb{R}^n$, $y(k) \in \mathbb{R}^{l_y}$, $z(k) \in \mathbb{R}^{l_z}$, $u(k) \in \mathbb{R}^{l_u}$, $w(k) \in \mathbb{R}^{l_w}$, and $k \geq 0$. The objective of the adaptive controller is to generate a control signal $u(k)$ that minimizes the performance variable $z(k)$ in the presence of exogenous signals $w(k)$. The exogenous signal $w(k)$ can represent a command signal to be tracked, an external disturbance, or both. The system (2.1) – (2.3) can represent a sampled-data system based on continuous-time dynamics with sample and hold operations.

2.3 Details of the RCAC Algorithm

In this section, we describe the details of RCAC. The main aspects of RCAC are broken into three parts: control law, retrospective performance variable construction, and retrospective cost function.

2.3.1 The Adaptive Control Law

Let the control signal be constructed as a strictly proper dynamic compensator of order n_c given by

$$u(k) = \sum_{i=1}^{n_c} P_i(k)u(k-i) + \sum_{i=1}^{n_c} Q_i(k)y'(k-i), \tag{2.4}$$

where, for all $i = 1, \ldots, n_c$, $P_i \in \mathbb{R}^{l_u \times l_u}$ and $Q_i \in \mathbb{R}^{l_u \times l_{y'}}$ are the gain matrices. The signal $y(k)'$ is usually chosen to be either the output $y(k)$, the performance $z(k)$, or both. The controller in (2.4) can be rewritten as

$$u(k) = \Phi(k)\theta(k), \tag{2.5}$$

where $\Phi(k)$ is the regressor matrix

$$\Phi(k) \triangleq I_{l_u} \otimes \begin{bmatrix} u(k-1) \\ \vdots \\ u(k-n_c) \\ y'(k-1) \\ \vdots \\ y'(k-n_c) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{l_u \times l_\theta}, \tag{2.6}$$

where $\otimes$ is the Kronecker Product and $\theta(k) = \mathrm{vec}\,[P(k) \;\; Q(k)] \in \mathbb{R}^{l_\theta}$ is the vector of the controller gains with size $l_\theta = n_c l_u (l_u + l_{y'})$.

2.3.2 Retrospective Performance Variable

We define the retrospective performance variable as

$$\hat{z}(\hat{\theta}(k), k) \triangleq z(k) + G_{\mathrm{f}}(\mathbf{q})[\Phi(k)\hat{\theta}(k) - u(k)], \tag{2.7}$$

$$G_{\mathrm{f}}(\mathbf{q}) \triangleq D_{\mathrm{f}}^{-1}(\mathbf{q}) N_{\mathrm{f}}(\mathbf{q}) \tag{2.8}$$

where

$$N_{\mathrm{f}}(\mathbf{q}) \triangleq \sum_{i=1}^{n_{\mathrm{f}}} N_i \mathbf{q}^{n_{\mathrm{f}}-i}, \; D_{\mathrm{f}}(\mathbf{q}) \triangleq I_{l_z} \mathbf{q}^{n_{\mathrm{f}}} - \sum_{i=1}^{n_{\mathrm{f}}} D_i \mathbf{q}^{n_{\mathrm{f}}-i}, \tag{2.9}$$

with $N_i \in \mathbb{R}^{l_z \times l_u}$ and $D_i \in \mathbb{R}^{l_z \times l_z}$ and $\mathbf{q}$ represents the forward shift operator. The filter G_{f} is of order $n_{\mathrm{f}} \geq 1$ and each polynomial entry of $D_{\mathrm{f}}(\mathbf{q})$ is asymptotically stable. Next, we are able to rewrite (2.7) as

$$\hat{z}(\hat{\theta}(k), k) = z(k) + \Phi_{\mathrm{f}}(k)\hat{\theta}(k) - u_{\mathrm{f}}(k), \tag{2.10}$$

where the filtered regressor and control are given by

$$\begin{aligned}\Phi_{\mathrm{f}}(k) &\triangleq G_{\mathrm{f}}(\mathbf{q})\Phi(k),\\ &= \sum_{i=1}^{n_{\mathrm{f}}} \left[N_i\Phi(k-i) + D_i\Phi_{\mathrm{f}}(k-i)\right], \qquad (2.11)\\ u_{\mathrm{f}}(k) &\triangleq G_{\mathrm{f}}(\mathbf{q})u(k),\\ &= \sum_{i=1}^{n_{\mathrm{f}}} \left[N_i u(k-i) + D_i u_{\mathrm{f}}(k-i)\right], \qquad (2.12)\end{aligned}$$

and $\hat{\theta}(k)$ is determined by the optimization below. For reasons discussed in Chapter 5, G_{f} is referred to as the *target model* and is chosen to be either a infinite-impulse-response (IIR) or a finite-impulse-response (FIR) filter.

2.3.3 The Cumulative Cost Function

Define the cumulative cost function to be minimized as

$$\begin{aligned}J(\hat{\theta}(k),k) \triangleq &\sum_{i=k_0}^{k} \hat{z}(i)^{\mathrm{T}} R_z(i)\hat{z}(i)\\ &+ \sum_{i=k_0}^{k} [\Phi(i)\hat{\theta}(k)]^{\mathrm{T}} R_u(i)[\Phi(i)\hat{\theta}(k)]\\ &+ [\hat{\theta}(k)-\hat{\theta}(0)]^{\mathrm{T}} R_\theta(k)[\hat{\theta}(k)-\hat{\theta}(0)], \qquad (2.13)\end{aligned}$$

where $R_z(i), R_u(i)$, and $R_\theta(k)$ are positive definite for all k. Here we assume that the weighting matrices are constant. Since (2.13) is a strictly convex function, its minimizer can be found by computing the partial derivative of $J(\hat{\theta}(k),k)$ with respect to $\hat{\theta}(k)$ and obtaining

$$\begin{aligned}\frac{\partial J(\hat{\theta}(k),k)}{\partial\hat{\theta}(k)} &= 2\sum_{i=k_0}^{k} [\hat{z}(i)^{\mathrm{T}} R_z \Phi_{\mathrm{f}}(i) + [\Phi(i)\hat{\theta}(k)]^{\mathrm{T}} R_u \Phi(i)]\\ &+2[\hat{\theta}(k)-\hat{\theta}(0)]^{\mathrm{T}} R_\theta = 2\mathcal{A}(k)^{\mathrm{T}} + 2\hat{\theta}(k)^{\mathrm{T}}\mathcal{P}(k)^{-1}, \qquad (2.14)\end{aligned}$$

where

$$\mathcal{A}(k) \triangleq \sum_{i=k_0}^{k} \left(\Phi_{\mathrm{f}}(i)^{\mathrm{T}} R_z \left[z(i) - u_{\mathrm{f}}(i)\right]\right) - R_\theta \hat{\theta}(0),$$

$$\mathcal{P}(k) \triangleq [\sum_{i=k_0}^{k} (\Phi_{\mathrm{f}}^{\mathrm{T}}(i) R_z \Phi_{\mathrm{f}}(i) + \Phi(i)^{\mathrm{T}} R_u \Phi(i)) + R_\theta]^{-1}.$$

2.3.4 The Recursive RCAC Update Law

In this section, we design the recursive least squares (RLS) algorithm used as an update law for obtaining the controller parameters $\hat{\theta}(k)$.

2.3.4.1 The RCAC Recursive Least Squares Update Law

To find a recursive solution for the controller parameters, we start with

$$\begin{aligned} \mathcal{A}(k) &= \mathcal{A}(k-1) + \Phi_{\mathrm{f}}(k)^{\mathrm{T}} R_z \left[z(k) - u_{\mathrm{f}}(k)\right] \\ &= \mathcal{A}(k-1) + X(k)^{\mathrm{T}} \bar{R}\, \bar{z}(k), \end{aligned} \tag{2.15}$$

where $\mathcal{A}(0) = -R_\theta \hat{\theta}(0)$ and

$$\mathcal{P}(k) = \mathcal{P}(k-1) - \mathcal{P}(k-1) X(k)^{\mathrm{T}} \Gamma(k)^{-1} X(k) \mathcal{P}(k-1), \tag{2.16}$$

$$X(k) \triangleq \begin{bmatrix} \Phi_{\mathrm{f}}(k) \\ \Phi(k) \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)\times l_\theta}, \tag{2.17}$$

$$\bar{R} \triangleq \begin{bmatrix} R_z & 0 \\ 0 & R_u \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)\times(l_z+l_u)}, \tag{2.18}$$

$$\bar{z}(k) \triangleq \begin{bmatrix} z(k) - u_{\mathrm{f}}(k) \\ 0 \end{bmatrix} \in \mathbb{R}^{l_z+l_u}, \tag{2.19}$$

$$\Gamma(k) \triangleq \bar{R}^{-1} + X(k) \mathcal{P}(k-1) X(k)^{\mathrm{T}}, \tag{2.20}$$

and $\mathcal{P}(0) \triangleq R_\theta^{-1}$. Combining (2.15) and (2.16) yields

$$\begin{aligned}\hat{\theta}(k) &= -\mathcal{P}(k)\mathcal{A}(k),\\ &= \hat{\theta}(k-1) - \mathcal{P}(k-1)X(k)^{\mathrm{T}}\Gamma(k)^{-1}\left[X(k)\theta(k-1)\right.\\ &\left.-X(k)\mathcal{P}(k-1)X(k)^{\mathrm{T}}\bar{R}\bar{z}(k) + \Gamma(k)\bar{R}\bar{z}(k)\right],\\ &= \hat{\theta}(k-1) + \mathcal{P}(k-1)X(k)^{\mathrm{T}}\Gamma(k)^{-1}\epsilon(k), \qquad (2.21)\end{aligned}$$

where $\hat{\theta}(0) = \hat{\theta}_0$ and $\epsilon(k) \triangleq -\bar{z}(k) - X(k)\hat{\theta}(k-1)$.

2.4 Application of RCAC on the NASA Generic Transport Model (GTM)

Aircraft control under emergency conditions poses severe challenges. For example, control surface faults may limit the maneuverability of the aircraft and require unconventional control strategies [38–41]. Although anticipated faults can be compensated for by contingency plans, unexpected faults require real-time adaptation under unknown conditions.

In this section, we are concerned with unanticipated and unknown changes in the aerodynamics of the aircraft as modeled by changes in its stability derivatives [42]. For each airspeed and altitude, stability derivatives provide a linearized approximation of the aerodynamic forces and moments on the aircraft as functions of perturbations from steady flight conditions. For aircraft certification and autopilot development, stability derivatives are typically determined through computational techniques and wind tunnel testing. These data can be stored in a lookup table for simulation studies.

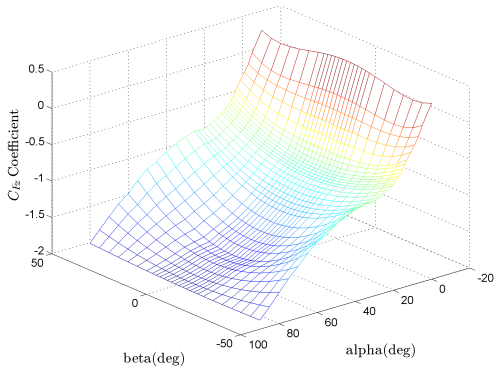
In this section, we consider emergency flight under abruptly or gradually changing stability derivatives. In particular, we apply retrospective cost adaptive control (RCAC) to various scenarios, such as slowly changing lift and drag coefficients to em-

ulate the effect of icing. Of particular interest is the evolution of the RCAC controller gains in response to unknown changes in the aircraft dynamics.

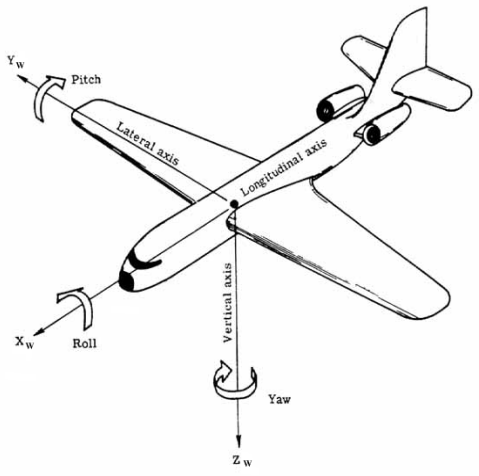
2.4.1 GTM Model and Controller Setup

In this section, we describe how the aerodynamic stability derivatives within GTM are accessed, and we present the RCAC architecture.

The GTM has an extensive lookup table of aerodynamic coefficients, which are programmed in Simulink as functions of angle of attack and side slip angle. These coefficients represent normalized forces and moments that can be acquired from either computational fluid dynamics software or empirically from an extensive wind tunnel experiment. Although these coefficients are related to the stability derivatives, individual stability derivatives are not specified. An illustration of this database is shown in Figure 2.1a for the force coefficient along the aircraft body z-axis, with the body-axis defined in Figure 2.1b. The aerodynamic coefficients that are modified in this chapter to illustrate RCAC are denoted by $C_{\mathrm{F}(\cdot)}$ for a force coefficient and $C_{\mathrm{M}(\cdot)}$ for a moment coefficient in the $(\cdot)$ -axis about the body frame.



(a) Body z-force aerodynamic coefficient mesh as a function of the angle of attack (α) and side slip angle (β)



(b) Aircraft body coordinate system.

Figure 2.1: A sample of the GTM aerodynamic data base 2.1a and an illustration of the aircraft body frame 2.1b

For this study, we focus on the ability of RCAC to control the aircraft to maintain

straight and level flight despite unknown time varying lift and drag aerodynamics. To achieve this objective, RCAC is set up as shown in Figure 2.2, where the components

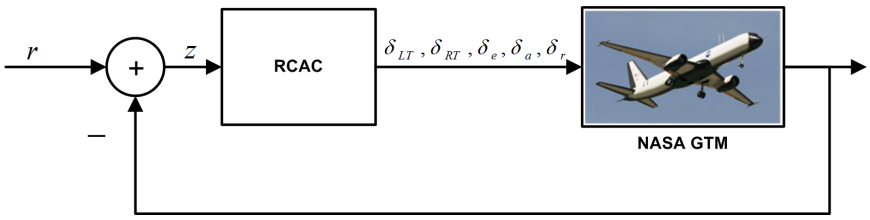


Figure 2.2: RCAC architecture applied on the GTM.

of $r \in \mathbb{R}^4$ represent altitude, airspeed, roll angle, and heading angle commands. Since the simulation is initialized at a given trim state, the reference commands are just these trim conditions. Feedback is utilized from the model to define the components of $z \in \mathbb{R}^4$ as the respective performance error signals of altitude, airspeed, roll angle, and heading angle. The output of RCAC is $\delta_{\text{LT}}, \delta_{\text{RT}}, \delta_e, \delta_a, \delta_r \in \mathbb{R}$, which are the commands to the left and right engines, elevator, aileron, and rudder, respectively. Also, Figure 2.2 shows the RCAC setup and how it is integrated with GTM. It is important to note that RCAC is not given any knowledge of when or how the aerodynamic coefficients will be changing. This implies that to compensate for these changes, RCAC must apply real-time adaptation and utilize the available control authority.

2.4.2 GTM Results

This section presents results based on the control architecture outlined in Section 2.3.1. As noted above, each example is commanded to maintain the initial trim altitude and airspeed. Since the goal is to maintain straight and level flight despite unknown time varying lift and drag forces, the force coefficients modifications are applied along the aircraft body x- and z-axes. For each case, RCAC is tasked to maintain a desired altitude, heading, roll angle, and airspeed while the aerodynamic coefficients are modified at time $t = 500$ sec. These changes range from sudden trans-

formations as in Example 2.1 and 2.2 to a linearly time varying transformation in Example 2.3.

In all of the examples below, RCAC utilizes a single tuning with the parameters

$$n_c = 10, \tag{2.22}$$

$$P(0) = 10 I_{(l_r+l_u)n_c}, \tag{2.23}$$

$$N_{1-5} = \begin{bmatrix} H_1 & H_2 & H_3 & H_4 & H_5 \end{bmatrix}, \tag{2.24}$$

$$R_z = \begin{bmatrix} R_{z_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & R_{z_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & R_{z_3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & R_{z_4} \end{bmatrix}, \tag{2.25}$$

$$R_u = \begin{bmatrix} R_{u_1} & 0 & 0 & 0 & 0 \\ 0 & R_{u_2} & 0 & 0 & 0 \\ 0 & 0 & R_{u_3} & 0 & 0 \\ 0 & 0 & 0 & R_{u_4} & 0 \\ 0 & 0 & 0 & 0 & R_{u_5} \end{bmatrix}, \tag{2.26}$$

where

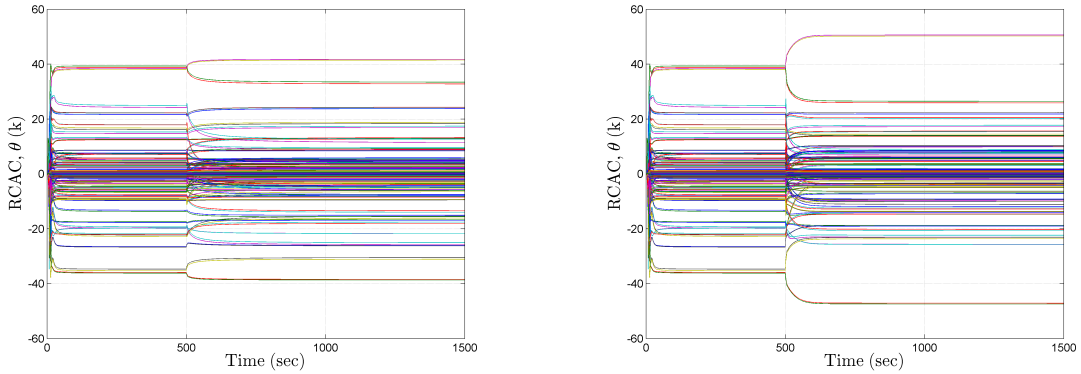
$$R_{z_1} = R_{z_2} = R_{z_3} = R_{z_4} = \begin{bmatrix} 1.5 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 30 \end{bmatrix},$$

and

$$\begin{bmatrix} R_{u_1} & R_{u_2} & R_{u_3} & R_{u_4} & R_{u_5} \end{bmatrix} = \begin{bmatrix} 0.002 & 0.002 & 0.70 & 0.70 & 0.07 \end{bmatrix}.$$

The target model is designed to be FIR with Markov parameters (2.24) calculated based on a single linearized version of GTM with the appropriate states modeled in the system. Thus, the Markov parameters in terms of the adaptive standard problem (2.1)-(2.3) are defined as

$$H_i \triangleq E_1 A^{i-1} B. \tag{2.27}$$



(a) (b)

Figure 2.3: The evolution of the RCAC gains for Example 2.1. Figure 2.3a shows the gains for a 100% increase in drag, while Figure 2.3b shows the gains for a 200% drag increase. The modified aerodynamic coefficient C_{Fx} is introduced into the simulation at $t = 500$ sec.

Example 2.1 (Sudden Increase in Drag)**.** Consider the case where the drag abruptly increases by an unknown scale factor. In this example, only the coefficient C_{Fx} is modified. For this case, the coefficient is modeled as

$$C_{\mathrm{Fx}} = \left(1 + \frac{\eta_{\mathrm{Fx}}}{100}\right) C_{\mathrm{Fx0}}, \tag{2.28}$$

where C_{Fx0} is the x-axis force coefficient calculated from the aerodynamic database, η_{Fx} is the percentage value increase from C_{Fx0}, and C_{Fx} is the modified body x-axis force coefficient utilized in the simulation. The aerodynamic change occurs in the simulation at $t = 500$ sec.

Figure 2.3 shows the evolution of the RCAC gains, where Figure 2.3a is the result

of increasing the drag coefficient by 100% and Figure 2.3b is due to an increase of 200%. Note that RCAC has no prior knowledge of the abrupt change in drag.

Figure 2.4 shows the results for a 0%, 100%, and 200% increase in drag. As shown in Figures 2.4a and 2.4b, the performance variables of altitude and airspeed are affected at time 500 sec but are compensated for and remain close to the desired performance values. Both Figures 2.4c and 2.4d are the control variables used to compensate for this change in dynamics. As shown, the elevator deflects appropriately due to the changes in the aerodynamic coefficient in order to maintain a suitable angle of attack (Figure 2.4e) for generating the lift necessary for altitude performance. The thrust increases to compensate for the abrupt increase in $C_{\mathrm{F_x}}$.

Example 2.2 (Sudden Decrease in Lift)**.** As in Example 2.1, we consider a sudden scale factor change at time 500 sec but this time affecting the lift. For this case, the coefficient is modeled as

$$C_{\mathrm{F_z}} = \left(1 - \frac{\eta_{\mathrm{F}_z}}{100}\right) C_{\mathrm{F_{z0}}}, \tag{2.29}$$

where $C_{\mathrm{F_{z0}}}$ is the z-axis force coefficient calculated from the aerodynamic database, $\eta_{\mathrm{F_z}}$ is the percentage decrease from $C_{\mathrm{F_{z0}}}$, and $C_{\mathrm{F_z}}$ is the modified body z-axis force coefficient utilized in the aircraft equations of motion.

Figure 2.5 shows the evolution of the RCAC gains for a 20% decrease in lift shown in Figure 2.5a and 40% decrease in lift as shown in Figure 2.5b. Note that the gains evolve more rapidly in the 40% case than the 20% case due to the difference in magnitude of the change in the aerodynamic coefficient.

Figure 2.6 shows the results due to a 0%, 20%, and 40% decrease in lift. As shown in Figures 2.6a and 2.6b the performance variables (altitude and airspeed) are affected at time 500 sec but are compensated for and remain close to the desired performance. Both Figures 2.6c and 2.6d depict the control authority used to compensate for this

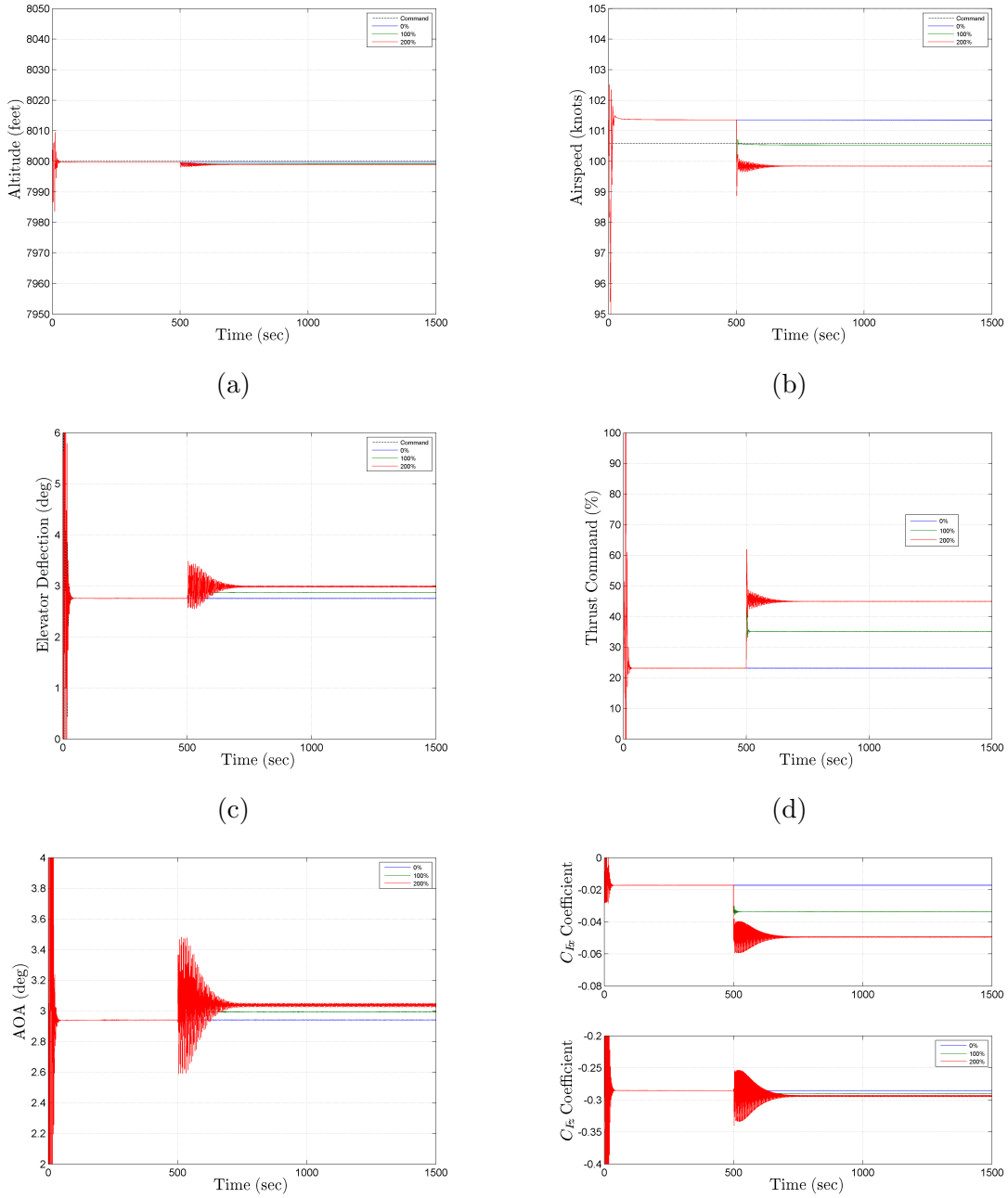
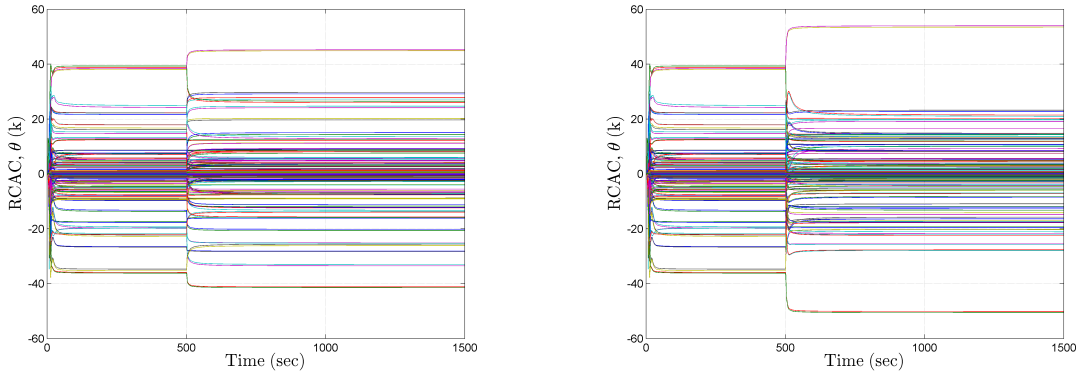


Figure 2.4: Example 2.1. The performance variable altitude in Figure 2.4a and airspeed in Figure 2.4b are affected at time 500 sec, RCAC recovers with the control authority of the elevator shown in Figure 2.4c and thrust in Figure 2.4d react appropriately to reject the abrupt increase in drag. Figure 2.4e shows the angle of attack (α) adjustment for the decrease in lift, and Figure 2.4f shows the aerodynamic coefficients. Note that the coefficient $C_{\mathrm{F_x}}$ illustrates the increases in drag while the lift coefficient $C_{\mathrm{F_z}}$ is only slightly modified due to dynamic coupling.



(a) (b)

Figure 2.5: Evolution of the gains for Example 2.2. a shows the gains of RCAC for a 20% decrease in lift, while b shows the gains for a 40% decrease in lift. Note that the gains evolve from one steady-state value for $t < 500$ sec to another for $t > 500$ sec due to the abrupt decrease in lift.

abrupt change in lift. As shown, the elevator deflects in the correct direction with suitable magnitude in order to increase the angle of attack (Figure 2.6e) and therefore maintain the lift necessary for the altitude, while the thrust is decreased slightly to maintain airspeed. By modifying R_z in the RCAC algorithm it is possible to weigh the altitude performance relative to the airspeed due to the aircraft's inability to produce the lift necessary to maintain altitude at a given airspeed without stalling.

Example 2.3 (Icing Example)**.** To emulate the effect of icing on the lifting surfaces of the aircraft, the aerodynamic force coefficients $C_{\mathrm{F_x}}$ and $C_{\mathrm{F_z}}$ are modified. In particular, we model icing as the degradation of lift produced by the wings and an increase in drag. The aerodynamic coefficients are modeled as

$$C_{\mathrm{F}(\cdot)} = \begin{cases} C_{\mathrm{F}(\cdot)_0}, & t < 500 \text{ sec}, \\ C_{\alpha\mathrm{F}(\cdot)}\,(t - 500) + C_{\mathrm{F}(\cdot)_0}, & 500 \text{ sec} \leq t < 1000 \text{ sec}, \\ 500 C_{\alpha\mathrm{F}(\cdot)} + C_{\mathrm{F}(\cdot)_0}, & t \geq 1000 \text{ sec}, \end{cases} \tag{2.30}$$

where $(\cdot)$ is the desired body axis label (x, y, z), $C_{\mathrm{F}(\cdot)_0}$ is the axis force coefficient calculated from the aerodynamic database, and $C_{\alpha\mathrm{F}(\cdot)}$ is the slope of the coefficient

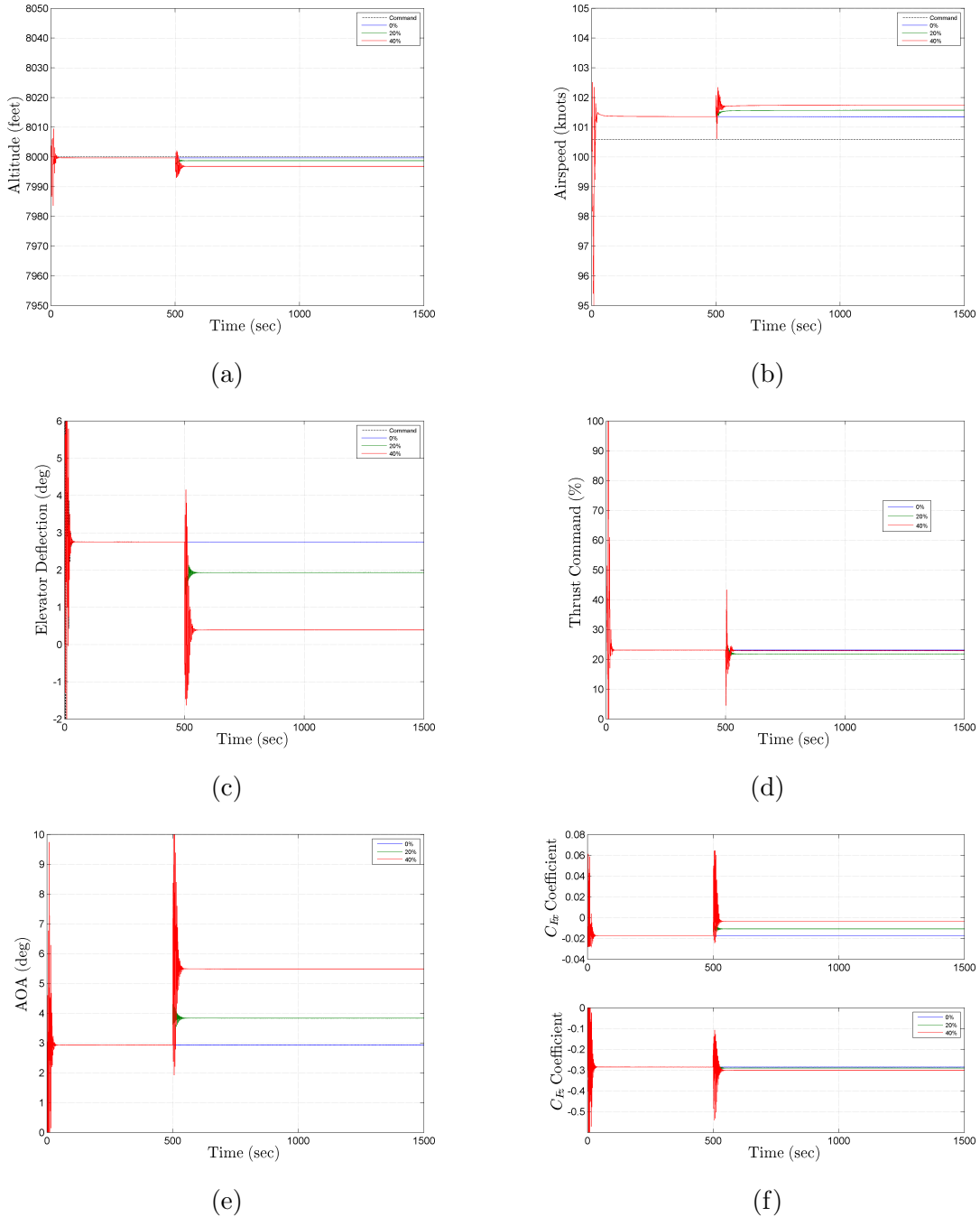


Figure 2.6: Example 2.2. These results show how the performance variables of altitude in Figure 2.6a and airspeed in Figure 2.6b are affected at time 500 sec but recovers with control authority from the elevator shown in Figure 2.6c and thrust in Figure 2.6d. Figure 2.6e shows the angle of attack (α) adjustment for the decrease in lift and Figure 2.6f shows the aerodynamic coefficients. Note that $C_{\mathrm{F_Z}}$, remains unchanged past 500 sec to maintain the appropriate lift in order to hold the desired altitude performance, while $C_{\mathrm{F_X}}$ decreases, resulting in less thrust necessary to maintain airspeed and steady level flight

modification governed by

$$C_{\alpha \mathrm{F}(\cdot)} = \frac{\eta_{\mathrm{F}(\cdot)}}{100} \frac{\Lambda_{\mathrm{F}(\cdot)}}{500}, \tag{2.31}$$

where $\Lambda_{\mathrm{F}(\cdot)}$ is the axis force coefficient from the database at $t = 500$ sec and $\eta_{\mathrm{F}(\cdot)}$ is the percent increase/decrease from $\Lambda_{\mathrm{F}(\cdot)}$. For this example, these constants are

$$\begin{aligned} \Lambda_{\mathrm{F}x} &= -0.01676, \quad \eta_{\mathrm{F}x} = 700, \\ \Lambda_{\mathrm{F}z} &= -0.29051, \quad \eta_{\mathrm{F}z} = -30. \end{aligned} \tag{2.32}$$

Note that at time 500 sec, C_{Fx} is linearly increased by 700%, while C_{Fz} is linearly decreased by 30% in 500 sec as dictated by (2.30).

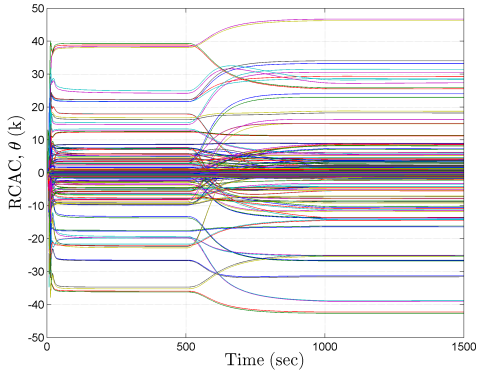


Figure 2.7: Evolution of the RCAC gains for Example 2.3. Note that the controller gain evolve considerably over 500 sec $< t <$1000 sec to adapt to the unknown time-varying aerodynamic coefficients.

Figure 2.7 shows the evolution of the gains of the RCAC algorithm. Note that at $t < 500$ sec the gains converge to a steady value, but during 500 sec $< t <$ 1000 sec, the gains evolve to compensate for the aerodynamic coefficient modifications. For $t >$ 1000 sec, the gains again converge to a steady-state value since the coefficients are no longer changing.

Figures 2.7 and 2.8 show the results for the icing example. Figure 2.8a and Figure 2.8b show how the performance variables of altitude and airspeed are maintained

near the desired reference signal. The control authority needed to maintain the desired states are shown in Figure 2.8c and Figure 2.8d. Note that the thrust increases to compensate for the increase in drag, and the elevator decreases to increase the angle of attack shown in Figure 2.8e. Figure 2.8f illustrates the aerodynamic coefficients before and after they are modified. Note that $C_{\mathrm{F}(\cdot)_0}$ is the coefficient from the database prior to modification.

2.5 Conclusions

In this chapter, we showed the basic retrospective cost adaptive control (RCAC) algorithm and derived a recursive least squares solution to solve for the adaptive gains. We applied RCAC to the NASA GTM model under unanticipated and unknown changes to the aerodynamics of the aircraft. Specifically, we used a single RCAC block within the simulation that controls the aircraft to a desired steady level flight by commanding five actuation channels (left and right engines, aileron, elevator, and rudder). The goal is to examine the evolution of the RCAC controller gains in response to changes in the aerodynamic coefficients from the linearized aircraft dynamics. To show this, simulations in the Simulink GTM were run, all with the desire for the aircraft to fly in a straight and level flight configuration. At a certain time (unknown to RCAC) the aerodynamic coefficients changed. Presented were three different aerodynamic parameter modifications, including an icing example, a sudden increase in drag, and a sudden decrease in lift. For all three examples, only a single tuning was used for each RCAC block.

Results showed that RCAC is able to adapt its gains in order to compensate for the unknown time-varying aerodynamic perturbations while maintaining the desired performance. The icing example showed that both the elevator and engines were used to compensate for an increase of 700% drag and decrease of 30% in lift production over a 500 sec interval. In the drag example, RCAC overcame a increase in the

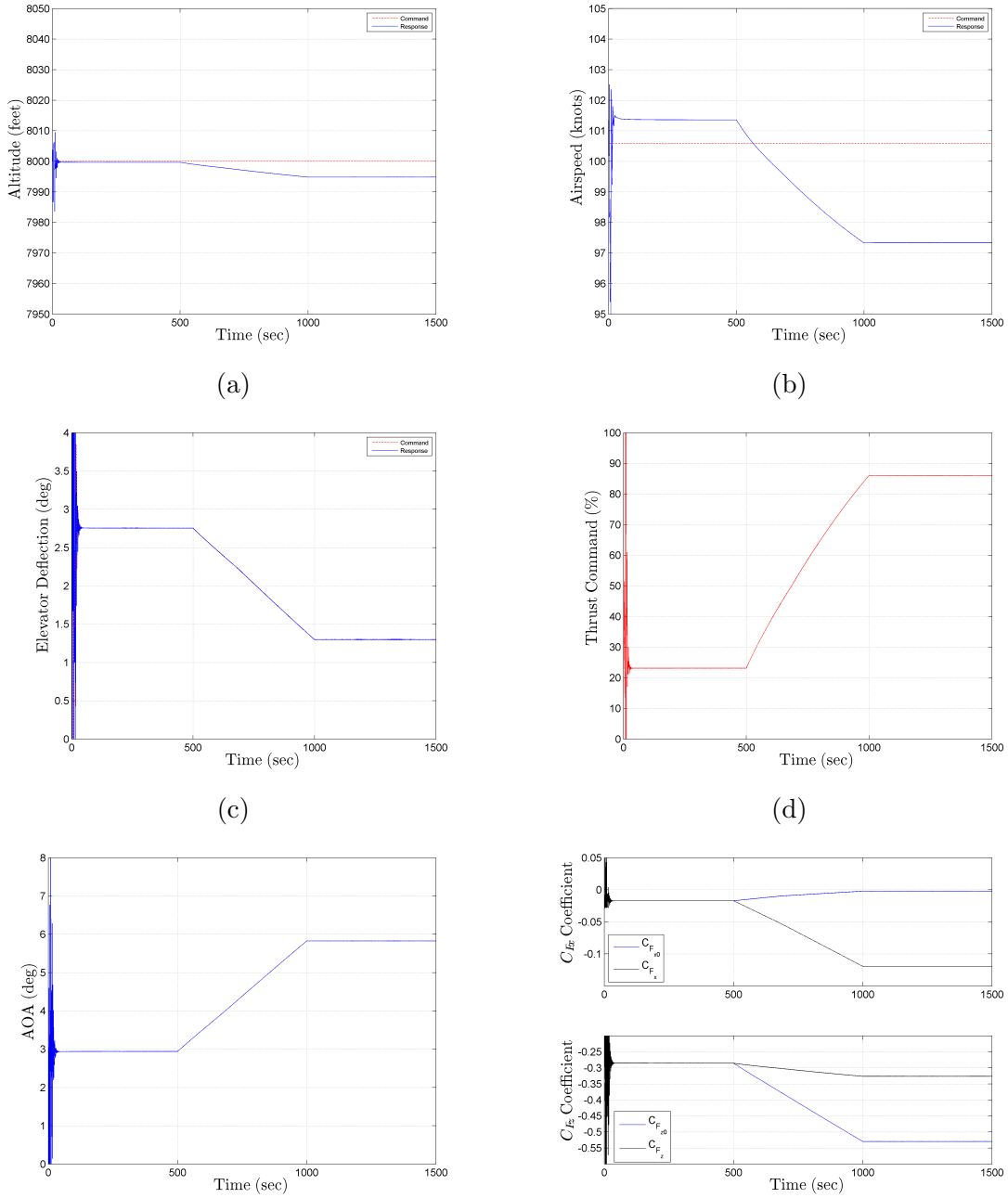


Figure 2.8: Example 2.3. The performance variable of altitude a and airspeed b with control authority of the elevator c and engine thrust d follow the altitude and airspeed commands well despite changes in the aerodynamic coefficients at 500 sec. e shows the angle of attack (α) adjustment for the decrease in lift, and f shows the aerodynamic coefficients. Note that unlike $C_{\mathrm{F_x}}$, $C_{\mathrm{F_z}}$ does not vary greatly after 500 sec. This is because the elevator deflects to increase α to maintain the same lift coefficient for the desired airspeed, while the thrust is used to compensate for the increase in drag.

aerodynamic drag coefficient by over 200% by increasing the engine thrust and using minimal elevator deflections. The gains evolved substantially to compensate for the amount of drag induced. Finally, we considered an example where the lift decreased by 40%. Given the constraints imposed by the performance variable on the airspeed, RCAC used the elevator to increase the angle of attack to maintain altitude. Future research focus on additional aerodynamic stability derivative changes as well as a method for relaxing constraints to maintain safe level flight.

CHAPTER 3

Extensions to the RCAC Recursive Update Law

3.1 Introduction

In this chapter, extensions to the RCAC recursive update law are developed. Specifically, we are interested in applications where the system dynamics are time-varying. The recursive least squares (RLS) update requires an initial covariance matrix $\mathcal{P}$ which represents an initial "error" in the estimated gains. During each step in adaptation, this covariance matrix either remains the same or decreases. A number of adhoc methods have been developed to reset, or increase, the covariance matrix when it becomes too low [43] thus leading to insufficient or nonexistent adaptation. Presented in this chapter are methods that include both multiplicative and additive covariance matrix updates. We extend the formulation in [12, 13] to include additional features, namely, variable forgetting factor (VFF) and Kalman Filter (KF) extensions of the recursive least squares (RLS) update of the controller gains. VFF techniques [44, 45] are used to adjust the forgetting factor λ in RLS based on the information provided by the latest measurement. At the end of the chapter, we apply these methods to a nonlinear missile model through the aid of an inner-loop/outer-loop control architecture.

3.2 Development of Multiplicative and Additive RCAC Recursive Least Squares (RLS) Covariance Updates

In this section, we design an update law for obtaining the RCAC controller parameters $\hat{\theta}(k)$. For convenience, the derivation of the recursive least squares (RLS) algorithm is repeated followed by the modifications to RLS which include the constant forgetting factor (CFF), the variable forgetting factor (VFF), and the Kalman Filter (KF).

3.2.1 Recursive Least Squares (RLS)

To find a recursive solution for the controller parameters, we start with

$$\begin{aligned}\mathcal{A}(k) &= \mathcal{A}(k-1) + \Phi_{\mathrm{f}}(k)^{\mathrm{T}} R_z \left[z(k) - u_{\mathrm{f}}(k)\right] \\ &= \mathcal{A}(k-1) + X(k)^{\mathrm{T}} \bar{R}\, \bar{z}(k), \end{aligned} \tag{3.1}$$

where $\mathcal{A}(0) = -R_\theta \hat{\theta}(0)$ and

$$\begin{aligned}\mathcal{P}(k) &= \left[\mathcal{P}(k-1)^{-1} + \Phi_{\mathrm{f}}^{\mathrm{T}}(k) R_z \Phi_{\mathrm{f}}(k) + \Phi(k)^{\mathrm{T}} R_u \Phi(k)\right]^{-1} \\ &= \mathcal{P}(k-1) - \mathcal{P}(k-1) X(k)^{\mathrm{T}} \Gamma(k)^{-1} X(k) \mathcal{P}(k-1), \end{aligned} \tag{3.2}$$

$$X(k) \triangleq \begin{bmatrix} \Phi_{\mathrm{f}}(k) \\ \Phi(k) \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)\times l_\theta}, \tag{3.3}$$

$$\bar{R} \triangleq \begin{bmatrix} R_z & 0 \\ 0 & R_u \end{bmatrix} \in \mathbb{R}^{(l_z+l_u)\times(l_z+l_u)}, \tag{3.4}$$

$$\bar{z}(k) \triangleq \begin{bmatrix} z(k) - u_{\mathrm{f}}(k) \\ 0 \end{bmatrix} \in \mathbb{R}^{l_z+l_u}, \tag{3.5}$$

$$\Gamma(k) \triangleq \bar{R}^{-1} + X(k) \mathcal{P}(k-1) X(k)^{\mathrm{T}}, \tag{3.6}$$

and $\mathcal{P}(0) \triangleq R_\theta^{-1}$. Combining (3.1) and (3.2) yields

$$\begin{aligned}\hat{\theta}(k) &= -\mathcal{P}(k)\mathcal{A}(k),\\ &= \hat{\theta}(k-1) - \mathcal{P}(k-1)X(k)^{\mathrm{T}}\Gamma(k)^{-1}\left[X(k)\theta(k-1)\right.\\ &\left.-X(k)\mathcal{P}(k-1)X(k)^{\mathrm{T}}\bar{R}\bar{z}(k) + \Gamma(k)\bar{R}\bar{z}(k)\right],\\ &= \hat{\theta}(k-1) + \mathcal{P}(k-1)X(k)^{\mathrm{T}}\Gamma(k)^{-1}\epsilon(k), \end{aligned} \tag{3.7}$$

where $\hat{\theta}(0) = \hat{\theta}_0$ and $\epsilon(k) \triangleq -\bar{z}(k) - X(k)\hat{\theta}(k-1)$.

3.2.2 RCAC with Constant Forgetting Factor (CFF)

The recursive algorithm based on RLS is modified to include a constant forgetting factor by redefining the weight matrices

$$R_z(k,i) = \lambda^{k-i}R_z, \quad R_u(k,i) = \lambda^{k-i}R_u, \quad R_\theta(k) = \lambda^k R_\theta,$$

where $\lambda \in (0,1]$. The recursion on (3.1) and (3.2) is then

$$\mathcal{A}(k) = \lambda\mathcal{A}(k-1) + \Phi_{\mathrm{f}}(k)^{\mathrm{T}}R_z(k)\left[z(k) - u_{\mathrm{f}}(k)\right], \tag{3.8}$$

$$\mathcal{P}(k) = \left[\lambda\mathcal{P}(k-1)^{-1} + X(k)^{\mathrm{T}}\bar{R}(k)X(k)\right]^{-1}, \tag{3.9}$$

which yields the recursive update

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \mathcal{P}(k-1)X(k)^{\mathrm{T}}\tilde{\Gamma}(k)^{-1}\epsilon(k), \tag{3.10}$$

where $\tilde{\Gamma}(k) \triangleq \lambda\bar{R}(k)^{-1} + X(k)\mathcal{P}(k)X(k)^{\mathrm{T}}$.

3.2.3 Variable Forgetting Factor (VFF)

Following [44] and [45], we define

$$K(k) \triangleq \mathcal{P}(k)X(k)^{\mathrm{T}}\Gamma(k)^{-1}, \tag{3.11}$$

$$\mathcal{E}(k) \triangleq \epsilon(k)^{\mathrm{T}}\left(I - X(k)K(k)\right)\epsilon(k), \tag{3.12}$$

$$W(k) \triangleq \mathcal{P}(k) - K(k)X(k)\mathcal{P}(k). \tag{3.13}$$

The VFF variable $\lambda(k)$ is then defined as $\lambda(k) \triangleq 1 - \mathcal{E}(k)/\Sigma_0$, where $\Sigma_0 = \sigma_0^2 N_0$, σ_0^2 is the expected measurement noise variance, and N_0 determines the speed of adaptation, which corresponds to the nominal asymptotic memory length [44]. Note that $\lambda(k)$ is bounded by the rule

if $\lambda(k) < \lambda_{min}$, **then** $\lambda(k) = \lambda_{min}$,
elseif $1/\lambda(k)\mathrm{tr}(W(k)) > c$, **then** $\lambda(k) = 1$,
endif.

The covariance matrix is then updated as

$$\mathcal{P}(k) = \frac{1}{\lambda(k)}W(k), \tag{3.14}$$

with $\mathcal{P}(0) = \delta I, c > \delta$, and the update for $\hat{\theta}(k)$ is (3.7).

3.2.4 Kalman Filter Update

The Kalman Filter update, where $A = I$, is of the form

$$\hat{\theta}(k) = \hat{\theta}(k-1) + w_v(k), \tag{3.15}$$

where $w_v(k)$ is a white sequence with covariance $Q_{\mathrm{KF}}(k)$. The Kalman Filter can then be implemented by computing the Kalman gain as

$$K(k) = \mathcal{P}(k)X(k)^{\mathrm{T}}\Gamma(k)^{-1}. \tag{3.16}$$

Note that the structure of $K(k)$ is the same as (3.11), and the matrix $\bar{R}^{-1}$ can be viewed as the measurement noise covariance. The next step is to update the estimate and compute the error covariance as

$$\begin{aligned}\hat{\theta}(k) &= \hat{\theta}(k|k-1) + K(k)[-\bar{z}(k) - X(k)\hat{\theta}(k|k-1)],\\ \mathcal{P}(k) &= [I - K(k)X(k)]\mathcal{P}(k|k-1).\end{aligned}$$

Finally, we project ahead as

$$\hat{\theta}(k+1|k) = \hat{\theta}(k), \tag{3.17}$$

$$\mathcal{P}(k+1|k) = \mathcal{P}(k) + Q_{\mathrm{KF}}(k). \tag{3.18}$$

3.3 Missile Application of RCAC with Multiplicative and Additive Covariance Updates

Autopilot design for high-performance missiles presents multiple challenges to control technology. Missiles typically fly through a wide range of Mach numbers, high angles of attack, and under high 'g' loading, leading to strongly nonlinear dynamics [46]. The standard approach to addressing these nonlinearities is to schedule the control gains based on the missile's aerodynamics as they vary during the course of the flight [46, 47].

Unfortunately, the standard approach to missile autopilot design requires aerodynamic lookup tables based on extensive wind tunnel test data, and thus is both

expensive and time-consuming. This situation motivates the use of adaptive control laws that can compensate online for changing aerodynamics and uncertainty in the aerodynamic model. In order to successfully control a high-performance missile, however, an adaptive control law must be able to adapt sufficiently quickly to rapidly varying commands from the missile guidance system. With these challenges in mind, the present chapter extends the work of [12, 13], where retrospective cost adaptive control (RCAC) was used to control the planar missile model described in [48]. This model has been used extensively [49, 50] to study the feasibility of various autopilot control schemes.

RCAC, which was developed in [6–8], has been shown to be effective on nonminimum-phase systems. This property is relevant to the planar missile model in [48], where the nose-mounted gyro and tail-fin actuation give rise to nonminimum-phase pitch dynamics. The goal then, is to apply RCAC to the planar missile in [48] under more aggressive commands than were considered in [12, 13].

An additional novel element in this chapter is an inner-loop/outer-loop control architecture. Specifically, RCAC adaptively adjusts the pitch-rate command based on the normal acceleration command from guidance. This adaptation would not be needed if the missile were commanded to fly along a circular arc with a constant velocity. However, for arbitrary normal acceleration commands, the appropriate pitch-rate command cannot be inferred, and this motivates the use of RCAC/VFF and RCAC/KF to adaptively specify the appropriate pitch-rate command. The pitch-rate command is then used to drive an inner-loop controller that acts on the error between the pitch-rate command and the pitch-rate measurement. The pitch-rate loop architecture is a static gain proportional/integral controller that stabilizes the pitch rate about a single trim point.

3.3.1 Problem Formulation and Nonlinear Missile Model

To intercept a moving target, the missile is equipped with an active seeker that provides the normal acceleration ('g') command that the missile must follow in order to reach its target. In this section we briefly describe the model used. See [48] for more details.

The target dynamics in an inertial frame are given by

$$\dot{X}_{\mathrm{t}} = V_{\mathrm{t}}(0)\cos\theta_{\mathrm{t}}(0), \quad \dot{Z}_{\mathrm{t}} = V_{\mathrm{t}}(0)\sin\theta_{\mathrm{t}}(0), \tag{3.19}$$

where X_{t} and Z_{t} represent the inertial coordinates of the target and $V_{\mathrm{t}}(0)$, $\theta_{\mathrm{t}}(0)$, $X_{\mathrm{t}}(0)$, and $Z_{\mathrm{t}}(0)$ are the target's initial conditions.

The nonlinear three-degree-of-freedom missile dynamics described in the body frame are given by

$$m\dot{U} = \sum F_{B_X} - mQW + T_{\mathrm{Thrust}}, \tag{3.20}$$

$$m\dot{W} = \sum F_{B_Z} + mQU, \tag{3.21}$$

$$I_Y\dot{Q} = \sum M_Y, \tag{3.22}$$

$$\dot{\theta} = Q, \tag{3.23}$$

$$\dot{X} = U\cos\,\theta + W\sin\theta, \tag{3.24}$$

$$\dot{Z} = -U\sin\,\theta + W\cos\theta, \tag{3.25}$$

where $U(0) = U_0$, $W(0) = W_0$, $Q(0) = Q_0$, $\theta(0) = \theta_0$, $X(0) = X_0$, $Z(0) = Z_0$.

Assuming a flat Earth, the moment and forces about the center of gravity are

$$\sum F_{B_X} = \bar{q}SC_A - mg\sin\theta, \tag{3.26}$$

$$\sum F_{B_Z} = \bar{q}SC_N(\alpha, M, \delta_p) + mg\cos\theta, \tag{3.27}$$

$$\sum M_Y = \bar{q}SdC_m(\alpha, M, \delta_p, Q), \tag{3.28}$$

where $\bar{q} = \frac{1}{2}\rho V^2$ is the dynamic pressure.

The aerodynamic coefficients are modeled as functions of Mach number M, angle-of-attack α, fin deflection angle δ_p, and pitch rate Q, as

$$C_A = a_a, \tag{3.29}$$

$$C_N = a_n\alpha^3 + b_n\alpha|\alpha| + c_n\left(2 - \frac{M}{3}\right)\alpha + d_n\delta_p, \tag{3.30}$$

$$C_m = a_m\alpha^3 + b_m\alpha|\alpha| + c_m\left(-7 + \frac{8M}{3}\right)\alpha + d_m\delta_p + e_mQ, \tag{3.31}$$

where

$$\alpha \triangleq \tan^{-1}(W/U), \quad V^2 \triangleq U^2 + W^2, \quad M \triangleq V/a,$$

where a is the altitude-dependent speed of sound. The missile model assumes planar flight, and thus only the X, Z coordinates and pitch are modeled, yielding a 6th-order model. Out-of-plane effects, such as roll angle, yaw angle, and sideslip are therefore fixed at zero. Additionally, the fin actuator is modeled as a second-order system. For realism, an actuator rate and magnitude saturations are implemented at 500 deg/sec and 30 deg, respectively. Finally, the normalized normal acceleration is given by $n_z = F_N/(gm) + \cos\theta$.

3.3.2 Three-Loop Autopilot (3LA)

The goal of the 3LA [48] is to minimize the error between the commanded normal acceleration, generated by the guidance law, and the normal acceleration measurement provided by the inertial measurement unit (IMU). The states available for measurement are the normal acceleration n_z and the pitch rate Q, which is provided by an on-board accelerometer and gyroscope, respectively. 3LA is implemented as

$$u(s) = K_{\mathrm{Q}} Q(s) + \frac{1}{s}\left(K_\theta Q(s) + K_{\mathrm{I}}\left[K_{\mathrm{SS}} n_{z,\mathrm{cmd}} - n_{z,\mathrm{IMU}}\right]\right),$$

where $K_{\mathrm{Q}}, K_\theta, K_{\mathrm{I}}$, and K_{SS} are the control gains determined by modeling and analysis. Each gain is scheduled on a trim condition based on the missile angle of attack and Mach number. In practice, a digital version of the control law is implemented in discrete time. In this section, we use the 3LA as a baseline for comparative studies.

3.3.3 Adaptive Autopilot Architecture

RCAC generates a pitch-rate command by adapting on the normal acceleration error and by using the normal acceleration command in the controller regressor Φ. As shown in Fig. 3.1, this pitch-rate command is then used as the reference to a fixed-gain proportional-integral (PI) controller. The PI loop is a pitch-rate stabilizing loop based on a single trim condition. This differs from [12], where the adaptive control law acts directly on the normal acceleration error to generate fin deflection commands based on that error.

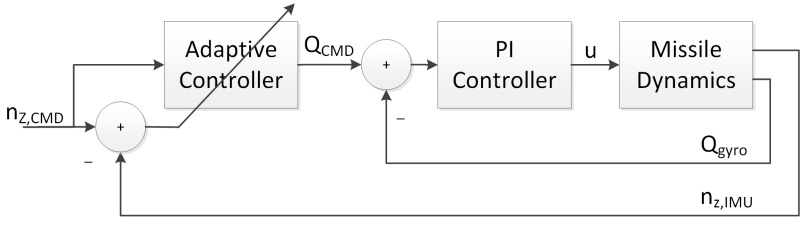


Figure 3.1: Inner-Loop/Outer-Loop Adaptive Autopilot Architecture

Note that the steady-state pitch-rate command for the missile flying in a circular arc with a constant velocity is $Q_{\text{CMD,SS}} = g(\sin\theta - n_{z,\text{CMD}})/V_{\text{T}}$, where V_{T} is the tangential velocity. Since thrust is not regulated (assuming post burn-out), maintaining a constant velocity is not feasible. Other than this particular maneuver, deriving the equations for a pitch-rate command involves information that is not directly measured, thus motivating the need for an adaptive pitch-rate command.

To apply RCAC. the missile dynamics are linearized about a single trim point, and these dynamics are augmented by the PI controller. The resulting matricies A, B, and C from (2.1) and (2.2) are used to create the filter G_{f} [51] in (2.8). D_1 and D_2 are not used since external disturbances are not considered. Note that the linearization is used only for constructing an FIR filter and goes into the design of N_{f}; all simulations are performed using the full nonlinear dynamics. For RCAC, the performance signal $z(k)$ in (2.3) is used in the regressor Φ as well as in the cost minimization (5.12), where $z(k)$ is the difference between the normal acceleration command $n_{\text{z,cmd}}$, which is represented by $w(k)$ as in (2.3), and the normal acceleration measurement $n_{\text{z,IMU}}$. Finally, for notational convenience, the time step k, represents the actual sample time of the controller, $k\text{T}_s$.

3.3.4 Example of RCAC with standard RLS

We present an example that illustrates the need for modifying the RLS update law. Consider the nonlinear missile model presented in Section 3.3.1. The missile is initialized at $M_0 = 3$, $\theta_0 = 25$ deg, $Z_0 = 3$ km, $Q_0 = X_0 = \alpha_0 = 0$, and the target is initialized at $X_{\text{t}}(0) = 4$ km, $Z_{\text{t}}(0) = 2.8$ km, $M_{\text{t}}(0) = 0.3$, and $\theta_{\text{t}}(0) = 180$ deg. RCAC has the initial values of $n_{\text{c}} = 4$, $n_{\text{f}} = 1$, $N_1 = -0.6$, $R_z = 1$, $R_u = 3$, and $\mathcal{P}(0) = 1e^{10}I_{2n_{\text{c}}}$. In order to intercept the target, the missile is commanded to pull an aggressive maneuver and consequently high angles of attack. This maneuver ensures that the missile's nonlinearities are exposed. Fig. 3.2 compares the 3LA to

the pitch-rate commanded RCAC architecture. As shown in the figure, the adaptive controller is initially able to track the acceleration command but, as the flight progresses, tracking becomes worse. This inability to track is depicted by the controller gains and covariance in Fig.3.2. After 0.6 sec, the controller gains converge and the eigenvalues in the covariance matrix $\mathcal{P}$ approach zero.

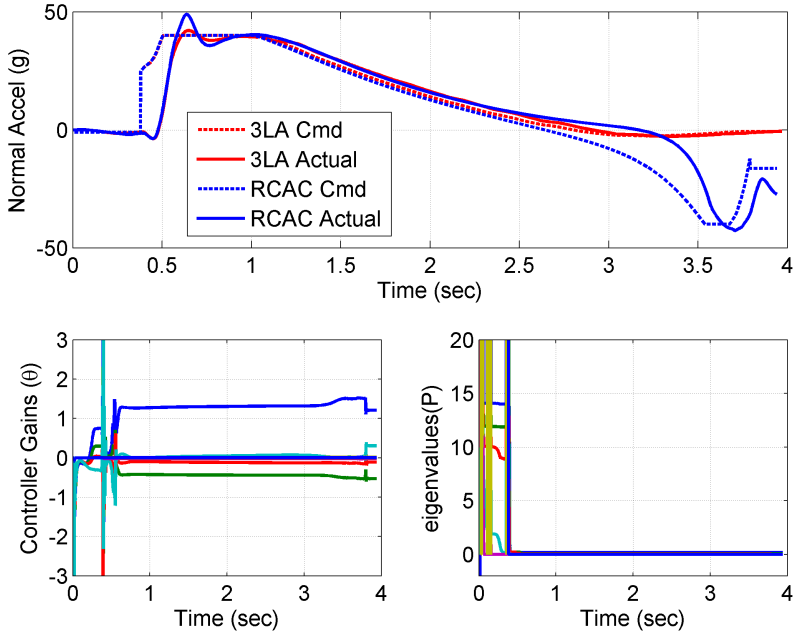


Figure 3.2: Missile 'g' command tracking of the gain-scheduled three-loop autopilot vs. RCAC without a forgetting factor. [Top] compares the gain-scheduled 'g' command/response versus the RCAC 'g' command/response, where the tracking error increases significantly toward the end of flight, [Bottom] shows that the RCAC gains remain almost constant after about 0.6 sec due to the eigenvalues of the covariance matrix tending toward zero.

3.3.5 Comparison of 3LA, RCAC/CFF, and RCAC/VFF

We revisit the example in Section 3.3.4 and compare RCAC/CFF and RCAC/VFF with the three-loop autopilot (3LA). In this example, we vary the constant forgetting factor from 0.975 to 1.0 and compare the calculated miss distance (MD), which, after the target has been acquired, is the distance between the missile and the target at the instant the seeker loses sight of the target. The variable forgetting factor parameters are set to $\Sigma_0 = 1000$, $\lambda_{\min} = 0.98$, and $c = 10\delta$. Fig. 3.3 shows the miss distance

compared to a range of constant forgetting factors. Included in the plot is both the miss distance with the 3LA and the miss distance with RCAC/VFF. As the CFF is increased past 0.995, the miss distance becomes increasingly large (above the 2 meter plot limit). Note that the RCAC/VFF as well as some RCAC/CFF miss distances are below that of the 3LA.

To compare the CFF with VFF, note that with $\lambda = 0.979$, the CFF miss distance is the lowest of all the FF. The bottom of Fig. 3.3 depicts the adaptive gains through the missile flight. Notice how the gains are more aggressive and oscillatory during the latter half of the flight time for the CFF compared to VFF. This type of behavior is undesirable since it may cause the adaptive controller to become unstable leading to erroneous pitch-rate commands. The top right of Fig. 3.3 shows how the variable forgetting factor λ varies throughout the flight. This aids in producing adaptation that is smoother then the constant forgetting factor. From this we conclude that the VFF is the more desirable algorithm in comparison to the CFF for this application.

3.3.6 Comparison of RCAC/VFF and RCAC/KF

We revisit the example in Section 3.3.4 to compare RCAC/VFF with RCAC/KF. The VFF tunings are the same as in Section 3.3.5, and the KF tuning is $Q_{\text{KF}}(k) = 0.06$. Fig. 3.4 compares these algorithms. The calculated miss distance (MD) is similar, but unlike the 'g' response, the controller gains are significantly different as shown in the second row of the figures. Notice that the magnitude of the VFF controller gains are smaller than the magnitude of the KF gains and also vary less during flight.The bottom of Fig. 3.4 compares the eigenvalues of the covariance matrix. At around 0.4 sec, covariance values for both algorithms suddenly drop to zero. After this time, the trends look similar but the KF has a larger number of gains that are increasing.

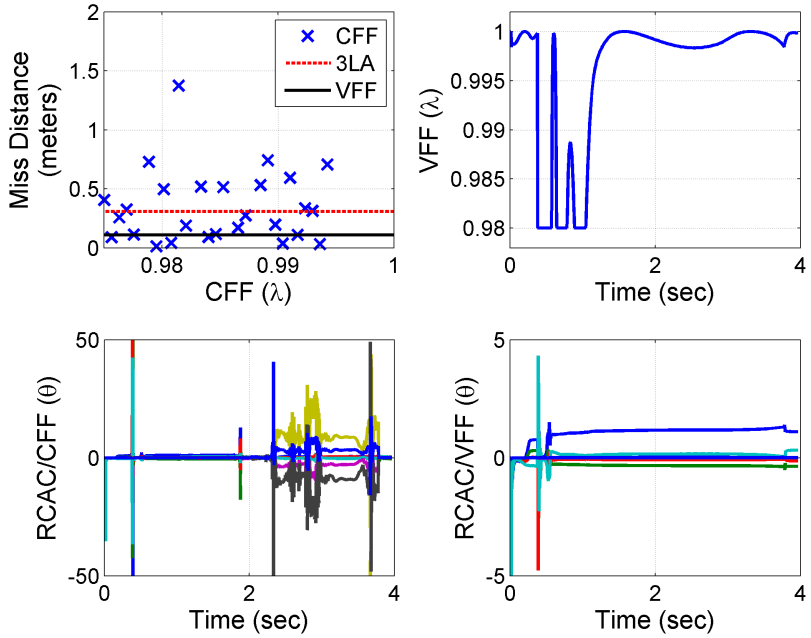


Figure 3.3: Comparison of 3LA, RCAC/CFF, and RCAC/VFF. [Top Left] shows the calculated miss distance for a CFF that is varied from 0.975 to 1.0. The additional horizontal lines represent the miss distance of 3LA and RCAC/VFF with $\Sigma_0 = 1000$, $\lambda_{\min} = 0.98$, and $c = 10\delta$. [Bottom] shows the gains for RCAC/VFF and RCAC/CFF with CFF set to 0.9795, which has the best miss distance performance. [Top Right] shows how the VFF $\lambda(k)$ varies throughout the flight.

3.3.7 Robustness Comparison of 3LA, RCAC/VFF, and RCAC/KF

In this section, we present a final comparison between 3LA, RCAC/VFF, and RCAC/KF by considering four examples to test the robustness of the adaptive algorithms. The RCAC parameter G_{f} is modified to optimize performance through a range of flight Mach numbers. To do this, N_1 is a function of Mach number where $N_1(M) = -0.02M^2 + 0.25M - 0.87$. This modification is a gain-schedule in missile Mach number for increased performance. Unlike the gain-scheduled 3LA, we do not schedule based on angle of attack.

Example 3.1. In this example, the initial missile Mach number is increased to $M_0 = 4$, and the initial rotation angle is decreased to $\theta_0 = 15$ deg with the remaining initial conditions left as before. The top of Fig. 3.5 shows the normal acceleration error of each of the three controllers along with the miss distances. As shown, RCAC/VFF

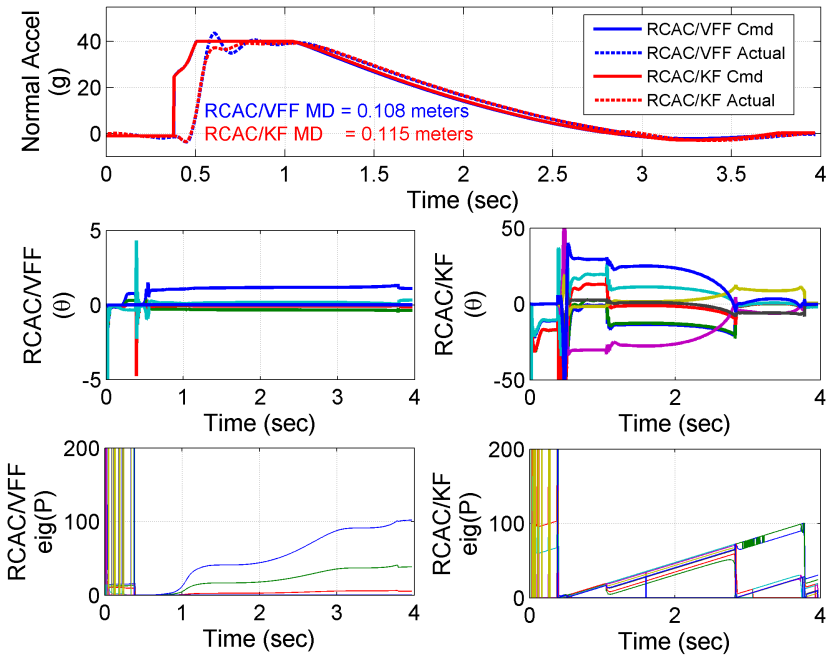


Figure 3.4: Comparison of RCAC/VFF and RCAC/KF. Results shown use the parameters presented in the previous section with the addition of the KF variable $Q_{\text{FK}}(k) = 0.06$. [Top] shows the similarity of the command and response of the two algorithms. The calculated miss distances are also similar, unlike the controller gains shown [Middle]. The magnitudes of the gains in RCAC/KF are much larger and vary more through the flight. [Bottom] shows how both eigenvalues of the covariance matrix drop to zero at time 0.4 sec and steadily increase as the flight progresses to optimize gains.

has the least miss distance but also has the largest amount of overshoot error around 0.5 sec when the maximum 'g' command is given. The overshoot is attributed to this method having the best miss distance value. The remaining two methods have similar responses with the gain-scheduled 3LA having a better miss distance value compared to RCAC/KF. ■

Example 3.2. In this example, the initial missile Mach number is $M_0 = 3$, $\theta_0 = 15$ deg, and the aerodynamic coefficient C_{z_α} in the body Z direction due to the angle-of-attack is scaled by 3, that is $C_{z_{\alpha_n}} = 3C_{z_\alpha}$ with the remaining initial conditions left as before. The middle of Fig. 3.5 shows the normal acceleration error of each of the three controllers as well as their calculated miss distances. In this example, RCAC/KF has the lowest miss distance and provides the best tracking despite this

aerodynamic modification. RCAC/VFF is able to adapt to the modified aerodynamic coefficient but yields the worst miss distance. 3LA struggles with the aerodynamic coefficient modification and response similar to that of a lightly damped system. ■

Example 3.3. In this example, the initial missile Mach number is $M_0 = 2.5$, $\theta_0 = 25$ deg, and the aerodynamic moment coefficient due to the deflection of the control surface C_{M_δ} is scaled by 3, that is, $C_{M_{\delta_n}} = 3C_{M_\delta}$ with the remaining initial conditions left as before. The bottom of Fig. 3.5 shows the normal acceleration error of each controller as well as their calculated miss distances. In this example, RCAC/KF has the lowest miss distance followed by RCAC/VFF. Both of these controllers quickly adapted to the modification in the aerodynamic coefficient and are able to track the normal acceleration command. 3LA struggled with this aerodynamic modification and continually oscillated throughout the entire flight. ■

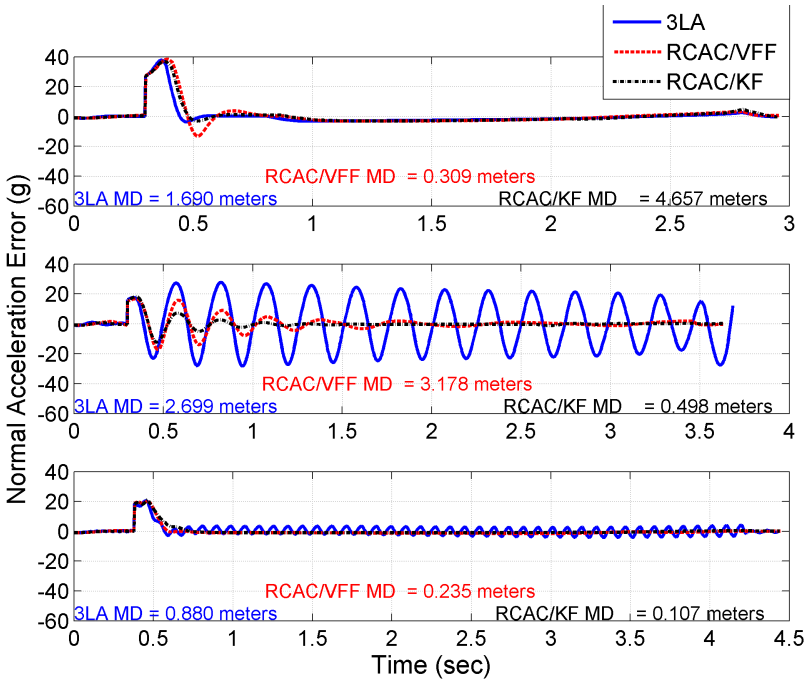


Figure 3.5: Comparison of the normal acceleration error between 3LA, RCAC/VFF, and RCAC/KF on three representative cases. [Top] shows the case where the initial missile Mach number is increased to 4 and θ_0 decreased to 15 deg. [Middle] shows the case where the initial missile Mach number of 3, $\theta_0 = 15$ deg, and the aerodynamic coefficient $C_{z_{\alpha_n}} = 3C_{z_\alpha}$. [Bottom] shows the case where the initial missile Mach number is 2.5, $\theta_0 = 25$ deg, and the aerodynamic coefficient $C_{M_{\delta_n}} = 3C_{M_\delta}$.

Example 3.4. In this example, we revisit the scenario presented in Section 3.3.4 with an unmodeled decrease in the actuator bandwidth from 150 Hz to 40 Hz throughout the flight. In doing this, the optimized design of 3LA will suffer due to the increase in the missiles' time constant. Fig. 3.6 shows the normal acceleration error of each of the three controllers as well as the actuator position and rate, respectively. 3LA struggles with the decrease in actuator performance whereas both the RCAC/VFF and the RCAC/KF are able to dampen out the response and adapt to the lower bandwidth actuator. ■

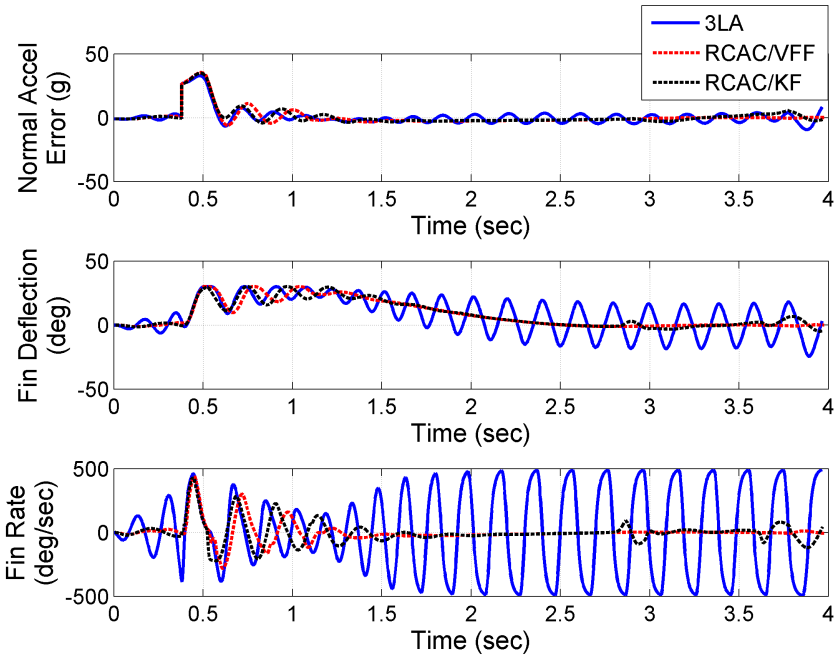


Figure 3.6: Comparison of 3LA, RCAC/VFF, and RCAC/KF with an unmodeled decrease in actuator bandwidth. Nominally the actuator has a bandwidth of 150 Hz, but the results above assume an actuator with a bandwidth of 40 Hz. [Top] The normal acceleration error is shown, while the actuator deflection angle [Middle] and rate [Bottom] are shown.

3.4 Conclusions

We extended [12, 13] in the RCAC formulation by including a variable forgetting factor and Kalman Filter. This extension allowed for continual parameter estimation in the adaptive controller update, a feature that is essential to controlling a system

with nonlinear dynamics. Additionally, an inner-loop/outer-loop control architecture is used to adaptively adjust the pitch-rate command based on the normal acceleration command. The adaptive pitch-rate command loop is appropriate due to the inability to infer such a command when given an arbitrary normal acceleration command. Results show that the CFF, although effective, leads to aggressive adaptation that may lead to instabilities. We also showed that both RCAC/VFF and RCAC/KF allow for gain adapting through the entire flight and were able to track the normal acceleration command as well as the 3LA. The adaptive controller excelled when an aerodynamic coefficient modification was introduced to the system as well as having a lower bandwidth actuator.

CHAPTER 4

Kalman-Filter-Based System Identification via Retrospective Optimization of the Process Noise Covariance

4.1 Introduction

The Kalman filter is a stochastically optimal observer that utilizes statistical information about the measurement and process noise. In particular, the measurement and process noise are assumed to be white Gaussian processes with zero mean and covariances R and Q, respectively. Although R is relatively easy to characterize in practice, it is often challenging to estimate Q. Consequently, there is an extensive amount of literature on adaptive Kalman filtering, where the innovations signal is used to construct an estimate of Q [14–17].

In the present chapter, we develop an alternative approach to estimating Q and apply this technique to parameter estimation. First, we show that the process noise covariance Q that minimizes the cumulative innovations does not (in general) correspond to the value of Q that, when used in the Kalman filter, minimizes the cumulative state-estimate error. However, for parameter estimation, we show that under certain assumptions the value of Q that minimizes the cumulative state-estimate error does in fact minimize the cumulative innovations. This insight provides the basis for

a novel approach to online estimation of the process noise covariance for parameter estimation problems.

The estimation technique uses a past window of data to rerun the Kalman filter along with an optimization routine that retrospectively determines the value of Q that minimizes the cumulative innovations at the present time. Based on the previous analysis, this estimate is assumed to minimize the state-estimate error. Since the optimization does not lend itself to analytical gradients, we use gradient-free optimization techniques.

This technique is applied to the system identification problem, specifically, estimation of the coefficients of an input-output model. The dynamical system state components for the Kalman filter are thus the coefficients of the input-output model, and the dynamics matrix is the identity. The ultimate objective is to use this technique to identify linear systems whose coefficients are time-varying.

Adaptive Kalman filter techniques have been developed for this sort of problem in [14–16, 18, 19], where the covariances R and Q are determined in real-time. These methods are also innovations-based and are not derived in the sense of a minimization optimization problem of the innovation itself. Rather, they are based on an averaging inside a moving estimation window. Another method commonly used for covariance identification is the maximum likelihood estimation (MLE). This method assumes that the noise is characterized by a Gaussian model and that the best fit is determined by minimizing the error for the given noise structure [19]. The type of parameter uncertainty investigated in the present chapter are only constrained in the structure of Q.

4.2 The State Estimation Problem

Consider state estimation for the linear system

$$x_{k+1} = A_k x_k + w_k, \tag{4.1}$$

$$y_k = C_k x_k + v_k, \tag{4.2}$$

where $x_k \in \mathbb{R}^n$ is the state to be estimated, $w_k \in \mathbb{R}^n$ is the process noise, $y_k \in \mathbb{R}^m$ is the measurement, and $v_k \in \mathbb{R}^m$ is the measurement noise. We assume that the measurement and process noise are stationary and have zero mean with covariances $R = \mathbb{E}[v_k v_k^{\mathrm{T}}]$ and $Q = \mathbb{E}[w_k w_k^{\mathrm{T}}]$, respectively. Assuming (A_k, C_k) is observable, the Kalman filter is given by

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k}, \tag{4.3}$$

$$= A_k(\hat{x}_{k|k-1} + K_k z_k), \tag{4.4}$$

$$P_{k+1|k} = A_k P_{k|k} A_k^{\mathrm{T}} + Q, \tag{4.5}$$

$$= A_k[(I - K_k C_k)P_{k|k-1}]A_k^{\mathrm{T}} + Q, \tag{4.6}$$

where the error covariance is defined as

$$P_{k|k-1} \triangleq \mathbb{E}[(x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})^{\mathrm{T}}], \tag{4.7}$$

and where

$$K_k \triangleq P_{k|k-1} C_k^{\mathrm{T}} (C_k P_{k|k-1} C_k^{\mathrm{T}} + R)^{-1}, \tag{4.8}$$

$$z_k \triangleq y_k - C_k \hat{x}_{k|k-1}, \tag{4.9}$$

are the optimal gain and innovations, respectively [52].

In the present chapter, we use the innovations to estimate the process noise covariance. We define the cumulative state-estimate error as

$$e_{x,k+k_0} \triangleq \sum_{i=k_0}^{k+k_0} (x_i - \hat{x}_{i|i-1})^{\mathrm{T}}(x_i - \hat{x}_{i|i-1}), \tag{4.10}$$

with expected value

$$\begin{aligned}\mathbb{E}[e_{x,k+k_0}] &= \sum_{i=k_0}^{k_0+k} \operatorname{tr}(P_{i|i-1}), \\ &= \sum_{i=k_0}^{k+k_0} \operatorname{tr}(A_i P_{i-1|i-1} A_i^{\mathrm{T}}) + k \operatorname{tr}(Q).\end{aligned} \tag{4.11}$$

Next, we define the cumulative innovations as

$$e_{z,k+k_0} \triangleq \sum_{i=k_0}^{k+k_0} z_i^{\mathrm{T}} z_i, \tag{4.12}$$

with expected value

$$\begin{aligned}\mathbb{E}[e_{z,k+k_0}] &= \sum_{i=k_0}^{k+k_0} \mathbb{E}[(y_i - C_i\hat{x}_{i|i-1})^{\mathrm{T}}(y_i - C_i\hat{x}_{i|i-1})], \\ &= \sum_{i=k_0}^{k+k_0} \operatorname{tr}(C_i(A_i P_{i-1|i-1} A_i^{\mathrm{T}} + Q)C_i^{\mathrm{T}}) + k\operatorname{tr}(R).\end{aligned} \tag{4.13}$$

Note that, since the state error at step i is independent of the measurement noise at step i, it follows that $\mathbb{E}[(x_i - \hat{x}_{i|i-1})^{\mathrm{T}} C_i^{\mathrm{T}} v_i] = \mathbb{E}[(x_i - \hat{x}_{i|i-1})^{\mathrm{T}}] C_i^{\mathrm{T}} \mathbb{E}[v_i] = 0$. Also, note that (4.11) and (4.13) both depend on Q due to (4.5). Furthermore, (4.11) and (4.13) are closely related since both involve $P_{i|i-1}$. However, unless $C_i = I_n$, there is no guarantee that minimizing (4.10) implies that (4.12) is minimized and vice versa.

4.3 The Parameter Estimation Problem

In this section, we specialize the state estimation problem to a problem of parameter estimation, which is subsequently applied to system identification. We show that, under some assumptions, the value of Q that minimizes the cumulative state-estimate error also minimizes the cumulative innovations for the parameter estimation problem. In particular, consider the parameter estimation problem

$$\theta_{k+1} = \theta_k + w_k, \tag{4.14}$$

$$y_k = \phi_k^{\mathrm{T}} \theta_k + v_k, \tag{4.15}$$

where $\theta_k \in \mathbb{R}^n$ are the parameters to be estimated, $y_k \in \mathbb{R}^m$ are the measurements, $\phi_k \in \mathbb{R}^{n \times m}$ is the regression matrix, $v_k \in \mathbb{R}^m$ is the measurement noise, and $w_k \in \mathbb{R}^n$ is the process noise. Note that (4.14), (4.15) is a special case of (4.1), (4.2) with $A_k = I_n$ and $C_k = \phi_k^{\mathrm{T}}$. Furthermore, if the parameters to be estimated are constant, that is, $w_k = 0$, and the measurement noise $v_k \sim \mathcal{N}(0, 1)$, then the parameter estimation problem becomes the standard recursive least-squares problem [18]. For the case where the parameters are time-varying, that is $w_k \neq 0$, we write (4.3) – (4.9) recursively as

$$\hat{\theta}_{k+1|k} = \hat{\theta}_{k|k-1} + K_k z_k, \tag{4.16}$$

$$P_{k+1|k} = (I - K_k \phi_k^{\mathrm{T}}) P_{k|k-1} + Q, \tag{4.17}$$

where

$$K_k = P_{k|k-1} \phi_k (\phi_k^{\mathrm{T}} P_{k|k-1} \phi_k + R)^{-1}, \tag{4.18}$$

$$z_k = y_k - \phi_k^{\mathrm{T}} \hat{\theta}_{k|k-1}. \tag{4.19}$$

Since $A_k = I_n$, the expected value of the cumulative state-estimate error (4.11) and the expected value of the cumulative innovations (4.13) becomes

$$\mathbb{E}[e_{\theta,k+k_0}] = \sum_{i=k_0}^{k+k_0} \operatorname{tr}(P_{i-1|i-1}) + k \operatorname{tr}(Q), \tag{4.20}$$

and

$$\mathbb{E}[e_{z,k+k_0}] = \sum_{i=k_0}^{k+k_0} \operatorname{tr}(\phi_i^{\mathrm{T}}(P_{i-1|i-1} + Q)\phi_i) + k \operatorname{tr}(R), \tag{4.21}$$

respectively. Within the context of parameter estimation, the notion of observability depends on persistent excitation [18]. To show that the same value of Q minimizes (4.20) and (4.21), we introduce the following assumptions.

Assumption 1. The measurement noise covariance R is known.

Assumption 2. There exists a positive integer N such that the regressor matrix ϕ_k satisfies

$$\beta_l I \leq \sum_{i=n}^{n+N} \phi_i \phi_i^{\mathrm{T}} \leq \beta_u I, \tag{4.22}$$

where β_l and β_u are positive constants [18].

Assumption 3. The process noise w_k is uncorrelated at each time step k.

In practice, Assumption 1 can be met by performing a sensor characterization. Assumption 2 requires that the system whose parameters are to be identified is persistently excited and hence (I_n, ϕ_k) is observable. Assumption 3 states that the process noise is uncorrelated at each time step, thus making Q diagonal.

Lemma 1. Consider the parameter estimation problem (4.14), (4.15) that satis-

fies Assumptions 1-3. Then, the value of Q that minimizes the expected value of the cumulative state-estimate error (4.20) also minimizes the expected value of the cumulative innovations (4.21).

Proof. For all $k \geq N$, the expected value of the cumulative innovations (4.21) is written as

$$\begin{aligned}
\mathbb{E}[e_{z,k_0+k}] &= \sum_{i=k_0}^{k+k_0} \operatorname{tr}((P_{i-1|i-1} + Q)\phi_i\phi_i^{\mathrm{T}}) + k \operatorname{tr}(R), \\
&= \sum_{i=k_0}^{k+k_0} \operatorname{tr}(P_{i-1|i-1}\phi_i\phi_i^{\mathrm{T}}) + \operatorname{tr}(Q\phi_i\phi_i^{\mathrm{T}}) + k \operatorname{tr}(R), \\
&\geq \sum_{i=k_0}^{k+k_0} \operatorname{tr}(P_{i-1|i-1}\phi_i\phi_i^{\mathrm{T}}) + k\beta_l \operatorname{tr}(Q) + k \operatorname{tr}(R), \qquad (4.23)
\end{aligned}$$

which is a linear combination of the expected value of the cumulative state estimate (4.20) and the innovations matrix $\phi_i\phi_i^{\mathrm{T}}$. Since the sum of N innovation matrices are full rank, it follows that the value of Q that minimizes (4.20) also minimizes (4.23).

In the next example, we examine the residuals of the state and innovations for a parameter estimation problem.

Example 4.1. Consider the ARMAX model described in (4.14), (4.15) as

$$\theta_{k+1} = \theta_k + w_k, \qquad (4.24)$$

$$y_k = \begin{bmatrix} u_{k-1} & u_{k-2} & y_{k-1} & y_{k-2} \end{bmatrix} \theta_k + v_k, \qquad (4.25)$$

where $\theta_k \in \mathbb{R}^4$ with initial condition $\theta_0 = [0\ 0\ 0\ 0]^{\mathrm{T}}$. The input u_k is a zero-mean Gaussian with standard deviation of 1. The true process noise w_k has zero mean with covariance $Q = 2.5 \times 10^{-5} I_4$. We assume that $v_k \sim \mathcal{N}(0, R)$, where $R = 0.01$ is known. Fig. 4.1 shows the cumulative state-estimate error and the cumulative innovations as functions of α, where $Q = \alpha I_4$. The simulation is run for 6000 time steps for varying

values of α. The minimizing values of α for the cumulative state-estimate error and cumulative innovations are 2.70×10^{-5} and 2.37×10^{-5}, respectively. Note that since the parameter estimation problem (4.25) is a statistical process, the estimate of Q becomes more accurate as the amount of data increases and the value of α will converge to the true value 2.5×10^{-5}. ■

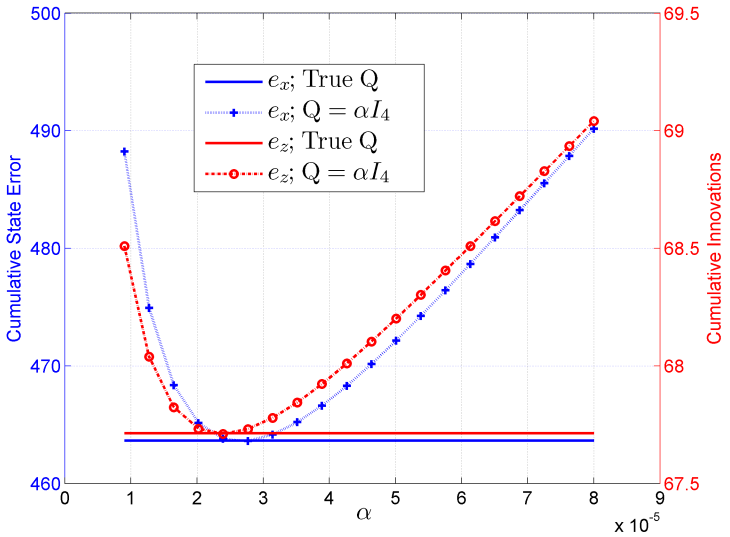


Figure 4.1: Results for Example 4.1. The cumulative state-estimate error and cumulative innovations as a function of the entry α of Q for Example 4.1. Each simulation is run for 6000 time steps for varying values of α. The minimizing value of α for the cumulative state-estimation error and the cumulative innovations correspond to the true value $Q = 2.5 \times 10^{-5} I$. The cumulative state-estimation error achieves its minimum at $\alpha = 2.70 \times 10^{-5}$, while the cumulative innovations achieves its minimum at $\alpha = 2.37 \times 10^{-5}$.

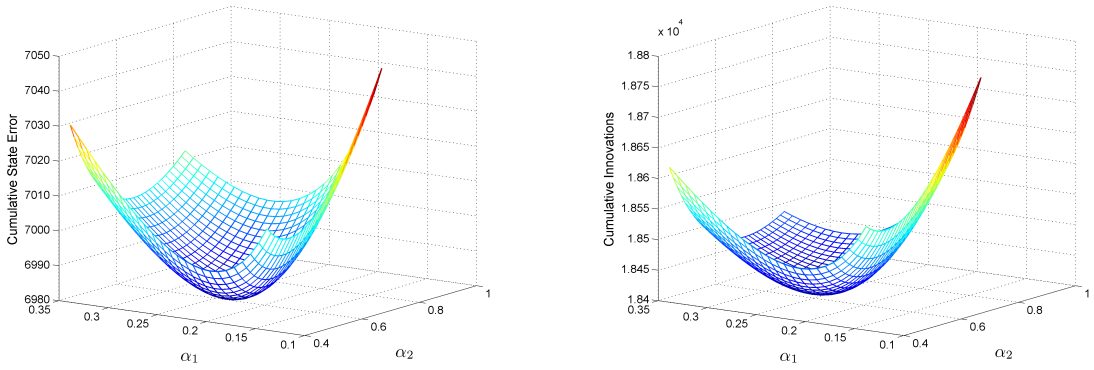
Example 4.2. Consider the finite impulse response (FIR) filter described in (4.14), (4.15) as

$$\theta_{k+1} = \theta_k + w_k, \tag{4.26}$$

$$y_k = [u_{k-1} \quad u_{k-2}]\theta_k + v_k, \tag{4.27}$$

where $\theta_k \in \mathbb{R}^2$ with parameter initial condition $\theta_0 = [0 \quad 0]^{\mathrm{T}}$. The system input u_k is a white noise sequence with zero mean and standard deviation of 1. The true

process noise w_k has zero mean with the covariance matrix $Q = \text{diag}([0.25 \quad 0.64])$. We assume that the statistical properties of the measurement are known and that $v_k \sim \mathcal{N}(0, R)$, with $R = 1$. Fig. 4.2 shows the cumulative state-estimate error and cumulative innovations as a function of $\alpha = [\alpha_1 \quad \alpha_2]$, where $Q = \text{diag}(\alpha)$. The simulation is run for 5000 time steps for varying values and combinations of α. The minimizing values of α for the cumulative state-estimate error and the cumulative innovations is $\alpha = [0.225 \quad 0.578]$ and $\alpha = [0.25 \; 0.62]$, respectively. Note that since the parameter estimation problem (4.27) is a statistical process, the estimate of Q becomes more accurate as the amount of data increases and will converge to the true value of $[0.25 \quad 0.64]$. ■



(a) (b)

Figure 4.2: Results for Example 4.2. The cumulative state-estimate error and cumulative innovations as a function of the entries α_1 and α_2 of Q for Example 4.2. Each simulation is run for 5000 time steps for varying values and combinations of α. (a) shows the cumulative state-estimate error as a function of α, with the minimizer $\alpha = [0.225 \quad 0.578]$. (b) shows the cumulative innovations, with the minimizer $\alpha = [0.25 \quad 0.62]$.

4.4 Innovations-Based Sliding Window-Q Optimization

The innovations-based sliding window-Q optimization (ISW-QO) is a method that minimizes the cumulative innovations by using a retrospective optimization to update the process noise covariance at each time step. Note that a version of this algorithm

has been mentioned in [53] for state estimation but no formal proof of convergence is given nor is it used for online estimation. Thus, the problem is to minimize

$$J(Q) = \sum_{j=j_0}^{k} z_j^{\mathrm{T}} z_j, \tag{4.28}$$

$$\text{s.t. } Q \succeq 0, \tag{4.29}$$

where $j_0 = k-N+1$, and N is the window size determined by the user that guarantees Assumption 2. Fig 4.3 illustrates the method via a flow diagram. The Kalman filter is first initialized and is run for $N-1$ time steps. At time step $k = k+1$, the initial estimates $\hat{\theta}$, error covariance P, input u, and measurement y for that interval is sent to the optimizer. The current value of Q_k is used as an initial guess in the optimizer. In Fig 4.3, the data $y_{k-N+1:k} = [y_{k-N+1} \ \dots \ y_k]$ and $u_{k-N+1:k} = [u_{k-N+1} \ \dots \ u_k]$. Once the optimal Q is found, the optimization routine passes the optimized parameter estimates, error covariance, and Q to the Kalman filter to process the next step. The index k is incremented and the process loops again.

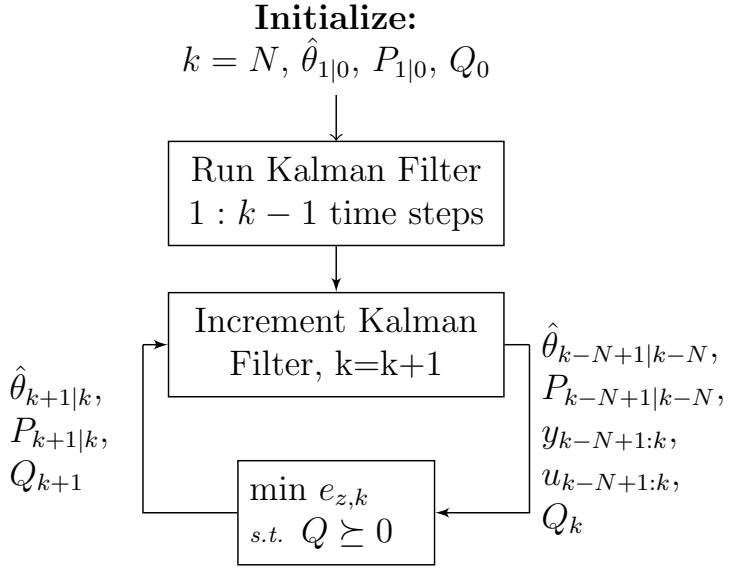


Figure 4.3: Block diagram illustrating the innovation-based sliding window Q optimization (ISW-QO) iteration process.

For the following examples, we use the gradient-free optimization method *fminsearch* in MATLAB® to optimize Q based on the cumulative innovations. This method is a multidimensional unconstrained nonlinear minimization routine [54]. Since this method is unconstrained, we must impose the positive semidefiniteness of Q. First, Q is enforced to have a diagonal structure, e.g. $Q = \text{diag}(\alpha)$, where $\alpha = [\alpha_1 \ \dots \ \alpha_n]$. Second, at every iteration of the algorithm, the diagonal entries of Q are squared to ensure positive semidefiniteness.

Example 4.3. We again consider the FIR filter presented in Example 4.2. To test the ISW-QO method, we allow the optimizer to calculate Q based on the cumulative innovations with a window size of 800 time steps, running the simulation for a total of 6000 time steps. Fig. 4.4 shows the optimal α for each time step and a comparison between the true and the estimated parameters. Note that in this example, α is initialized at [0.3 0.81] and is used in the optimizer until $k = 800$. After $k = 800$, the optimizer updates α at each time step. The true value of α is shown as reference. Note that the window used for optimization is only 800 time steps, whereas in Example 4.2, the optimization uses the entire 6000 time steps. Although the optimized values of α do not converge to the true values, the cumulative state-estimation error is 1.1×10^4 whereas, if the true value of Q were used, the cumulative state-estimation error is 7.0×10^3 after the 6000 time steps. ■

Example 4.4. In this example, we compare a static Q Kalman filter to the ISW-QO method to the same FIR filter as in Example 4.2, except with $R = 0.01$. The parameters are initialized at $\theta_0 = [-0.8 \ \ 1.0]^{\text{T}}$ and there is no process noise added to the system, that is $w_k = 0$. The parameter θ_1 induces a sudden step change at $k = 250$ from -0.8 to 0.1, while θ_2 remains constant at 1. Note that unless the exact time of the parameter step change is known, their is no way to create an accurate noise model of the parameter variation. Thus, implementing $Q = 0_{2\times 2}$ is the best option

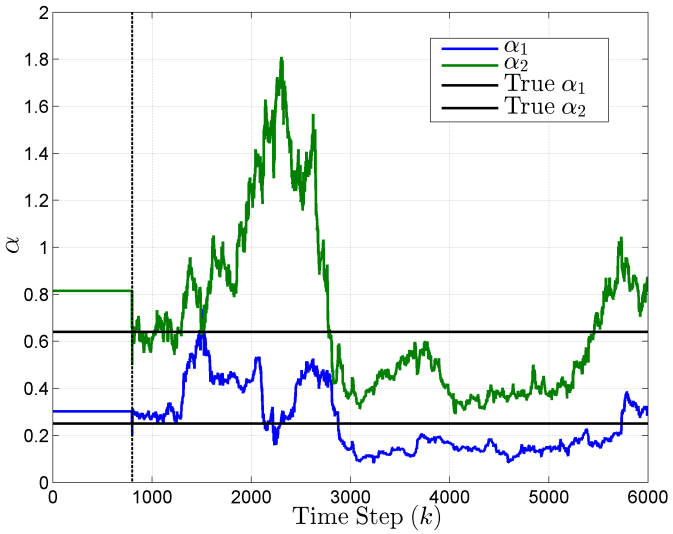
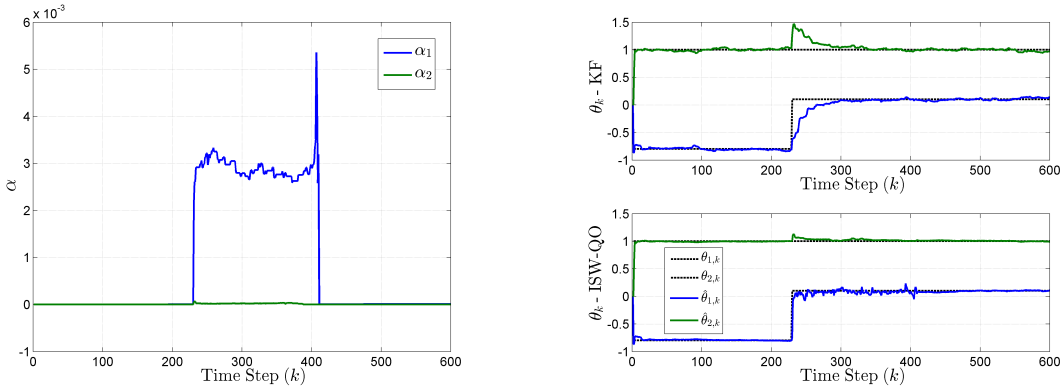


Figure 4.4: Results for Example 4.3. The optimized values of α and the parameters θ_k and $\hat{\theta}_k$ for Example 4.3. The Q optimizer is run with a window size of 800 time steps after the initial 800 time steps. In this example, the true value of Q is held constant throughout the entire simulation. The true α and the optimized value of α are shown.

and leads to recursive least squares (RLS), which has slow convergence properties. In this example, the Kalman filter is initialized with the same initial conditions as the ISW-QO method except that the process noise covariance is constant, that is $Q = 1 \times 10^{-4} I_2$, with an anticipation that a parameter might change. The ISW-QO is initialized at $Q = 0_{2\times 2}$ with a sliding window of 180 time steps. Fig. 4.5 shows how the parameters α evolve over the length of the simulation as well as how the parameter estimates compare to the actual values. Notice the correlation between the parameter that changes θ_1 and the covariance value that changes α_1. ISW-QO correctly identifies the parameter in Q, namely α_1, that aided in reducing the innovations and since the sliding window is of size $N = 180$ time steps, the value of α_1 remains active until the step outside the window. Also notice that with a small disturbance in $\hat{\theta}_2$, $\hat{\theta}_1$ converges faster to the true parameter than the Kalman filter with an added small process noise covariance. ■

Example 4.5. In this example, we revisit Example 4.4 but allow the parameters



(a) (b)

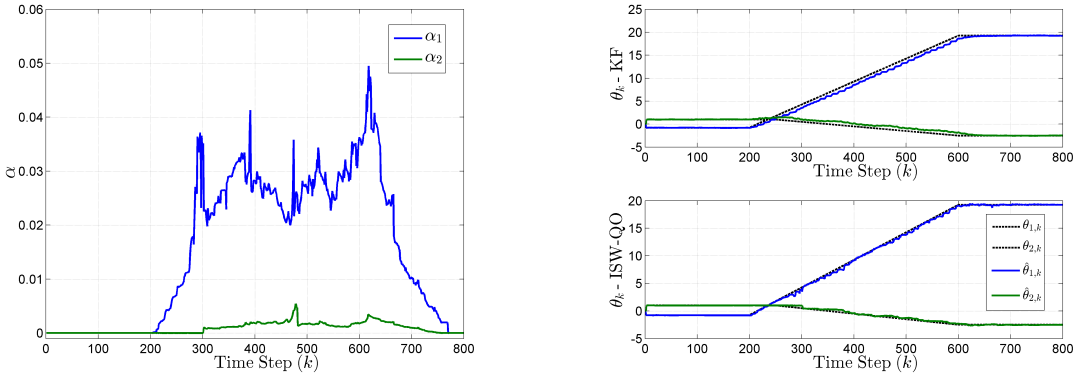
Figure 4.5: Results for Example 4.4. The optimized values of α and the parameters θ_k and $\hat{\theta}_k$ for Example 4.4. (a) shows the optimized Q parameters with α_1 increased due to the step change in the parameter θ_1. (b) compares the Kalman filter to ISW-QO estimates. Notice the convergence of $\hat{\theta}_1$ and the minimal disruption to $\hat{\theta}_2$ in ISW-QO compared to the Kalman Filter with constant $Q = 1 \times 10^{-4} I_2$.

θ_1 and θ_2 to ramp. At $k = 200$, the parameters ramp with a slope of 0.05 and -0.01, respectively and remain constant after $k = 600$ at $\theta_1 = 19.25$ and $\theta_2 = -3.01$. Figure 4.6 shows the how the parameters in Q vary during and after the course of the ramp as well as how the estimated parameters track the true values. Notice that the Kalman filter estimation looks like a delayed version of the true parameters, unlike the estimated parameters of the ISW-QO method, which tracks the ramp. ■

4.5 Innovations-based Adaptive Kalman filter

The innovations-based adaptive Kalman filter (IAKF) adapts the covariance matrices as the measurements evolve with time [14–17]. In this method, the estimate of Q is

$$\hat{Q}_k = \frac{1}{N} \sum_{j=j_0}^{k} \Delta x_j \Delta x_j^{\mathrm{T}} + P_{k|k} - A_{k-1} P_{k-1|k-1} A_{k-1}^{\mathrm{T}}, \tag{4.30}$$



(a) (b)

Figure 4.6: Results for Example 4.5. The optimized values of α and the estimated and true parameters $\hat{\theta}_k$ and θ_k, respectively for Example 4.5. (a) shows the evolution of α_1 and α_2 due to a ramp increase in both the θ parameters, with the magnitude of α_1 being greater due to the more aggressive ramp. (b) compares the Kalman filter to ISW-QO estimates. The Kalman filter looks like a delayed version of the true parameters, unlike the estimated parameters of ISW-QO, which tracks the ramp.

where Δx_k is a first order, steady-state term [16] defined as

$$\Delta x_k \triangleq K_k z_k, \tag{4.31}$$

and $j_0 = k - N + 1$ is the first sample within the estimation of moving window size N. Note that in the parameter estimation problem, $A_k = I_n$, and (4.30) reduces to

$$\hat{Q}_k = \frac{1}{N} \sum_{j=j_0}^{k} \Delta x_j \Delta x_j^{\mathrm{T}} + P_{k|k} - P_{k-1|k-1}. \tag{4.32}$$

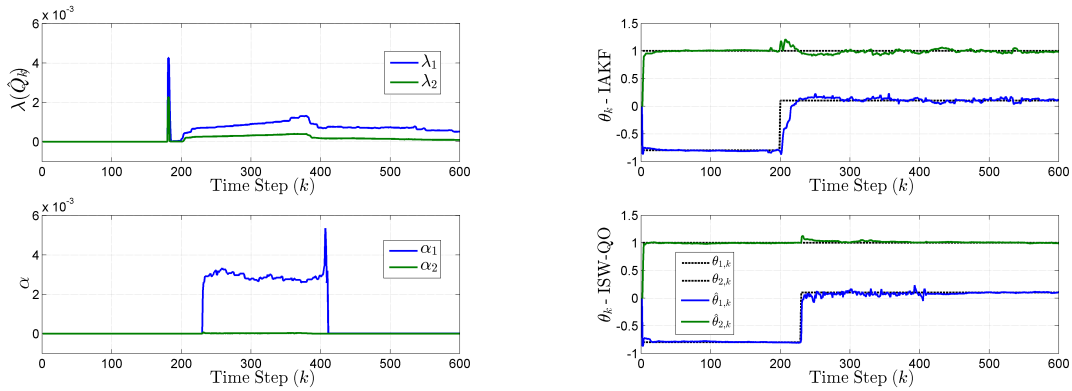
In this method, it is also assumed that the measurement covariance R is known but the structure of Q is not forced to be diagonal.

4.6 ISW-QO Compared to IAKF

We compare ISW-QO and IAKF by revisiting Examples 4.4 and 4.5. In these examples, both ISW-QO and IAKF have the same moving window size of 180 time steps. Note that unlike ISW-QO, where the user has the ability to define the structure

of Q to be optimized, IAFK does not have this feature. In order to compare how the estimate $\hat{Q}$ evolves over the length of the simulation, we chose to compare the eigenvalues $\lambda(\hat{Q}_k)$ of $\hat{Q}_k$, to the values of α_k in ISW-QO where $Q_k = \text{diag}(\alpha_{1,k}, \alpha_{2,k})$. This comparison is used to identify which parameters IAKF views as having the largest uncertainty.

Example 4.6. In this example, we compare ISW-QO to IAKF for a sudden step change in θ_1. Figure 4.7 compares the evolution of Q along with the parameter estimates. Notice that the two methods have different eigenvalue profiles but both identify which value needs to be greater in order to minimize the innovations. The parameter estimates for IAKF take longer to converge to the true parameters than ISW-QO but has a faster response then the Kalman filter in Fig. 4.5. The cumulative state-estimation errors for the full 600 time steps for IAKF and ISW-QO are 33.2 and 19.2, respectively. ■

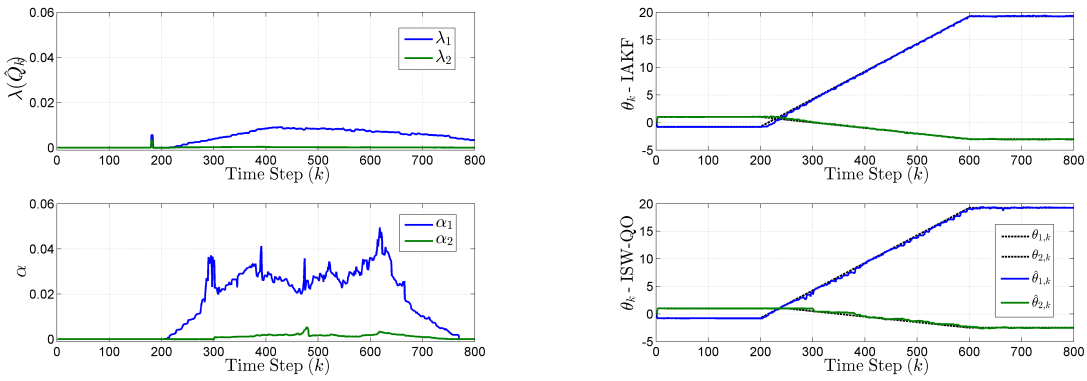


(a) (b)

Figure 4.7: Results for Example 4.6. A comparison of ISW-QO to IAKF for a sudden step change in θ_1. (a) shows how the eigenvalue profiles are different but share in common which value is greater. (b) shows the parameter estimates of the two methods compared to the true values. ISW-QO has a faster response but IAKF performs better compared to the Kalman filter in Fig. 4.5. The cumulative state-estimation errors for the full 600 time steps for IAKF and ISW–QO is 33.2 and 19.2, respectively.

Example 4.7. In this example, we compare ISW-QO to IAKF for a ramp change in both θ_1 and θ_2. Fig. 4.8 shows the evolution of Q as well as the parameter esti-

mates. Although the methods produce different eigenvalue profiles, they both identify which value needs to be greater in order to minimize the innovations. The parameter estimates for IAKF are smoother than those of ISW-QO and the cumulative state-estimation errors for the full 800 time steps for IAKF and ISW-QO are 104.8 and 156.7, respectively. ■



(a) (b)

Figure 4.8: Results for Example 4.7. A comparison ISW-QO and IAKF for a ramp change in both of the parameters θ_1 and θ_2. (a) shows how the eigenvalue profiles are different but both identify which value is needed to be greater to minimize the innovations. (b) shows the parameter estimates of the two methods compared to the true values. The cumulative state-estimation errors for the full 800 time steps for IAKF and ISW-QO are 104.8 and 156.7, respectively.

4.7 Conclusions

In this chapter, we proposed a technique that minimizes the innovations based on retrospective optimization of the process noise covariance. We showed that minimizing the cumulative innovations is equivalent to minimizing the cumulative state-estimation error for the parameter estimation problem under certain assumptions. This technique is applied to system identification problems where the parameters to be estimated can be time-varying and thus have an unknown Q value. We compared our technique to the standard Kalman filter and IAKF. Results show that when applied to time-varying parameter estimation problem, this technique performs as well if not better than IAKF.

CHAPTER 5

Batch Retrospective Cost Adaptive Control using Concurrent Controller and Target-Model Optimization

5.1 Introduction

Although the overarching motivation for using feedback control versus open-loop control is the ability to overcome uncertainty, feedback control depends on a model of the plant in order to operate reliably and without risking instability. Assuming an exact and complete model, LQG can be used to stabilize all MIMO plants with optimal H_2 performance regardless of plant order, open-loop pole and zero locations, and channel coupling. In practice, however, uncertainty may be unavoidable due to complex, unknown, or unpredictably changing physics. To overcome model uncertainty, robust control techniques can be used to guarantee stability and performance, albeit at the expense of performance. Adaptive control can be viewed as a form of robust control, wherein the control law adjusts itself to the plant during operation, thereby circumventing the performance sacrifice inherent in robust control.

A fundamental goal of feedback control is to achieve maximal closed-loop performance in the presence of prior model uncertainty. In the case of adaptive control, closed-loop performance must account for transient performance as the controller ad-

justs itself to the actual plant characteristics. For example, universal adaptive control laws [20] can adapt to uncertainty in the sign of the leading coefficient of the plant transfer function, although the transient response may be large.

Nonminimum-phase (NMP) zeros also present a challenge to adaptive control; for example, the control laws in [1–3] assume that the plant is minimum phase. Adaptive control of NMP plants is considered in [8, 21–24]. In [8, 24] knowledge of the NMP zeros is embedded in the target model G_{f}, which is used to filter the past data in order to retrospectively optimize the controller coefficients.

The goal of this chapter is to extend retrospective cost adaptive control (RCAC) as presented in [8, 24] to alleviate the need for prior modeling of both the sign of the leading coefficient of the plant transfer function as well as its NMP zeros. The key element of this extension is concurrent optimization of both the target model and the controller coefficients. In particular, we show that concurrent optimization facilitates the application of RCAC with less prior modeling information than is assumed in [8, 24]. In particular, the number and location of the NMP zeros need not be known aside from the parity of the number of positive NMP zeros.

Concurrent optimization of the target model and controller coefficients is a quadratic optimization problem in the target model and controller coefficients separately. However, this optimization problem is not convex as a joint function of both variables, and therefore nonconvex optimization methods are needed. In the present chapter, we address this problem in two different ways. First we take advantage of the biquadratic structure of the cost function by applying an alternate convex search algorithm [25]. Related techniques for biconvex and bilinear optimization are given in [26–29]. For comparison, the Matlab fminsearch routine is used to simultaneously optimize the controller and target model coefficients.

5.2 Problem Formulation

Consider the SISO discrete-time system

$$x(k+1) = Ax(k) + Bu(k) + D_1 w(k), \tag{5.1}$$

$$y(k) = Cx(k) + D_2 w(k), \tag{5.2}$$

$$z(k) = E_1 x(k) + E_0 w(k), \tag{5.3}$$

where $x(k) \in \mathbb{R}^n$ is the state, $y(k) \in \mathbb{R}$ is the measurement, $u(k) \in \mathbb{R}$ is the control input, $w(k) \in \mathbb{R}$ is the exogenous input, and $z(k) \in \mathbb{R}$ is the measured performance variable. The goal is to develop an adaptive output feedback controller that minimizes z in the presence of the exogenous signal w with limited modeling information about (5.1)–(5.3). The components of w can represent either command signals to be followed, external disturbances to be rejected, or both, depending on the choice of D_1 and E_0. Depending on the application, components of w may or may not be measured. These components are included in y by suitable choice of C and D_2. No assumptions are made concerning the state space realization since RCAC requires only input-output model information.

5.3 RCAC Algorithm

In this section, we introduce the RCAC algorithm for use on systems that are strictly single-input, single-output (SISO).

5.3.1 Controller Structure

The adaptive control algorithm is constructed as a strictly proper time-series dynamic compensator of order n_c, such that the control $u(k)$ is given as

$$u(k) = \sum_{i=1}^{n_c} P_i(k)u(k-i) + \sum_{i=1}^{n_c} Q_i(k)z(k-i), \tag{5.4}$$

where $P_i(k), Q_i(k) \in \mathbb{R}$ are the controller coefficients. In terms of the Z-transform variable $\mathbf{z}$, the transfer function of the controller from z to u is given by

$$G_{\mathrm{c}}(\mathbf{z}) = \left(\mathbf{z}^{n_{\mathrm{c}}} - \mathbf{z}^{n_{\mathrm{c}}-1}P_1 - \cdots - P_{n_{\mathrm{c}}}\right)^{-1} \cdot \left(\mathbf{z}^{n_{\mathrm{c}}-1}Q_1 + \cdots + Q_{n_{\mathrm{c}}}\right). \tag{5.5}$$

In this chapter we focus on SISO controllers, and hence G_{c} can be written as

$$G_{\mathrm{c}}(\mathbf{z}) = \frac{Q_1\mathbf{z}^{n_{\mathrm{c}}-1} + \cdots + Q_{n_{\mathrm{c}}}}{\mathbf{z}^{n_{\mathrm{c}}} - P_1\mathbf{z}^{n_{\mathrm{c}}-1} - \cdots - P_{n_{\mathrm{c}}}}. \tag{5.6}$$

Note that (5.5) is an infinite impulse response (IIR) controller. The controller (5.4) can be expressed as

$$u(k) = \phi(k)\theta(k), \tag{5.7}$$

where the regressor matrix $\phi(k)$ is defined as

$$\phi(k)^{\mathrm{T}} \triangleq \begin{bmatrix} u(k-1) \\ \vdots \\ u(k-n_{\mathrm{c}}) \\ z(k-1) \\ \vdots \\ z(k-n_{\mathrm{c}}) \end{bmatrix} \in \mathbb{R}^{l_\theta},$$

$$\theta(k) \triangleq \begin{bmatrix} P_1(k) \cdots P_{n_{\mathrm{c}}}(k) \; Q_1(k) \cdots Q_{n_{\mathrm{c}}}(k) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{l_\theta},$$

where $l_\theta \triangleq 2n_{\mathrm{c}}$.

5.3.2 Retrospective Performance

The *retrospective control* is defined as

$$\hat{u}(k, \hat{\theta}) \triangleq \phi(k)\hat{\theta}, \tag{5.8}$$

where $\hat{\theta} \in \mathbb{R}^{l_\theta}$ is determined by the optimization in Section 5.3.3. In terms of the forward shift operator $\mathbf{q}$, the corresponding *retrospective performance variable* is defined as

$$\hat{z}(k, \hat{\theta}) \triangleq z(k) + G_{\mathrm{f}}(\mathbf{q}) \left[\hat{u}(k, \hat{\theta}) - u(k) \right], \tag{5.9}$$

where the target model $G_{\mathrm{f}}(\mathbf{q})$ is a finite impulse response (FIR) filter written as

$$G_{\mathrm{f}}(\mathbf{q}) = \hat{N}_{n_{\mathrm{f}}} \mathbf{q}^{\mathrm{n_f}-1} + \cdots + \hat{N}_1, \tag{5.10}$$

where the filter order $n_{\mathrm{f}} \geq 1$, and $\hat{N}_i \in \mathbb{R}$ for $1 \leq i \leq n_{\mathrm{f}}$. The *extended performance vector* $Z(k) \in \mathbb{R}^p$, the *extended control vector* $U(k) \in \mathbb{R}^p$, and the *extended regressor* $\Phi(k) \in \mathbb{R}^{p \times l_\theta}$ are defined as

$$Z(k) \triangleq \begin{bmatrix} z(k) \\ \vdots \\ z(k-p-1) \end{bmatrix}, \; U(k) \triangleq \begin{bmatrix} u(k) \\ \vdots \\ u(k-p-1) \end{bmatrix},$$

$$\Phi(k) \triangleq \begin{bmatrix} \phi(k) \\ \vdots \\ \phi(k-p-1) \end{bmatrix},$$

where $p \triangleq p_{\mathrm{c}} + n_{\mathrm{f}}$, and $p_{\mathrm{c}} \geq 1$ is the batch window size.

Next, we define the *extended retrospective performance vector* as

$$\hat{Z}(k, \hat{\theta}, \hat{N}) = Z(k) + N(\Phi(k)\hat{\theta} - U(k)) \tag{5.11}$$

where $\hat{Z}(k, \hat{\theta}, \hat{N}) \in \mathbb{R}^{p_{\mathrm{c}}}$, and the matrix $N \in \mathbb{R}^{p_{\mathrm{c}} \times p}$ is a block-Toeplitz representation of the FIR filter written as

$$N = \begin{bmatrix} 0 & \hat{N} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & \hat{N} \end{bmatrix},$$

where $\hat{N} \triangleq [\hat{N}_{n_{\mathrm{f}}} \; \cdots \; \hat{N}_1]$.The target model G_{f} is known to set target locations for the closed-loop poles [8]. Since we define G_{f} as an FIR filter, the closed-loop target pole locations are attracted to the origin. Also, in [8], the ideal G_{f} is chosen to match the

NMP zeros of the plant. This feature ensures that the NMP zeros of the plant are not canceled by G_{c}. In this chapter, we apply optimization techniques to concurrently optimize the controller gains and the target model G_{f}.

5.3.3 Retrospective Cost

Consider the retrospective cost function

$$J(k,\hat{\theta},\hat{N}) = \hat{Z}(k,\hat{\theta},\hat{N})^T R_z \hat{Z}(k,\hat{\theta},\hat{N}) + (\hat{\theta}-\theta(k-1))^T R_\delta (\hat{\theta}-\theta(k-1)), \tag{5.12}$$

where the positive scalar R_z and the positive-definite matrix $R_\delta \in \mathbb{R}^{l_\theta \times l_\theta}$ are the performance and learning rate weightings, respectively.

5.4 Biquadratic Optimization

We use the Matlab fminsearch function for nonlinear function optimization, as well as the alternate convex search (ACS) algorithm [25]. In the latter code, only the variables that are active are optimized while the other variables are kept fixed. Although the ACS algorithm converges to a stationary point of the cost function, it is shown in [25] that global convergence is not guaranteed.

5.4.1 Alternate Convex Search Minimizers

5.4.1.1 Minimizing $\hat{\theta}$ for fixed $\hat{N}$

For fixed $\hat{N}$, the minimizing $\hat{\theta}$ is found by substituting (5.11) into (5.12) and letting $U_{\mathrm{f}}(k) = N \cdot U(k)$ and $\Phi_{\mathrm{f}}(k) = N \cdot \Phi(k)$. The resulting cost function can be

written as

$$J(k,\hat{\theta},\hat{N}) = \hat{\theta}^T A(k)\hat{\theta} + \hat{\theta}^T b(k) + c(k), \tag{5.13}$$

where

$$\begin{aligned} A(k) &= \Phi_{\mathrm{f}}(k)^T R_z \Phi_{\mathrm{f}}(k) + R_\delta, \\ b(k) &= 2\Phi_{\mathrm{f}}(k)^T R_z (Z(k) - U_{\mathrm{f}}(k)) - 2R_\delta \theta(k-1), \\ c(k) &= (Z(k) - U_{\mathrm{f}}(k))^T R_z (Z(k) - U_{\mathrm{f}}(k)) + \\ &\quad \theta(k-1)^T R_\delta \theta(k-1). \end{aligned}$$

Since $A(k)$ is positive definite, $J(k,\hat{\theta},\hat{N})$ has the unique global minimizer

$$\hat{\theta}(k) = -\frac{1}{2} A(k)^{-1} b(k). \tag{5.14}$$

5.4.1.2 Minimizing $\hat{N}$ for fixed θ

For fixed $\theta(k)$, the minimizing $\hat{N}$ is found by substituting (5.11) into (5.12). Letting $\Delta U(k) \triangleq \Phi(k)\theta(k) - U(k)$ and $\Delta\theta(k) \triangleq \theta(k) - \theta(k-1)$, (5.12) can be written as

$$\begin{aligned} J(k,\hat{\theta},\hat{N}) &= \Delta U(k)^{\mathrm{T}} N^{\mathrm{T}} R_z N \Delta U(k) + 2Z(k)^{\mathrm{T}} R_z N \Delta U(k) \\ &\quad + \Delta U(k)^{\mathrm{T}} R_\delta \Delta U(k) + Z(k)^{\mathrm{T}} R_z Z(k). \end{aligned} \tag{5.15}$$

The minimizer is found by setting $\nabla J(k,\hat{\theta},\hat{N}) = 0$ and solving for $\hat{N}$. Defining

$$C_{k_1:k_2} \triangleq \begin{bmatrix} c(k-k_1) \\ \vdots \\ c(k-k_2) \end{bmatrix}, \tag{5.16}$$

the minimizer of (5.15) is

$$\hat{N}(k) = -\xi^{\mathrm{T}}(k)M(k)^{-1}, \tag{5.17}$$

where the positive-definite matrix $M(k) \in \mathbb{R}^{n_{\mathrm{f}} \times n_{\mathrm{f}}}$ has the entries

$$M_{i,j}(k) \triangleq \Delta U^{\mathrm{T}}_{i+1:p_c+i} R_z \Delta U_{j+1:p_c+j}, \tag{5.18}$$

where $1 \leq i, j \leq n_{\mathrm{f}}$, and $\xi \in \mathbb{R}^{n_{\mathrm{f}}}$ has the entries

$$\xi_i \triangleq Z_{\mathrm{f}}^{\mathrm{T}} R_z \Delta U_{i+1:p_c+i} - \Delta\theta(k)^{\mathrm{T}} R_\delta \Delta\theta(k), \tag{5.19}$$

where $1 \leq i \leq n_{\mathrm{f}}$.

5.4.2 The ACS Algorithm

The modified ACS algorithm consists of the following steps:

Step 1 Choose a nonzero starting point $(\hat{N}(k), \hat{\theta}(k)) \in \mathbb{R}^{n_{\mathrm{f}}+l_\theta}$ for $k = p + 1$.

Step 2 For fixed $\hat{N}(k)$, solve for $\hat{\theta}(k)$ using (5.14) and the associated cost using (5.13).

Step 3 For fixed $\hat{\theta}(k)$, solve for $\hat{N}(k)$ using (5.17) and the associated cost using (5.15).

Step 4 Determine whether Step 2 or Step 3 produces the lower cost. If a stopping criteria is satisfied, exit. Otherwise, set $\hat{N}(k+1)$ and $\hat{\theta}(k+1)$ to the corresponding step that produced the lower cost, increment k, and go to Step 2.

For the examples in Section 5.5, the stopping criteria consists of 800 evaluations and a function tolerance of 1×10^{-4}.

5.4.3 fminsearch

The method used with the Matlab fminsearch algorithm consists of the following steps:

Step 1 Choose a starting point $(\hat{N}(k), \hat{\theta}(k)) \in \mathbb{R}^{n_{\mathrm{f}}+l_\theta}$ for $k = p + 1$.

Step 2 At the current time step, minimize the cost function (5.12) until a stopping criteria is satisfied.

Step 3 Set $\hat{N}(k+1) = \hat{N}(k)$ and $\hat{\theta}(k+1) = \hat{\theta}(k)$ for the next starting point, increment k, and go to Step 2.

For the examples in Section 5.5, the stopping criteria consists of 5000 function evaluations and iterations along with a function and variable tolerance of 1×10^{-4}.

5.5 Numerical Examples

In this section, we present numerical examples to illustrate the concurrent optimization technique for the command-following problem shown in Fig. 5.1. The exogeneous signal w is the command r, and $z = y - r$. This problem is a special case of (5.1)-(5.3) with $C = E_1$, $D_2 = 0$, and $E_0 = -1$. Hence $G(\mathbf{z}) = C(\mathbf{z}I - A)^{-1}B$.

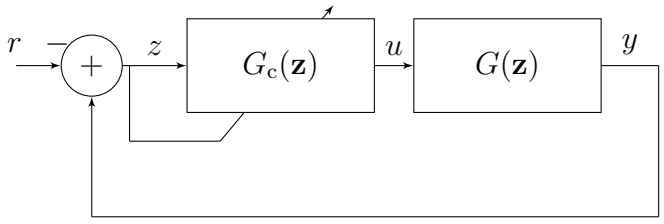


Figure 5.1: The command-following problem.

In each of the examples below, except for Example 5.5.7, all of the filter coefficients are initialized at 0 except for $\hat{N}_1$. Furthermore, $\hat{N}_1$ is initialized based on the parity

of the positive NMP zeros, that is, at $k \leq p+1$

$$\hat{N}_1 = (-1)^h, \tag{5.20}$$

where h is the number of positive NMP zeros in G.

Example 5.1 (Asymptotically stable, minimum-phase plant.)**.** Consider the asymptotically stable, minimum-phase plant

$$G(\mathbf{z}) = \frac{2.0(\mathbf{z}-0.2)}{(\mathbf{z}-0.3)(\mathbf{z}-0.6)}, \tag{5.21}$$

and let r be a unit-height step command. We choose $n_\mathrm{f} = 2$ and note that since G has no positive NMP zeros, $h = 0$ and by (5.20), $\hat{N}_1 = 1$. Thus the filter coefficients in $\hat{N}$ are initialized as [0 1]. Let the controller order $n_\mathrm{c} = 2$, the batch window size $p_\mathrm{c} = 20$, $R_\delta = 1 \times 10^{-3} I_{l_\theta}$, $R_z = 1.0$, and $x(0) = [1\ 1]^\mathrm{T}$. For $k < p+1$, both algorithm controller coefficients are $0_{l_\theta \times 1}$, and at $k = p+1$, the controller coefficients are initialized to $0_{l_\theta \times 1}$ and $1_{l_\theta \times 1}$ for fminsearch and ACS, respectively. Fig. 5.2 shows the results of the concurrent optimization. Both algorithms give comparable results in the converged filter and controller gain coefficients. Note that the filter coefficient $\hat{N}_2$ obtained by both algorithms converge to the value 2.0 of the first nonzero Markov parameter, while the controller converges to an integrator internal model in order to follow the step command. ■

Example 5.2 (Unstable, minimum-phase plant.)**.** Consider the unstable, minimum-phase plant

$$G(\mathbf{z}) = \frac{\mathbf{z}-0.2}{(\mathbf{z}-0.6)(\mathbf{z}-1.3)}, \tag{5.22}$$

and let r be a unit-height step command. We choose $n_\mathrm{f} = 4$ and note that since G

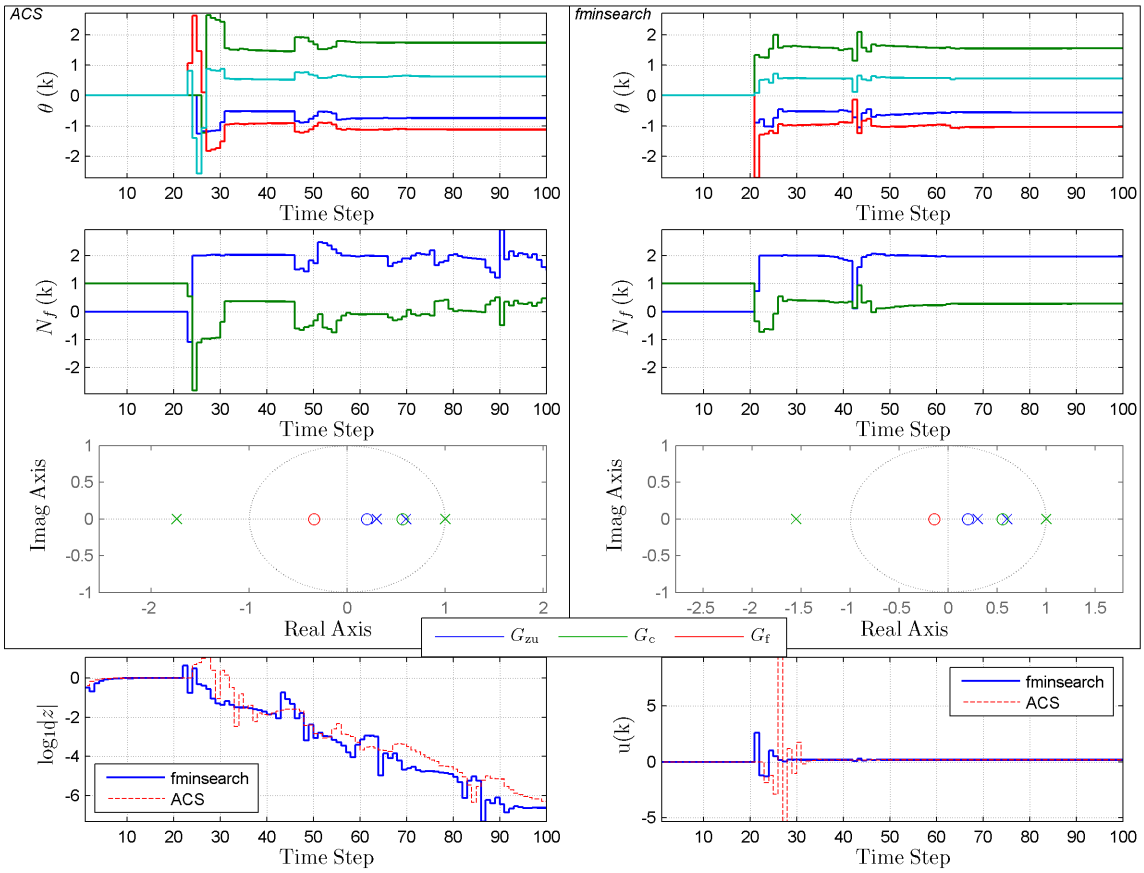


Figure 5.2: Example 5.1*Asymptotically stable, minimum-phase system.* Concurrent optimization is applied to step-command following for the asymptotically stable minimum-phase plant (5.21). The upper three left figures show the result of using the ACS algorithm, and the upper right three plots show the result of using fminsearch. Note that the filter coefficient $\hat{N}_2$ converges to the first nonzero Markov parameter 2.0 of the plant, and the controller converges to an integrator internal model in order to follow the step command

has no positive NMP zeros, $h = 0$ and by (5.20), $\hat{N}_1 = 1$. Thus the filter coefficients in $\hat{N}$ are initialized as [0 0 0 1]. Let the controller order $n_c = 4$, and the batch window size $p_c = 20$, $R_\delta = 1 \times 10^{-1} I_{l_\theta}$, $R_z = 1.0$, and $x(0) = [0\ 0]^{\mathrm{T}}$. For $k < p + 1$, both algorithm controller coefficients are $0_{l_\theta \times 1}$, and at $k = p+1$, the controller coefficients are initialized to $0_{l_\theta \times 1}$ and $1_{l_\theta \times 1}$ for fminsearch and ACS, respectively. Fig. 5.3 shows the results of the concurrent optimization. Note that both algorithms converge to a controller with an integrator internal model in order to follow the step command and that the filter coefficient $\hat{N}_4$ converges to the first nonzero Markov parameter 1.0. ■

Example 5.3 (Asymptotically stable, NMP plant.)**.** Consider the asymptotically

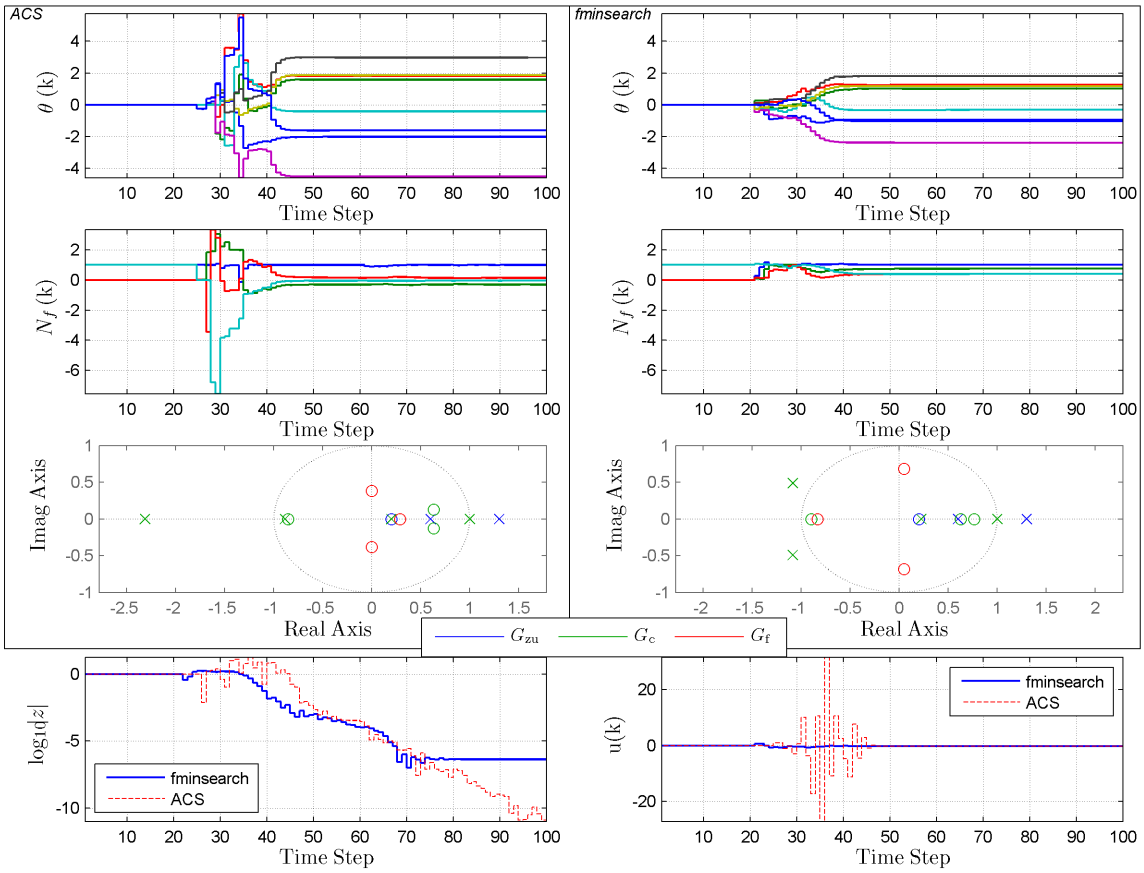


Figure 5.3: Example 5.2: *Unstable, minimum-phase plant.* Concurrent optimization is applied to step-command following for the unstable minimum-phase plant (5.22). Note that the controller stabilizes the unstable plant and develops an integrator internal model in order to follow the step command.

stable, NMP plant

$$G(\mathbf{z}) = \frac{\mathbf{z} - 1.1}{(\mathbf{z} - 0.3)(\mathbf{z} - 0.6)}, \tag{5.23}$$

and let r be a unit-height step command. We choose $n_\mathrm{f} = 4$ and note that since G has one positive NMP zero, $h = 1$ and by (5.20), $\hat{N}_1 = -1$. Thus the filter coefficients in $\hat{N}$ are initialized as $[0\ 0\ 0\ \ -1]$. Let the controller order $n_\mathrm{c} = 4$, and the batch window size $p_\mathrm{c} = 30$, $R_\delta = 1 \times 10^{-3} I_{l_\theta}$, $R_z = 1$, and $x(0) = [0\ 0]^\mathrm{T}$. For $k < p+1$, both algorithm controller coefficients are $0_{l_\theta \times 1}$, and at $k = p+1$, the controller coefficients are initialized to $0_{l_\theta \times 1}$ and $1_{l_\theta \times 1}$ for fminsearch and ACS, respectively. Fig. 5.4 shows the results of the concurrent optimization. Note that the location of the NMP zero

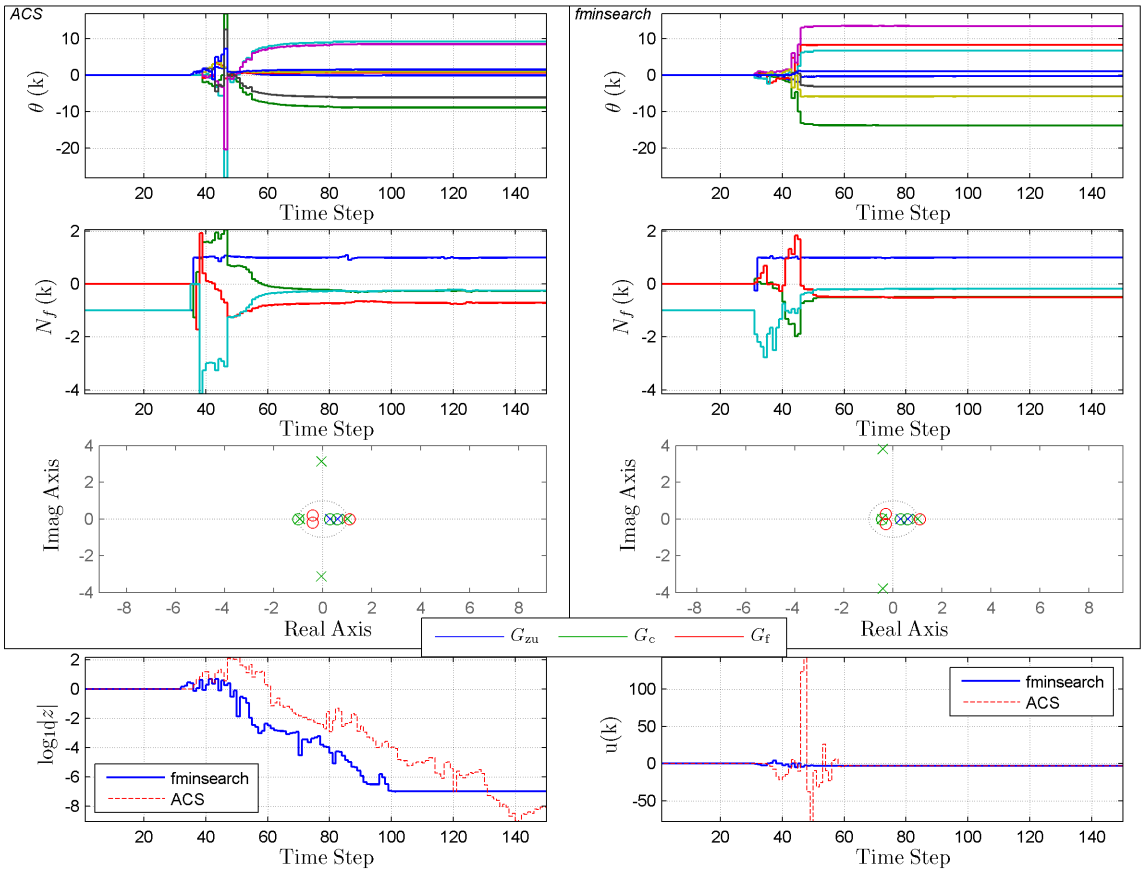


Figure 5.4: Example 5.3:*Asymptotically stable, NMP plant.* Concurrent optimization is applied to step-command following for the asymptotically stable, NMP plant (5.23). Note that the location of the NMP zero is unknown to both algorithms. Also note that the controller develops an integrator internal model in order to follow the step command, and the filter captures the NMP zero location. Figure 5.5 shows a zoomed in view of the plant, controller, and filter pole/zero locations after convergence.

location is unknown to both algorithms. Fig. 5.5 shows the poles and zeros of the plant (5.23), the controller, and the filter. Both algorithms converge to a filter that places a zero at the same location as the plant zero. Also, both algorithms converge to a controller with an integrator internal model in order to follow the step command.

■

Example 5.4 (Asymptotically stable, NMP plant with a NMP negative zero.)**.** Consider the asymptotically stable, NMP plant with a negative NMP zero

$$G(\mathbf{z}) = \frac{\mathbf{z} + 1.1}{(\mathbf{z} - 0.3)(\mathbf{z} - 0.6)}, \tag{5.24}$$

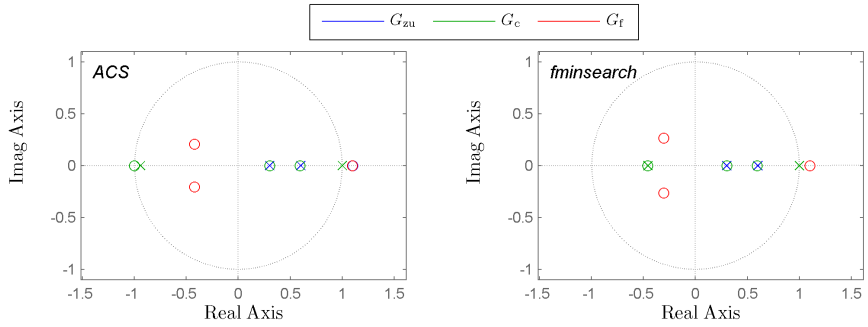


Figure 5.5: Example 5.3: *Asymptotically stable, NMP plant.* This figure shows a zoomed in pole/zero map of the plant, converged controller, and converged filter for the ACS [left] and fminsearch [right] algorithms. Note that the filter captures the NMP zero location of the plant and that the controller converges to an integrator internal model in order to follow the step command.

and let r be a unit-height step command. We choose $n_f = 4$ and note that since G has no positive NMP zeros, $h = 0$ and by (5.20), $\hat{N}_1 = 1$. Thus the filter coefficients in $\hat{N}$ are initialized as [0 0 0 1]. Let the controller order $n_c = 4$, and the batch window size $p_c = 50$, $R_\delta = 1 \times 10^{-1} I_{l_\theta}$, $R_z = 1$, and $x(0) = [1\ 1]^T$. For $k < p + 1$, both algorithm controller coefficients are $0_{l_\theta \times 1}$, and at $k = p+1$, the controller coefficients are initialized to $0_{l_\theta \times 1}$ and $1_{l_\theta \times 1}$ for fminsearch and ACS, respectively. Fig. 5.6 shows the results of the concurrent optimization. Both algorithms give comparable results in the converged filter and controller gain coefficients. Note that the filter coefficient $\hat{N}_4$ for both algorithms converge to the value 1.0 of the first nonzero Markov parameter, while the controller converges to an integrator internal model in order to follow the step command. Note that the location of the NMP zero is unknown to both algorithms and the zero of the converged target model converges to the NMP zero of the plant (5.24). ■

Example 5.5 (Asymptotically stable, NMP plant with relative degree 2.)**.**

Consider the asymptotically stable, NMP plant

$$G(\mathbf{z}) = \frac{\mathbf{z} - 1.2}{(\mathbf{z} - 0.1)(\mathbf{z} - 0.3)(\mathbf{z} - 0.6)}, \tag{5.25}$$

and let r be a unit-height step command. We choose $n_f = 5$ and note that since G

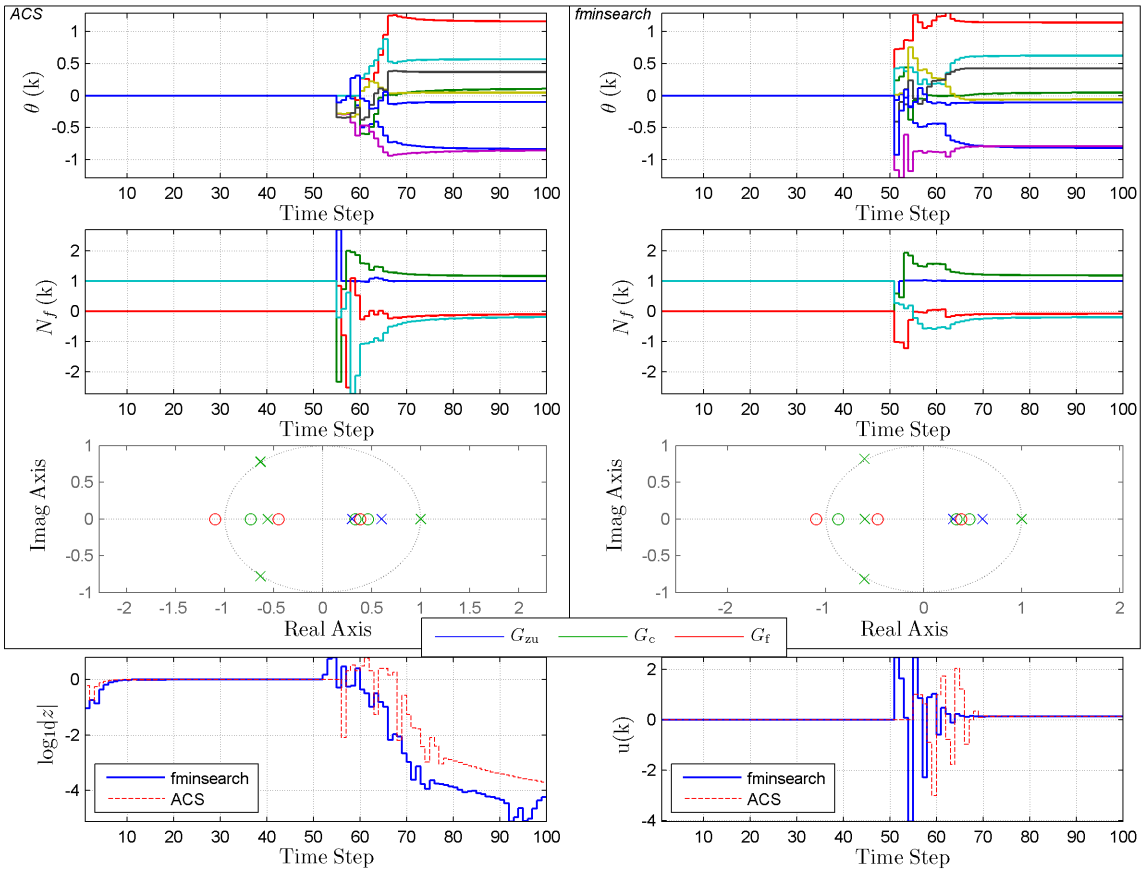


Figure 5.6: Example 5.4: *Asymptotically stable plant with a negative NMP zero.* Concurrent optimization is applied to a step-command following problem for the asymptotically stable plant with a negative NMP zero (5.24). Note that the location of the NMP zero is unknown to both algorithms. The controller converges to an integrator internal model to follow the step command, and the target model captures the location of the negative NMP zero.

has one positive NMP zero, $h = 1$ and by (5.20), $\hat{N}_1 = -1$. Thus the filter coefficients in $\hat{N}$ are initialized as $[0\ 0\ 0\ 0\ \ -1]$. Let the controller order $n_c = 6$, and the batch window size $p_c = 40$, $R_\delta = 1\times10^{-2} I_{l_\theta}$, $R_z = 1$, and $x(0) = [1\ 1\ 1]^{\mathrm{T}}$. For $k < p+1$, both algorithm controller coefficients are $0_{l_\theta \times 1}$, and at $k = p+1$, the controller coefficients are initialized to $0_{l_\theta \times 1}$ and $1_{l_\theta \times 1}$ for fminsearch and ACS, respectively. Fig. 5.7 shows the results of the concurrent optimization. Note that the location of the NMP zero is unknown to both algorithms and the zero of the converged target model converges to to the NMP zero of the plant (5.25). Also note that since the relative degree of the plant (5.25) is 2, the filter coefficient $\hat{N}_5$ for both algorithms converge to 0.0, while the filter coefficient $\hat{N}_4$ converges to the value of the first nonzero Markov parameter

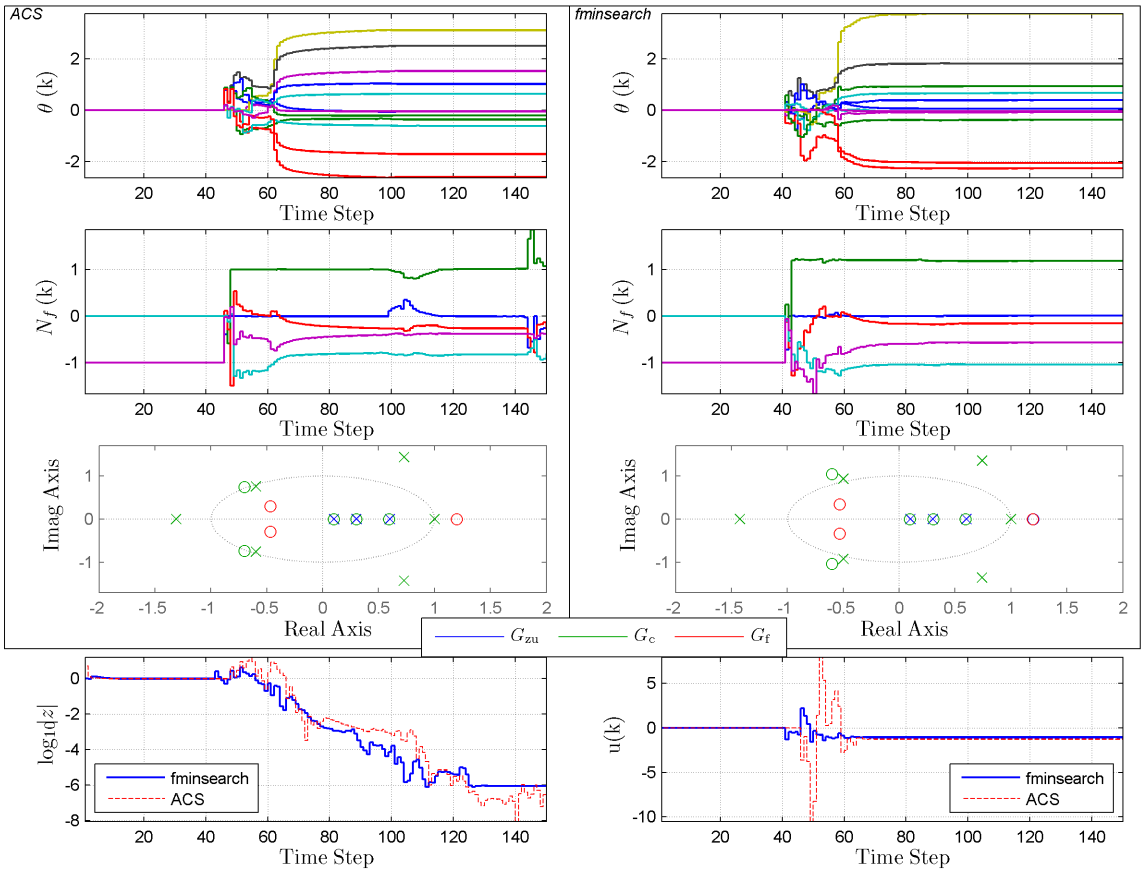


Figure 5.7: Example 5.5: *Asymptotically stable NMP plant of relative degree 2.* Concurrent optimization is applied to a step-command following problem for the asymptotically stable, NMP plant with relative degree 2 (5.25). Note that the location of the NMP zero is unknown to both algorithms. The controller converges to an integrator internal model in order to follow the step command, and the converged target model captures the location of the NMP zero.

1.0. Both algorithms produce a spurious filter zero of high magnitude, which is not included in Figure 5.7. The controller also converges to an integrator internal model in order to follow the step command. ■

Example 5.6 (Asymptotically stable, minimum-phase plant.)**.** Consider the asymptotically stable, minimum-phase plant

$$G(\mathbf{z}) = \frac{\mathbf{z} - 0.3}{(\mathbf{z} - 0.2)(\mathbf{z} - 0.6)}, \tag{5.26}$$

and let r be a unit-amplitude harmonic command with frequency $\omega = \frac{2\pi}{25}$ rad/sample. We choose $n_{\mathrm{f}} = 3$ and note that since G has no positive NMP zeros, $h = 0$ and by

(5.20), $\hat{N}_1 = 1$. Thus the filter coefficients in $\hat{N}$ are initialized as [0 0 1]. Let the controller order $n_c = 4$, and the batch window size $p_c = 20$, $R_\delta = 1 \times 10^{-3} I_{l_\theta}$, $R_z = 1$, and $x(0) = [0\ 0]^T$. For $k < p+1$, both algorithm controller coefficients are $0_{l_\theta \times 1}$, and at $k = p+1$, the controller coefficients are initialized to $0_{l_\theta \times 1}$ and $1_{l_\theta \times 1}$ for fminsearch and ACS, respectively. Fig. 5.8 shows the results of the concurrent optimization. Note that the controller converges to a harmonic internal model in order to follow

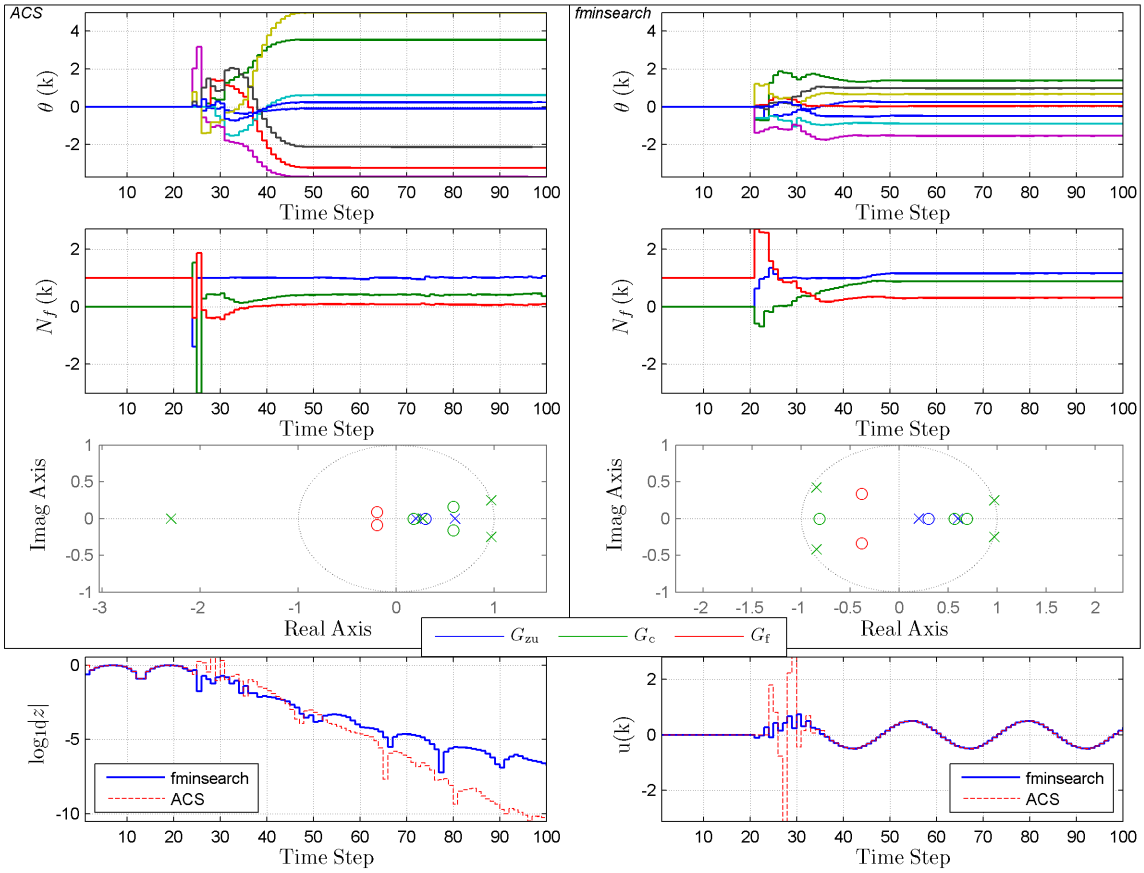


Figure 5.8: Example 5.6: *Asymptotically stable, minimum-phase plant with harmonic-command following.* Concurrent optimization is applied to a harmonic-command following problem for the asymptotically stable, minimum-phase plant (5.26). Note that both algorithms converge to an internal model controller with poles at the command frequency on the unit circle and the filter coefficient $\hat{N}_3$ converges to the first nonzero Markov parameter 1.0.

the same frequency of the command r and the filter coefficient $\hat{N}_3$ converges to the first nonzero Markov parameter 1.0. ■

Example 5.7 (Asymptotically stable, NMP plant with relative degree 2.)**.** Consider the asymptotically stable, minimum-phase plant given in (5.25), and let r be a unit-

height step command. We choose $n_f = 5$ and note that since G has one NMP zero, $h = 1$ and by (5.20), $\hat{N}_1 = -1$. In this example, we violate (5.20) by choosing the opposite sign of $\hat{N}_1$ to let $\hat{N}_1 = 1$. Thus the filter coefficients in $\hat{N}$ are initialized as [0 0 0 0 1]. Let the initialization parameters be the same as stated in Example 5.5. Fig. 5.9 shows the results of the concurrent optimization. Note that the location of

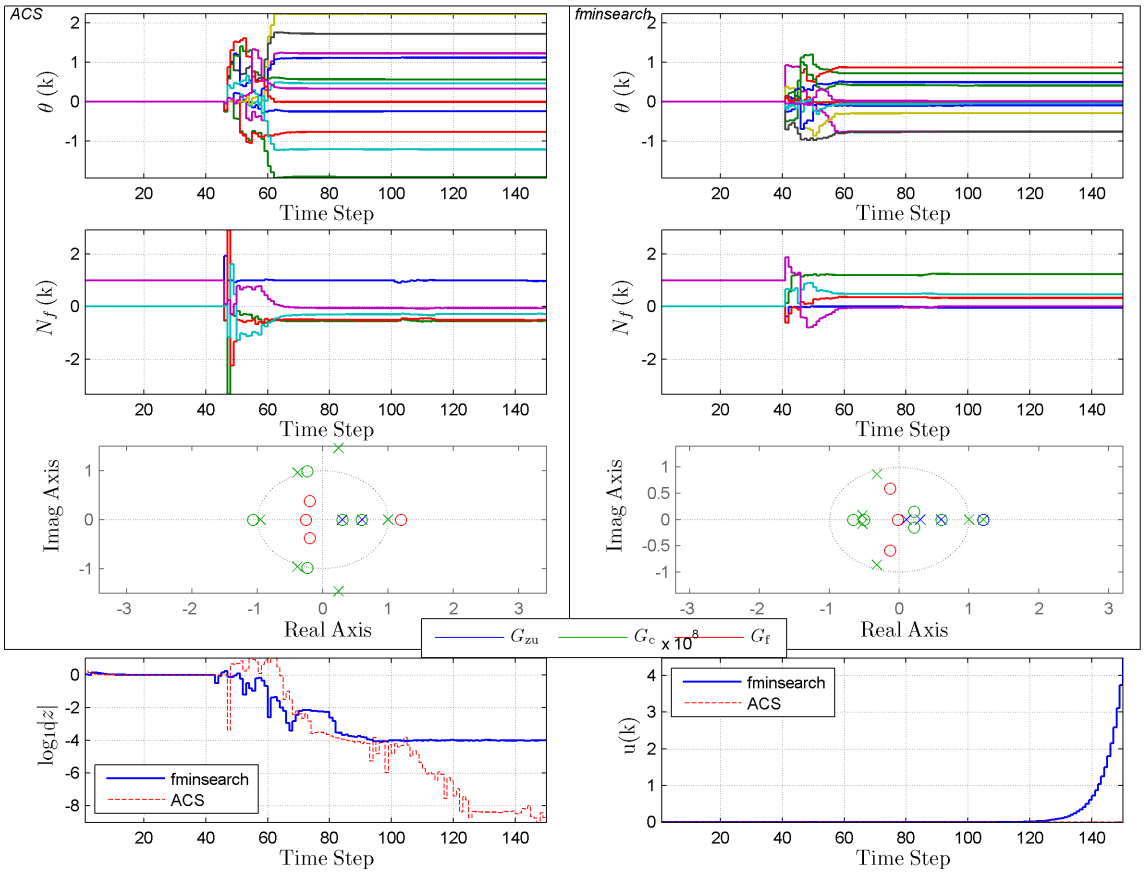


Figure 5.9: Example 5.7: *Asymptotically stable NMP plant of relative degree 2.* Concurrent optimization is applied to a step-command following problem for the asymptotically stable, NMP plant with relative degree 2 (5.25). Note that the location of the NMP zero is unknown to both algorithms. For both algorithms, the controller converges to an integrator internal model in order to follow the step command. Also note that for the ACS algorithm, the asymptotic target model captures the location of the NMP zero while the fminsearch algorithm does not capture the NMP zero leading to an unstable controller pole and plant NMP zero cancellation.

the NMP zero is unknown to both algorithms and that for the ACS algorithm the asymptotic target model captures the location of the NMP zero while the fminsearch algorithm does not capture the NMP zero leading to an unstable controller pole and plant NMP zero cancellation. ■

5.6 Conclusions

The cost function associated with the retrospective cost adaptive control algorithm (RCAC) was used for concurrent optimization of the controller gains and the target model due to its natural biquadratic structure. A modified version of the alternating convex search (ACS) and the Matlab fminsearch algorithm demonstrated the ability to concurrently optimize these coefficients for a command-following problem. The novel element in this chapter is the ability to minimize the cost function without prior knowledge of the plant transfer function, including NMP zero locations. Future work will focus on improving the computational efficiency of the concurrent optimization along with guarantees of global convergence.

CHAPTER 6

Direct and Indirect Closed-Loop Architectures for Estimating Nonminimum-Phase Zeros

6.1 Introduction

During closed-loop operation, it is often necessary to monitor the plant in order to detect changes that can degrade stability and performance. At the same time, more accurate modeling of the plant can facilitate the ability to enhance closed-loop performance through on-line controller modification [4, 55–57]. Accordingly, closed-loop identification has been extensively studied [30, 58–64] and successful applications utilizing identification during closed-loop operation are discussed in [65–70].

The starting point for the present chapter is the survey paper [30], which analyzes various architectures for closed-loop identification. That work emphasizes the practical importance of the problem and demonstrates the richness of the subject in terms of the diverse architectures that can be employed. The presentation in [30], however, does not include a numerical investigation of the relative merits of candidate architectures and identification algorithms. Consequently, as a complement to [30], the contribution of this chapter is a detailed numerical study that compares multiple closed-loop identification architectures in terms of their accuracy in estimating nonminimum-phase (NMP) zeros.

In contrast to open-loop identification, closed-loop identification presents unique challenges in system identification [71]. For example, the model input may lack sufficient persistency due to the limited spectral content of the command and disturbances. In addition, because of closed-loop operation, the model input may be correlated with the exogenous noise, potentially leading to parameter bias.

As noted above, closed-loop identification can be performed with a variety of architectures. The basic architecture for closed-loop identification is *direct closed-loop identification*, where the control signal and measurement, that is, the plant input and output, are used to construct a plant model. Since the control signal arises in response to disturbance and sensor noise, the control signal is correlated with these noise signals, potentially leading to bias in the parameter estimates.

A variation of direct closed-loop identification is to add an auxiliary signal to the control signal and then use the sum of these signals as the model input. This approach, called *auxiliary direct closed-loop identification*, uses the auxiliary signal to enhance identification accuracy at the cost of disrupting closed-loop performance. The challenge of designing minimally disruptive signals for identification is discussed in [72].

An alternative approach to direct closed-loop identification is to estimate a closed-loop transfer function rather than the plant itself. This approach requires an external signal. The simplest approach of this type, called *indirect closed-loop identification*, uses the command signal as the model input. To facilitate the method, the command signal can be chosen to be sufficiently persistent albeit at the cost of degrading closed-loop operation. A model of the plant is subsequently extracted from the estimated closed-loop model.

A variation of indirect closed-loop identification is to add an auxiliary signal to the control signal and then use only the auxiliary signal as the model input. This approach, which is known as *auxiliary indirect closed-loop identification* identifies the

overall closed-loop system. As in the case of indirect closed-loop identification, a model of the plant is subsequently extracted from the estimated closed-loop model. In order to simplify the identification, the auxiliary signal is chosen in this chapter to be a multiple of the command signal.

In addition to direct closed-loop, auxiliary direct closed-loop, indirect closed-loop, and auxiliary indirect closed-loop identification, we consider two nonstandard ways in which the external signal used for auxiliary direct closed-loop identification and auxiliary indirect closed-loop identification can be injected into the loop. In the standard approach, the auxiliary signal is added to the control signal. In contrast, in *intercalated injection*, the auxiliary signal is added to an internal signal in the feedback controller. This technique arises naturally in retrospective cost adaptive control [35, 36], where the effect of the controller update at each step is equivalent to intercalated injection of a control-input perturbation. For closed-loop identification, the *intercalated auxiliary indirect closed-loop* and *auxiliary indirect closed-loop* architectures are characterized by the fact that the resulting transfer functions have restricted numerators.

This chapter thus considers six architectures for identifying a plant operating in closed loop, namely, direct closed-loop (DCL) identification, standard auxiliary direct closed-loop (ADCL/S) identification, intercalated auxiliary direct closed-loop (ADCL/I) identification, indirect closed-loop (ICL) identification, standard auxiliary indirect closed-loop (AICL/S) identification, intercalated auxiliary indirect closed-loop (AICL/I) identification.

The goal of this chapter is to assess the advantages and disadvantages of these six architectures in estimating the nonminimum-phase (NMP) zeros of a plant operating in closed loop. NMP zeros are one of the most challenging aspects of feedback control in terms of limiting achievable performance [31]. Consequently, after constructing a model of the open- or closed-loop plant, the metric used to assess the estimation

accuracy is defined to be the accuracy of the estimate of the NMP zero. This objective is motivated by retrospective cost adaptive control [32–36], which requires knowledge of NMP plant zeros.

In order to estimate NMP zeros, we consider infinite impulse response (IIR) models. In order to focus attention on a comparison of architectures, we make the simplifying assumption that the plant order is known. In the case where the plant order is unknown, additional techniques for order estimation are needed. This can be done, for example, by overestimating the plant order, impulsing the estimated model, and then applying Ho-Kalman realization theory [73, 74] along with heuristic nuclear norm minimization techniques [75, 76] to estimate the model order and construct a reduced-order model. Alternatively, this can be done by using a finite impulse response (FIR) model structure to directly estimate the Markov parameters for use in the Hankel matrix. This is the approach used in [77], where noncausal FIR models are used to account for the possibility that the plant may be open-loop unstable. The approach of [77], however, is not applicable to plants with poles on the unit circle. Consequently, we confine our attention to IIR models under the assumption that the plant order is known.

We consider two techniques for identification, namely, least squares and prediction error methods. Least squares (LS) provides a baseline technique for estimation. In most cases, least-squares estimates are biased, and, assuming that the model input is white, we provide an analytical expression for the bias. Setting the bias to zero yields a necessary and sufficient condition under which the model estimate is consistent. This result extends the consistency analysis provided in [78, 79].

A more effective approach to providing consistent estimates within the context of closed-loop identification is prediction error methods (PEM). PEM is described in [80] and applied to direct and indirect closed-loop identification architectures in [30, 60, 81, 82]. PEM can be viewed as an extension of LS that fits both the plant

and noise model parameters by propagating the outputs over a window of data. The minimization of the prediction error may be quadratic, as in LS, or nonquadratic, as in maximum-likelihood minimization [83, 84].

An alternative technique for closed-loop identification is instrumental variables (IV). An early version of IV was introduced in [85] for identification within the context of discrete-time adaptive control [86] and is extensively studied in [87–89]. As discussed in [90], the effectiveness of IV for closed-loop identification is sensitive to the choice of instruments. In addition, as noted in [90], an optimal IV estimator can be obtained only if the noise model is known. To simplify the comparison of architectures for closed-loop identification, we thus focus on least squares and PEM.

The noise models are confined to stationary white and colored noise random processes representing either process noise or sensor noise. We do not, however, consider noise that corrupts the signals used for model estimation. Noise of this type gives rise to an *errors-in-variables* (EIV) identification problem. EIV is extensively studied; see, for example, [91–94].

The contents of the chapter are as follows. Section 3 describes the estimation algorithms, including least squares and prediction error methods. Sections 4 and 5 describe the direct and indirect closed-loop identification architectures, respectively, with and without an auxiliary signal, including standard intercalated injection. Section 6 presents a numerical comparison of the direct closed-loop identification architectures. Two examples are considered. In the first example, the plant is open-loop stable; in the second example, the plant is open-loop unstable. Section 7 is analogous to Section 6 for the case of indirect architectures. Conclusions and directions for future research are given in Section 8.

Notation: I_n denotes the $n \times n$ identity matrix, $0_{n\times m}$ denotes the $n \times m$ zero matrix, and $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm.

6.2 Model Structure

Consider the discrete-time n_0^{th}-order SISO transfer function G_0 defined as

$$G_0(\mathbf{q}) \triangleq \frac{N_0(\mathbf{q})}{D_0(\mathbf{q})}, \tag{6.1}$$

where $\mathbf{q}$ is the forward-shift operator and

$$N_0(\mathbf{q}) = \sum_{i=0}^{n_0} b_i \mathbf{q}^{n_0-i}, \qquad D_0(\mathbf{q}) = \mathbf{q}^{n_0} + \sum_{i=1}^{n_0} a_i \mathbf{q}^{n_0-i}. \tag{6.2}$$

Letting u_0 and y_0 denote the input and output, respectively, of G_0, that is,

$$y_0(k) = G_0(\mathbf{q})u_0(k), \tag{6.3}$$

it follows that, for all $k \geq 0$,

$$y_0(k) = -a_1 y_0(k-1) - \cdots - a_{n_0} y_0(k-n_0) + b_0 u_0(k) + \cdots + b_{n_0} u_0(k-n_0), \tag{6.4}$$

which can be written as

$$y_0(k) = \phi_{y_0,u_0}(k)\theta, \tag{6.5}$$

where

$$\phi_{y_0,u_0}(k) \triangleq [-y_0(k-1) \quad \cdots \quad -y_0(k-n_0) \quad u_0(k) \quad \cdots \quad u_0(k-n_0)], \tag{6.6}$$

$$\theta \triangleq [a_1 \quad \cdots \quad a_{n_0} \quad b_0 \quad \cdots \quad b_{n_0}]^{\mathrm{T}}. \tag{6.7}$$

The measured output y of G_0 is defined as

$$y(k) \stackrel{\triangle}{=} y_0(k) + w_0(k) = G_0(\mathbf{q})u_0(k) + w_0(k), \tag{6.8}$$

where w_0 is a noise signal. Since w_0 is added to y_0 in (6.8), it appears to represent sensor noise. However, w_0 can represent either sensor or process noise or both by viewing w_0 as the output of a system G_w, that is,

$$w_0(k) = G_w(\mathbf{q})w(k) = \frac{N_w(\mathbf{q})}{D_w(\mathbf{q})}w(k), \tag{6.9}$$

where G_w is a proper transfer function of order $\bar{n}$ and w is zero-mean white noise with standard deviation σ_w. It follows that, for all $k \geq 0$,

$$w_0(k) = -\bar{a}_1 w_0(k-1) - \cdots - \bar{a}_{\bar{n}} w_0(k-\bar{n}) + \bar{b}_0 w(k) + \cdots + \bar{b}_{\bar{n}} w(k-\bar{n}), \tag{6.10}$$

which can be written as

$$w_0(k) = \phi_{w_0,w}(k)\theta_w, \tag{6.11}$$

where

$$\phi_{w_0,w}(k) \stackrel{\triangle}{=} [-w_0(k-1) \quad \cdots \quad -w_0(k-\bar{n}) \quad w(k) \quad \cdots \quad w(k-\bar{n})], \tag{6.12}$$

$$\theta_w \stackrel{\triangle}{=} [\bar{a}_1 \quad \cdots \quad \bar{a}_{\bar{n}} \quad \bar{b}_0 \quad \cdots \quad \bar{b}_{\bar{n}}]^{\mathrm{T}}. \tag{6.13}$$

We stress that the dynamics of (6.9) may or may not be the same as those of (6.3), and thus w_0 may or may not contribute to y in the same way that u_0 does. For example, suppose that G_0 represents the open-loop plant G. To represent process noise that drives the plant in the same way that the control does, G_w can be chosen to be equal to G_0 and the initial conditions of G_w and G can be set to provide the

correct free response. The initial conditions of G_0 and G_w are accounted for by the use of the forward shift operator "$\mathbf{q}$" rather than the Laplace "s." Consequently, by suitable choice of G_w and its initial conditions, the signal w_0 can represent either process noise or sensor noise.

6.3 Identification Algorithms

In this section we briefly review least squares (LS) and prediction error methods (PEM) for estimating the coefficients of G_0. For simplicity we assume throughout this chapter that the order n_0 of G_0 is known. For the direct closed-loop identification algorithms, n_0 represents the plant order, whereas, for the indirect closed-loop identification algorithms, n_0 represents the order of the closed-loop transfer function. We also assume that all random processes are ergodic so that ensemble averages are equal, with probability 1, to time averages of realizations.

6.3.1 Least Squares

In order to estimate G_0, we use the model structure

$$G_m(\mathbf{q}) \stackrel{\triangle}{=} \frac{N_m(\mathbf{q})}{D_m(\mathbf{q})}, \tag{6.14}$$

where $N_m(\mathbf{q}) \stackrel{\triangle}{=} \sum_{i=0}^{n_0} \hat{b}_i \mathbf{q}^{n_0-i}$ and $D_m(\mathbf{q}) \stackrel{\triangle}{=} \mathbf{q}^{n_0} + \sum_{i=1}^{n_0} \hat{a}_i \mathbf{q}^{n_0-i}$. Define

$$\phi_y(k) \stackrel{\triangle}{=} [-y(k-1) \quad \cdots \quad -y(k-n_0)], \quad \phi_{u_0}(k) \stackrel{\triangle}{=} [u_0(k) \quad \cdots \quad u_0(k-n_0)], \tag{6.15}$$

and $\tilde{n} \triangleq \max(n_0, \bar{n})$. Let $\ell \geq \tilde{n}$ be the number of samples of y and u_0, and define

$$\Phi_{y,\ell} \triangleq \begin{bmatrix} \phi_y(\tilde{n}) \\ \vdots \\ \phi_y(\ell) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times n_0}, \quad \Phi_{u_0,\ell} \triangleq \begin{bmatrix} \phi_{u_0}(\tilde{n}) \\ \vdots \\ \phi_{u_0}(\ell) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times (n_0+1)}, \tag{6.16}$$

$$\Phi_{y,u_0,\ell} \triangleq \begin{bmatrix} \Phi_{y,\ell} & \Phi_{u_0,\ell} \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times (2n_0+1)}, \quad \Psi_{y,\ell} \triangleq \begin{bmatrix} y(\tilde{n}) \\ \vdots \\ y(\ell) \end{bmatrix} \in \mathbb{R}^{\ell-\tilde{n}+1}. \tag{6.17}$$

Then, the least squares estimate $\hat{\theta}_\ell^{\mathrm{LS}}$ of θ, which is defined by

$$\hat{\theta}_\ell^{\mathrm{LS}} \triangleq \underset{\bar{\theta}\in\mathbb{R}^{2n_0+1}}{\arg\min} \left\| \Phi_{y,u_0,\ell}\bar{\theta} - \Psi_{y,\ell} \right\|_{\mathrm{F}}, \tag{6.18}$$

is given by

$$\hat{\theta}_\ell^{\mathrm{LS}} \triangleq (\Phi_{y,u_0,\ell}^{\mathrm{T}}\Phi_{y,u_0,\ell})^{+}\Phi_{y,u_0,\ell}^{\mathrm{T}}\Psi_{y,\ell}, \tag{6.19}$$

where "+" denotes the pseudoinverse.

Next, define the $(2n_0+1)\times(2n_0+1)$ positive-semidefinite covariance matrices

$$\Gamma \triangleq \mathbb{E}\begin{bmatrix} \mathcal{Y}^2(k) & \cdots & \mathcal{Y}(k)\mathcal{Y}(k+n_0-1) & -\mathcal{Y}(k)\mathcal{U}_0(k+1) & \cdots & -\mathcal{Y}(k)\mathcal{U}_0(k-n_0+1) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{Y}(k)\mathcal{Y}(k+n_0-1) & \cdots & \mathcal{Y}^2(k) & -\mathcal{Y}(k)\mathcal{U}_0(k+n_0) & \cdots & -\mathcal{Y}(k)\mathcal{U}_0(k) \\ -\mathcal{Y}(k)\mathcal{U}_0(k+1) & \cdots & -\mathcal{Y}(k)\mathcal{U}_0(k+n_0) & \mathcal{U}_0^2(k) & \cdots & \mathcal{U}_0(k)\mathcal{U}_0(k+n_0) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ -\mathcal{Y}(k)\mathcal{U}_0(k-n_0+1) & \cdots & -\mathcal{Y}(k)\mathcal{U}_0(k) & \mathcal{U}_0(k)\mathcal{U}_0(k+n_0) & \cdots & \mathcal{U}_0^2(k) \end{bmatrix}, \tag{6.20}$$

$$\Gamma_0 \stackrel{\triangle}{=}$$

$$\mathbb{E}\begin{bmatrix} \mathcal{Y}_0^2(k) & \cdots & \mathcal{Y}_0(k)\mathcal{Y}_0(k+n_0-1) & -\mathcal{Y}_0(k)\mathcal{U}_0(k+1) & \cdots & -\mathcal{Y}_0(k)\mathcal{U}_0(k-n_0+1) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{Y}_0(k)\mathcal{Y}_0(k+n_0-1) & \cdots & \mathcal{Y}_0^2(k) & -\mathcal{Y}_0(k)\mathcal{U}_0(k+n_0) & \cdots & -\mathcal{Y}_0(k)\mathcal{U}_0(k) \\ -\mathcal{Y}_0(k)\mathcal{U}_0(k+1) & \cdots & -\mathcal{Y}_0(k)\mathcal{U}_0(k+n_0) & \mathcal{U}_0^2(k) & \cdots & \mathcal{U}_0(k)\mathcal{U}_0(k+n_0) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ -\mathcal{Y}_0(k)\mathcal{U}_0(k-n_0+1) & \cdots & -\mathcal{Y}_0(k)\mathcal{U}_0(k) & \mathcal{U}_0(k)\mathcal{U}_0(k+n_0) & \cdots & \mathcal{U}_0^2(k) \end{bmatrix}. \tag{6.21}$$

It thus follows that

$$\Gamma \stackrel{\text{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y,u_0,\ell}^{\mathrm{T}}\Phi_{y,u_0,\ell}, \quad \Gamma_0 \stackrel{\text{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y_0,u_0,\ell}^{\mathrm{T}}\Phi_{y_0,u_0,\ell}. \tag{6.22}$$

Define the $n_0 \times \bar{n}$ matrix $\mathcal{H}_{G_w}$, where, if $\bar{n} < n_0$, then

$$\mathcal{H}_{G_w} \stackrel{\triangle}{=} \begin{bmatrix} \sum_{i=0}^{\infty} H_{w,i}^2 & \cdots & \sum_{i=0}^{\infty} H_{w,i}H_{w,\bar{n}-1+i} \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{\infty} H_{w,i}H_{w,\bar{n}-1+i} & \cdots & \sum_{i=0}^{\infty} H_{w,i}^2 \\ \vdots & \cdots & \vdots \\ \sum_{i=0}^{\infty} H_{w,i}H_{w,n_0-1+i} & \cdots & \sum_{i=0}^{\infty} H_{w,i}H_{w,n_0-\bar{n}+i} \end{bmatrix}, \tag{6.23}$$

if $\bar{n} = n_0$, then

$$\mathcal{H}_{G_w} \stackrel{\triangle}{=} \begin{bmatrix} \sum_{i=0}^{\infty} H_{w,i}^2 & \cdots & \sum_{i=0}^{\infty} H_{w,i}H_{w,n_0-1+i} \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{\infty} H_{w,i}H_{w,n_0-1+i} & \cdots & \sum_{i=0}^{\infty} H_{w,i}^2 \end{bmatrix}, \tag{6.24}$$

and, if $\bar{n} > n_0$, then

$$\mathcal{H}_{G_w} \triangleq \begin{bmatrix} \sum_{i=0}^{\infty} H_{w,i}^2 & \cdots & \sum_{i=0}^{\infty} H_{w,i} H_{w,n_0-1+i} & \cdots & \sum_{i=0}^{\infty} H_{w,i} H_{w,\bar{n}-1+i} \\ \vdots & \ddots & \vdots & \cdots & \vdots \\ \sum_{i=0}^{\infty} H_{w,i} H_{w,n_0-1+i} & \cdots & \sum_{i=0}^{\infty} H_{w,i}^2 & \cdots & \sum_{i=0}^{\infty} H_{w,i} H_{w,\bar{n}-n_0+i} \end{bmatrix}, \tag{6.25}$$

where $H_{w,i}$ is the i^{th} Markov parameter of G_w. Furthermore, define $\tilde{\mathcal{H}}_{G_w} \in \mathbb{R}^{n_0 \times n_0}$ to be the right-hand side of (6.24). Note that $\tilde{\mathcal{H}}_{G_w}$ has size $n_0 \times n_0$ irrespective of $\bar{n}$.

Lemma 2. Let u_0 and w be realizations of zero-mean, uncorrelated stationary white random processes $\mathcal{U}_0$ and $\mathcal{W}$, respectively, with finite second moments, and let w_0 be given by (6.9). Then,

$$\Gamma = \Gamma_0 + \begin{bmatrix} \sigma_w^2 \tilde{\mathcal{H}}_{G_w} & 0_{n_0 \times (n_0+1)} \\ 0_{(n_0+1) \times n_0} & 0_{(n_0+1) \times (n_0+1)} \end{bmatrix}. \tag{6.26}$$

Proof.

First, note that

$$\Phi_{y,\ell} = \Phi_{y_0,\ell} + \tilde{\Phi}_{w_0,\ell}, \tag{6.27}$$

where

$$\Phi_{y_0,\ell} = \begin{bmatrix} -y_0(\tilde{n}-1) & \cdots & -y_0(\tilde{n}-n_0) \\ \vdots & \cdots & \vdots \\ -y_0(\ell-1) & \cdots & -y_0(\ell-n_0) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times n_0}, \tag{6.28}$$

$$\tilde{\Phi}_{w_0,\ell} = \begin{bmatrix} -w_0(\tilde{n}-1) & \cdots & -w_0(\tilde{n}-n_0) \\ \vdots & \cdots & \vdots \\ -w_0(\ell-1) & \cdots & -w_0(\ell-n_0) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times n_0}. \tag{6.29}$$

Next, note that

$$\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{y,\ell} \overset{\text{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell}, \quad \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y,\ell} \overset{\text{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell}, \tag{6.30}$$

$$\lim_{\ell\to\infty} \frac{1}{\ell}\tilde{\Phi}_{w_0,\ell}^{\mathrm{T}}\Phi_{y,\ell} \overset{\text{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\tilde{\Phi}_{w_0,\ell}^{\mathrm{T}}\tilde{\Phi}_{w_0,\ell} \overset{\text{wp1}}{=} \sigma_w^2\tilde{\mathcal{H}}_{G_w}. \tag{6.31}$$

Using (6.22), (6.27), (6.30), and (6.31) yields

$$\begin{aligned}
\Gamma &\overset{\text{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell} \begin{bmatrix} \Phi_{y,\ell}^{\mathrm{T}}\Phi_{y,\ell} & \Phi_{y,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \\ \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y,\ell} & \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \end{bmatrix} = \lim_{\ell\to\infty} \frac{1}{\ell} \begin{bmatrix} \Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} + \tilde{\Phi}_{w_0,\ell}^{\mathrm{T}}\tilde{\Phi}_{w_0,\ell} & \Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \\ \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \end{bmatrix}, \\
&= \lim_{\ell\to\infty} \frac{1}{\ell} \begin{bmatrix} \Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \\ \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \end{bmatrix} + \lim_{\ell\to\infty} \frac{1}{\ell} \begin{bmatrix} \tilde{\Phi}_{w_0,\ell}^{\mathrm{T}}\tilde{\Phi}_{w_0,\ell} & 0_{n_0\times(n_0+1)} \\ 0_{(n_0+1)\times n_0} & 0_{(n_0+1)\times(n_0+1)} \end{bmatrix}, \\
&= \Gamma_0 + \begin{bmatrix} \sigma_w^2\tilde{\mathcal{H}}_{G_w} & 0_{n_0\times(n_0+1)} \\ 0_{(n_0+1)\times n_0} & 0_{(n_0+1)\times(n_0+1)} \end{bmatrix}.
\end{aligned}$$

□

Note that, if $\ell > 3\tilde{n}$, then $\Phi_{y,u_0,\ell}$ is a tall matrix. The following result is a consequence of the fact that $\ell \mapsto \sigma_{\min}(\Phi_{y,u_0,\ell}^{\mathrm{T}}\Phi_{y,u_0,\ell})$ is nondecreasing.

Lemma 3. Let $\tilde{n} \geq 1$. Then, Γ is positive definite if and only if, with probabil-

ity 1, there exists $\ell \geq 3\tilde{n}$ such that $\operatorname{rank} \Phi_{y,u_0,\ell} = 2n_0 + 1$.

Next, define the $n_0 \times (\bar{n}+1)$ matrix $\mathcal{H}_w$, where, if $\bar{n} < n_0$, then

$$\mathcal{H}_w \stackrel{\triangle}{=} \begin{bmatrix} 0 & H_{w,0} & \cdots & H_{w,\bar{n}-1} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & H_{w,0} \\ 0_{(n_0-\bar{n})\times 1} & \cdots & \cdots & 0_{(n_0-\bar{n})\times 1} \end{bmatrix} \in \mathbb{R}^{n_0 \times (\bar{n}+1)}, \tag{6.32}$$

if $\bar{n} = n_0$, then

$$\mathcal{H}_w \stackrel{\triangle}{=} \begin{bmatrix} 0 & H_{w,0} & \cdots & H_{w,n_0-1} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & H_{w,0} \end{bmatrix} \in \mathbb{R}^{n_0 \times (n_0+1)}, \tag{6.33}$$

and, if $\bar{n} > n_0$, then

$$\mathcal{H}_w \stackrel{\triangle}{=} \begin{bmatrix} 0 & H_{w,0} & \cdots & H_{w,\bar{n}-n_0} & \cdots & H_{w,\bar{n}-1} \\ \vdots & \ddots & \ddots & \vdots & \ddots & \ddots \\ 0 & \cdots & 0 & H_{w,0} & \cdots & H_{w,\bar{n}-n_0} \end{bmatrix} \in \mathbb{R}^{n_0 \times (\bar{n}+1)}. \tag{6.34}$$

Also, define

$$N_w \stackrel{\triangle}{=} \begin{bmatrix} \bar{b}_0 \\ \vdots \\ \bar{b}_{\bar{n}} \end{bmatrix}, \quad D_w \stackrel{\triangle}{=} \begin{bmatrix} \bar{a}_1 \\ \vdots \\ \bar{a}_{\bar{n}} \end{bmatrix}, \quad \theta_w \stackrel{\triangle}{=} \begin{bmatrix} D_w \\ N_w \end{bmatrix} \in \mathbb{R}^{2\bar{n}+1}, \tag{6.35}$$

where $N_w(\mathbf{q}) = \sum_{i=0}^{\bar{n}} \bar{b}_i \mathbf{q}^{\bar{n}-i}$, and $D_w(\mathbf{q}) = \mathbf{q}^{\bar{n}} + \sum_{i=1}^{\bar{n}} \bar{a}_i \mathbf{q}^{\bar{n}-i}$.

Theorem 6.1. Let u_0 and w be realizations of zero-mean, uncorrelated stationary white random processes $\mathcal{U}_0$ and $\mathcal{W}$, respectively, with finite second moments, and let

w_0 be given by (6.9). Then,

$$\Gamma \lim_{\ell\to\infty} \hat{\theta}_\ell^{\mathrm{LS}} \overset{\mathrm{wp1}}{=} \Gamma_0\theta + \begin{bmatrix} \sigma_w^2 \mathcal{H}_{G_w} D_w - \sigma_w^2 \mathcal{H}_w N_w \\ 0_{(n_0+1)\times 1} \end{bmatrix}. \tag{6.36}$$

If, in addition, Γ is positive definite, then the following statements are equivalent:

i) $\tilde{\mathcal{H}}_{G_w} D_0 = \mathcal{H}_{G_w} D_w - \mathcal{H}_w N_w$.

ii) $\lim_{\ell\to\infty} \hat{\theta}_\ell^{\mathrm{LS}} \overset{\mathrm{wp1}}{=} \theta$.

Proof.

Note that (6.19) implies that

$$\Phi_{y,u_0,\ell}^{\mathrm{T}} \Phi_{y,u_0,\ell} \hat{\theta}_\ell^{\mathrm{LS}} = \Phi_{y,u_0,\ell}^{\mathrm{T}} \Psi_{y,\ell}. \tag{6.37}$$

Dividing (6.37) by ℓ and letting $\ell \to \infty$ yields

$$\lim_{\ell\to\infty} \frac{1}{\ell} \Phi_{y,u_0,\ell}^{\mathrm{T}} \Phi_{y,u_0,\ell} \hat{\theta}_\ell^{\mathrm{LS}} = \lim_{\ell\to\infty} \frac{1}{\ell} \Phi_{y,u_0,\ell}^{\mathrm{T}} \Psi_{y,\ell}. \tag{6.38}$$

Using (6.22) we can write

$$\lim_{\ell\to\infty} \frac{1}{\ell} \Phi_{y,u_0,\ell}^{\mathrm{T}} \Phi_{y,u_0,\ell} \hat{\theta}_\ell^{\mathrm{LS}} \overset{\mathrm{wp1}}{=} \Gamma \lim_{\ell\to\infty} \hat{\theta}_\ell^{\mathrm{LS}} \tag{6.39}$$

Next, note that

$$\Psi_{y,\ell} = \Psi_{y_0,\ell} + \Psi_{w_0,\ell}, \tag{6.40}$$

where

$$\Psi_{y_0,\ell} = \Phi_{y_0,u_0,\ell}\theta, \quad \Psi_{w_0,\ell} = \Phi_{w_0,w,\ell}\theta_w, \tag{6.41}$$

and

$$\Phi_{y_0,u_0,\ell} \triangleq \begin{bmatrix} \phi_{y_0,u_0}(\tilde{n}) \\ \vdots \\ \phi_{y_0,u_0}(\ell) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times(2n_0+1)}, \Phi_{w_0,w,\ell} \triangleq \begin{bmatrix} \phi_{w_0,w}(\tilde{n}) \\ \vdots \\ \phi_{w_0,w}(\ell) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times(2\bar{n}+1)}.$$

Using (6.40) and (6.41) we can write

$$\begin{aligned} \Phi_{y,u_0,\ell}^{\mathrm{T}} \Psi_{y,\ell} &= \Phi_{y,u_0,\ell}^{\mathrm{T}} \Psi_{y_0,\ell} + \Phi_{y,u_0,\ell}^{\mathrm{T}} \Psi_{w_0,\ell} \\ &= \Phi_{y,u_0,\ell}^{\mathrm{T}} \Phi_{y_0,u_0,\ell} \theta + \Phi_{y,u_0,\ell}^{\mathrm{T}} \Phi_{w_0,w,\ell} \theta_w, \end{aligned} \tag{6.42}$$

where

$$\Phi_{y_0,u_0,\ell} \triangleq [\Phi_{y_0,\ell} \quad \Phi_{u_0,\ell}], \qquad \Phi_{w_0,w,\ell} \triangleq [\Phi_{w_0,\ell} \quad \Phi_{w,\ell}], \tag{6.43}$$

$$\Phi_{y_0,\ell} \triangleq \begin{bmatrix} -y_0(\tilde{n}-1) & \cdots & -y_0(\tilde{n}-n_0) \\ \vdots & \cdots & \vdots \\ -y_0(\ell-1) & \cdots & -y_0(\ell-n_0) \end{bmatrix}, \quad \Phi_{w,\ell} \triangleq \begin{bmatrix} w(\tilde{n}) & \cdots & w(\tilde{n}-\bar{n}) \\ \vdots & \cdots & \vdots \\ w(\ell) & \cdots & w(\ell-\bar{n}) \end{bmatrix}, \tag{6.44}$$

$$\Phi_{w_0,\ell} \triangleq \begin{bmatrix} -w_0(\tilde{n}-1) & \cdots & -w_0(\tilde{n}-\bar{n}) \\ \vdots & \cdots & \vdots \\ -w_0(\ell-1) & \cdots & -w_0(\ell-\bar{n}) \end{bmatrix}. \tag{6.45}$$

Moreover, note that

$$\Phi_{y,\ell} = \Phi_{y_0,\ell} + \tilde{\Phi}_{w_0,\ell},$$

where

$$\tilde{\Phi}_{w_0,\ell} \triangleq \begin{bmatrix} -w_0(\tilde{n}-1) & \cdots & -w_0(\tilde{n}-n_0) \\ \vdots & \cdots & \vdots \\ -w_0(\ell-1) & \cdots & -w_0(\ell-n_0) \end{bmatrix} \in \mathbb{R}^{(\ell-\tilde{n}+1)\times n_0}. \tag{6.46}$$

Using (6.43), (6.42) can be written as

$$\Phi_{y,u_0,\ell}^{\mathrm{T}}\Psi_{y,\ell} = \begin{bmatrix} \Phi_{y,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{y,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \\ \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \end{bmatrix} \theta + \begin{bmatrix} \Phi_{y,\ell}^{\mathrm{T}}\Phi_{w_0,\ell} & \Phi_{y,\ell}^{\mathrm{T}}\Phi_{w,\ell} \\ \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{w_0,\ell} & \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{w,\ell} \end{bmatrix} \theta_w \tag{6.47}$$

Since $\mathcal{U}$ and $\mathcal{W}$ are uncorrelated zero-mean random processes, it follows that

$$\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} \overset{\mathrm{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell}, \tag{6.48}$$

$$\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \overset{\mathrm{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell}, \tag{6.49}$$

$$\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y,\ell}^{\mathrm{T}}\Phi_{w_0,\ell} \overset{\mathrm{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\tilde{\Phi}_{w_0,\ell}^{\mathrm{T}}\Phi_{w_0,\ell} \overset{\mathrm{wp1}}{=} \sigma_w^2\mathcal{H}_{G_w}, \tag{6.50}$$

$$\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y,\ell}^{\mathrm{T}}\Phi_{w,\ell} \overset{\mathrm{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell}\tilde{\Phi}_{w_0,\ell}^{\mathrm{T}}\Phi_{w,\ell} \overset{\mathrm{wp1}}{=} -\sigma_w^2\mathcal{H}_w, \tag{6.51}$$

$$\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{w_0,\ell} \overset{\mathrm{wp1}}{=} 0_{n_0\times\bar{n}}, \quad \lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{w,\ell} \overset{\mathrm{wp1}}{=} 0_{n_0\times\bar{n}+1}. \tag{6.52}$$

Using (6.48)–(6.52), dividing (6.47) by ℓ, and letting $\ell\to\infty$ yields

$$\begin{aligned}\lim_{\ell\to\infty} \frac{1}{\ell}\Phi_{y,u_0,\ell}^{\mathrm{T}}\Psi_{y,\ell} &\overset{\mathrm{wp1}}{=} \lim_{\ell\to\infty} \frac{1}{\ell} \begin{bmatrix} \Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{y_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \\ \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{y_0,\ell} & \Phi_{u_0,\ell}^{\mathrm{T}}\Phi_{u_0,\ell} \end{bmatrix} \theta + \begin{bmatrix} \sigma_w^2\mathcal{H}_{G_w} & -\sigma_w^2\mathcal{H}_w \\ 0_{(n_0+1)\times\bar{n}} & 0_{(n_0+1)\times(\bar{n}+1)} \end{bmatrix} \theta_w \\ &= \Gamma_0\theta + \begin{bmatrix} \sigma_w^2\mathcal{H}_{G_w}D_w - \sigma_w^2\mathcal{H}_wN_w \\ 0_{(n_0+1)\times 1} \end{bmatrix}. \end{aligned} \tag{6.53}$$

Using (6.39) and (6.53), (6.38) yields

$$\Gamma \lim_{\ell\to\infty} \hat{\theta}_\ell^{\mathrm{LS}} = \Gamma_0\theta + \begin{bmatrix} \sigma_w^2\mathcal{H}_{G_w}D_w - \sigma_w^2\mathcal{H}_wN_w \\ 0_{(n_0+1)\times 1} \end{bmatrix}. \qquad \square$$

Theorem 6.1 assumes that the signals $\mathcal{U}_0$ and $\mathcal{W}$ used for regression are uncorrelated. These signals are correlated for the direct closed-loop identification architectures considered in Section 4, but they are uncorrelated for the indirect closed-loop architectures considered in Section 5. Note that, depending on the closed-loop identification architecture, the model input u_0 might or might not represent the control signal.

Assuming that Γ is positive definite, Theorem 6.1 shows that the bias b in the asymptotic least squares estimate $\lim_{\ell\to\infty}\hat{\theta}_\ell^{\mathrm{LS}}$ is given by

$$b \stackrel{\triangle}{=} \lim_{\ell\to\infty}\hat{\theta}_\ell^{\mathrm{LS}} - \theta = \left(\Gamma^{-1}\Gamma_0 - I_{2n_0+1}\right)\theta + \Gamma^{-1}\begin{bmatrix} \sigma_w^2\mathcal{H}_{G_w}D_w - \sigma_w^2\mathcal{H}_w N_w \\ 0_{(n_0+1)\times 1} \end{bmatrix}. \quad (6.54)$$

By setting $b = 0$, Theorem 6.1 gives the necessary and sufficient condition *i*) under which the least squares estimate is consistent. As a special case, assume that $\bar{n} = n_0$, $N_w = 1$, and $D_w = D_0$. Then $\tilde{\mathcal{H}}_{G_w} = \mathcal{H}_{G_w}$ and, since the relative degree of G_w is n_0, it follows that $\mathcal{H}_w = 0_{n_0\times(n_0+1)}$. Therefore, *i*) is satisfied, and thus $b = 0$. This case is considered in [95, p. 205] and [83, p. 186].

6.3.2 Prediction Error Methods

For prediction error methods, y is written as

$$y(k) = G_0(\mathbf{q})u_0(k) + w_0(k), \quad (6.55)$$

where

$$w_0(k) = G_w(\mathbf{q})w(k) \quad (6.56)$$

and G_w represents the noise dynamics, the order of G_w is $\bar{n}$, and w is zero-mean white noise. The one-step-ahead predictor of (6.55) is defined by [95]

$$\hat{y}(k|\hat{\theta}_\ell, \hat{\theta}_{w,\bar{n},\ell}) \stackrel{\triangle}{=} (1 - G_w(\mathbf{q}, \hat{\theta}_{w,\bar{n},\ell})^{-1})u_0(k) + G_w(\mathbf{q}, \hat{\theta}_{w,\bar{n},\ell})^{-1}G_0(\mathbf{q}, \hat{\theta}_\ell)y(k), \quad (6.57)$$

where $G_0(\mathbf{q}, \hat{\theta}_\ell)$ and $G_w(\mathbf{q}, \hat{\theta}_{w,\bar{n},\ell})$ are models of $G_0(\mathbf{q})$ and $G_w(\mathbf{q})$, respectively, and $G_w(\mathbf{q}, \hat{\theta}_{w,\bar{n},\ell})$ is minimum phase.

The prediction error is defined as

$$\varepsilon(k|\hat{\theta}_\ell, \hat{\theta}_{w,\bar{n},\ell}) \stackrel{\triangle}{=} y(k) - \hat{y}(k|\hat{\theta}_\ell, \hat{\theta}_{w,\bar{n},\ell}). \quad (6.58)$$

From the one-step-ahead predictor model (6.57) the prediction error estimate $\hat{\theta}_\ell^{\mathrm{PEM}}$ of θ is

$$\hat{\theta}_\ell^{\mathrm{PEM}} = \underset{\bar{\theta}\in\mathbb{R}^{2n_0+1}, \bar{\theta}_{w,\bar{n}}\in\mathbb{R}^{2\bar{n}+1}}{\arg\min} V_\ell(\bar{\theta}_\ell, \bar{\theta}_{w,\bar{n},\ell}), \quad (6.59)$$

$$V_\ell(\bar{\theta}_\ell, \bar{\theta}_{w,\bar{n},\ell}) = \frac{1}{\ell}\sum_{k=1}^{\ell} \|\varepsilon(k|\bar{\theta}_\ell, \bar{\theta}_{w,\bar{n},\ell})\|^2, \quad (6.60)$$

In Sections 6.6 and 6.7 we use the Matlab System Identification Toolbox [96] to obtain PEM-based estimates of the plant NMP zeros. Note that the predictor filters in (6.57) are functions of both G_0 and the noise model G_w. Unlike LS, PEM estimates a model of the noise dynamics G_w. As discussed in [83, p. 209], if the order of the model is underparameterized, the parameter estimates will converge to a minimum point of the asymptotic loss function, and thus will not give consistent estimates. Therefore, for all examples in this chapter, we choose the order of the model of G_w to be equal to order n_0 of G_0. When utilizing PEM within the System Identification Toolbox, we use the Box-Jenkins model structure. Through numerical testing (not shown) the choices of model structures yielded similar trends for each of the examples we consider and

are thus omitted in the results.

6.4 Direct Closed-Loop Architectures

Consider the discrete-time closed-loop system in Fig.6.1 consisting of the SISO transfer functions, plant G of order n, and controller G_c of order n_c. Note that the plant and controller are defined as

$$G(\mathbf{q}) \triangleq \frac{N(\mathbf{q})}{D(\mathbf{q})}, \qquad G_c(\mathbf{q}) \triangleq \frac{N_c(\mathbf{q})}{D_c(\mathbf{q})}. \tag{6.61}$$

The signal r is the command, y is the measured output, the error $e = r - y$, and w_0 represents sensor noise. Direct closed-loop identification is used to identify the open-loop transfer function within a closed-loop system by using measurements of the input and output of the plant. In this section we define three architectures associated with direct closed-loop identification. For all three of the direct closed-loop architectures, the estimated transfer function G_m is of order n, the plant $G = G_0$, and the noise transfer function $G_w = 1$. As stated in Section 6.3.2, the order of the PEM noise model estimate is chosen to be of order n.

6.4.1 Direct Closed-Loop Identification

In direct closed-loop identification (DCL), the plant $G = G_0$ is identified from the control input u to the output y as shown in Fig. 6.1. The measured output y is given by

$$y(k) = y_0(k) + w_0(k), \tag{6.62}$$

where

$$y_0(k) \triangleq G(\mathbf{q})u(k). \tag{6.63}$$

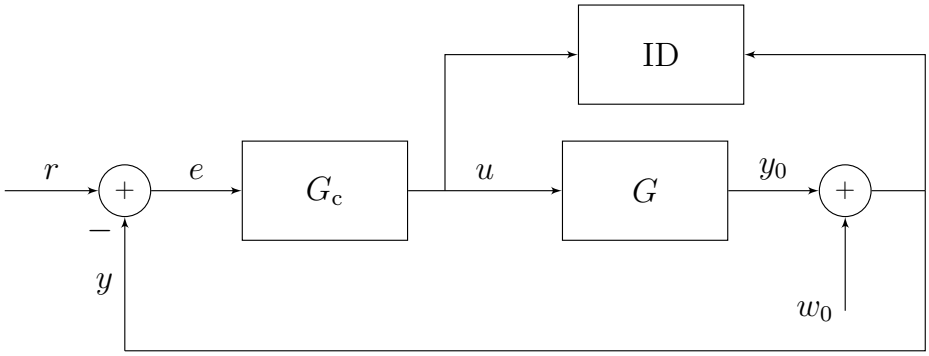


Figure 6.1: Direct closed-loop identification from u to y.

6.4.2 Standard Auxiliary Direct Closed-Loop Identification

In standard auxiliary direct closed-loop (ADCL/S) identification, the auxiliary signal v_0 is added to the controller output u_c, and $G = G_0$ is identified by using u and y, where

$$u(k) = u_c(k) + v_0(k), \tag{6.64}$$

as shown in Fig. 6.2. Note that, if $v_0 = 0$, then ADCL/S is equivalent to DCL identification.

6.4.3 Intercalated Auxiliary Direct Closed-Loop Identification

In intercalated auxiliary direct closed-loop (ADCL/I) identification, the auxiliary signal v_0 is added to the controller output u_{int} between the numerator and denomi-

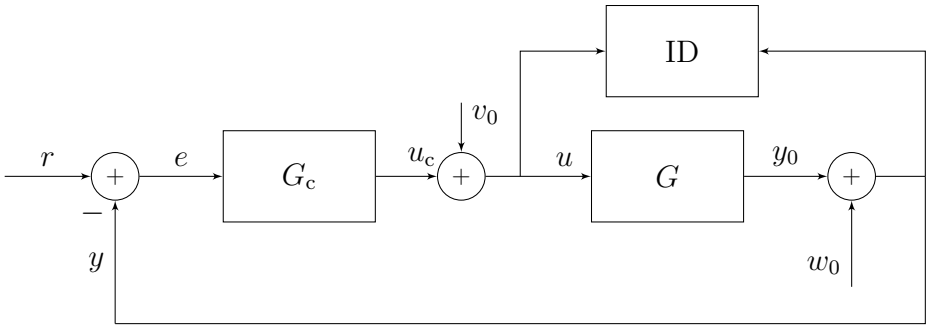


Figure 6.2: Standard auxiliary direct closed-loop identification from u to y.

nator of the controller, where

$$u_{\text{int}}(k) = \frac{N_c(\mathbf{q})}{D_c(\mathbf{q})} e(k) + \frac{\mathbf{q}^{n_c} - D_c(\mathbf{q})}{D_c(\mathbf{q})} v_0(k), \tag{6.65}$$

and $G = G_0$ is identified by using u and y as shown in Fig. 6.3, where, by using (6.64) and (6.65),

$$u(k) = \frac{N_c(\mathbf{q})}{D_c(\mathbf{q})} e(k) + \frac{\mathbf{q}^{n_c}}{D_c(\mathbf{q})} v_0(k). \tag{6.66}$$

Note that, if $v_0 = 0$, then $u = u_{\text{int}}$ and ADCL/I is equivalent to DCL identification.

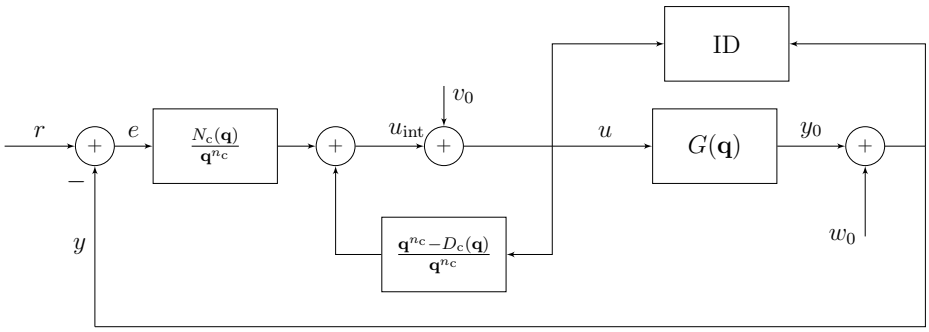


Figure 6.3: Intercalated auxiliary direct closed-loop identification from u to y.

6.5 Indirect Closed-Loop Architectures

Indirect closed-loop identification can be used to estimate the open-loop transfer function within a closed-loop system by first estimating the closed-loop system and then determining the open-loop plant by using knowledge of the controller. However, the zeros of the plant do not move when the loop is closed. Therefore, for the purpose of estimating the NMP plant zeros, it is not necessary to extract a model of the open-loop plant. In this section we define three architectures associated with indirect closed-loop identification. For all three of the indirect closed-loop architectures, the true model G_0 varies for each architecture but the order for each is the same, that is $n_0 = n + n_c$. The noise model G_w for each architecture is the sensitivity function and is also of order $\bar{n} = n + n_c$. Therefore, the estimated transfer function G_m is of order $n + n_c$, and as stated in Section 6.3.2, the order of the PEM noise model estimate is chosen to be of order n_0.

6.5.1 Indirect Closed-Loop Identification

In indirect closed-loop identification (ICL), the closed-loop transfer function from r to y is identified as shown in Fig. 6.4. The measured output y is given by

$$y(k) = T(\mathbf{q})r(k) + S(\mathbf{q})w_0(k), \tag{6.67}$$

where

$$T(\mathbf{q}) = \frac{G(\mathbf{q})G_{\mathrm{c}}(\mathbf{q})}{1 + G_{\mathrm{c}}(\mathbf{q})G(\mathbf{q})} = \frac{N(\mathbf{q})N_{\mathrm{c}}(\mathbf{q})}{D(\mathbf{q})D_{\mathrm{c}}(\mathbf{q}) + N(\mathbf{q})N_{\mathrm{c}}(\mathbf{q})}, \tag{6.68}$$

$$S(\mathbf{q}) = \frac{1}{1 + G_{\mathrm{c}}(\mathbf{q})G(\mathbf{q})} = \frac{D(\mathbf{q})D_{\mathrm{c}}(\mathbf{q})}{D(\mathbf{q})D_{\mathrm{c}}(\mathbf{q}) + N(\mathbf{q})N_{\mathrm{c}}(\mathbf{q})}. \tag{6.69}$$

Note that for ICL, $G_0(\mathbf{q}) = T(\mathbf{q})$ with $n_0 = n + n_c$, and $G_w(\mathbf{q}) = S(\mathbf{q})$.

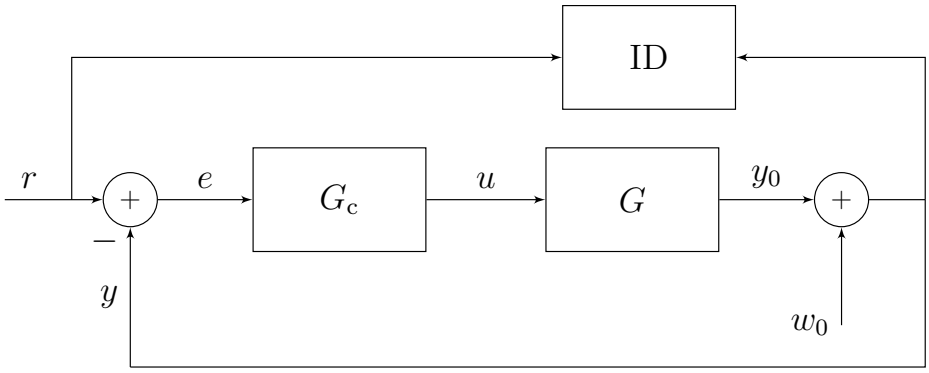


Figure 6.4: Indirect closed-loop identification from r to y.

6.5.2 Standard Auxiliary Indirect Closed-Loop Identification

In auxiliary indirect closed-loop identification (AICL) as well as in ADCL identification, we specify an auxiliary signal v_0 that is added to the controller output. Unlike ADCL identification, however, we identify the closed-loop transfer function from r to y. In standard indirect auxiliary closed-loop identification (AICL/S), the auxiliary signal v_0 is added to the controller output u_c, as shown in Fig. 6.5, where $v_0(k) = \alpha r(k)$ and α is a constant scalar value. The measured output y is given by

$$\begin{aligned} y(k) &= T(\mathbf{q})r(k) + \Gamma(\mathbf{q})v_0(k) + S(\mathbf{q})w_0(k), \\ &= (T(\mathbf{q}) + \alpha\Gamma(\mathbf{q}))r(k) + S(\mathbf{q})w_0(k) \end{aligned} \tag{6.70}$$

where T and S are defined in (6.68) and (6.69), respectively, and

$$\Gamma(\mathbf{q}) = \frac{G(\mathbf{q})}{1 + G_c(\mathbf{q})G(\mathbf{q})} = \frac{N(\mathbf{q})D_c(\mathbf{q})}{D(\mathbf{q})D_c(\mathbf{q}) + N(\mathbf{q})N_c(\mathbf{q})}, \tag{6.71}$$

Note that the zeros of the transfer functions T and Γ both include the zeros of G and the transfer function from r to y is

$$G_0(\mathbf{q}) = T(\mathbf{q}) + \alpha\Gamma(\mathbf{q}) = \frac{N(\mathbf{q})(N_c(\mathbf{q}) + \alpha D_c(\mathbf{q}))}{D(\mathbf{q})D_c(\mathbf{q}) + N(\mathbf{q})N_c(\mathbf{q})}. \tag{6.72}$$

and the noise transfer function $G_w(\mathbf{q}) = S(\mathbf{q})$.

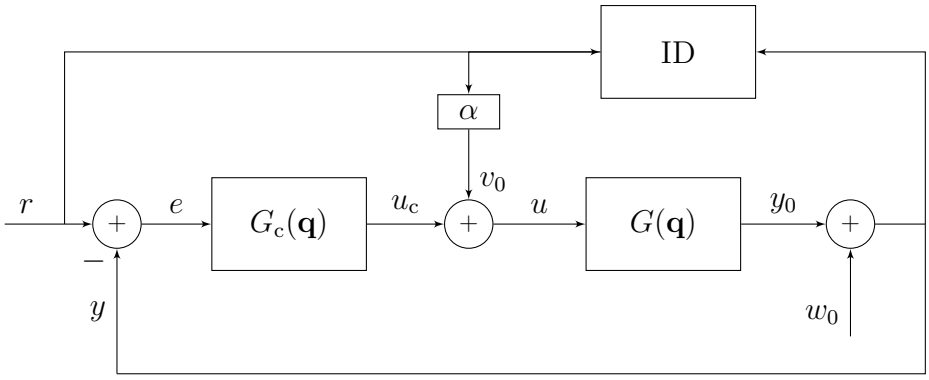


Figure 6.5: Standard auxiliary indirect closed-loop identification from r to y.

6.5.3 Intercalated Auxiliary Indirect Closed-Loop Identification

In intercalated auxiliary indirect closed-loop identification (AICL/I), the auxiliary signal v_0 is added to the controller output u_{int} between the numerator and denominator of the controller written as in (6.65) with the plant input u specified in (6.66). The transfer function from r to y is identified as shown in Fig. 6.6, where $v_0(k) = \alpha r(k)$ and α is a constant scalar.

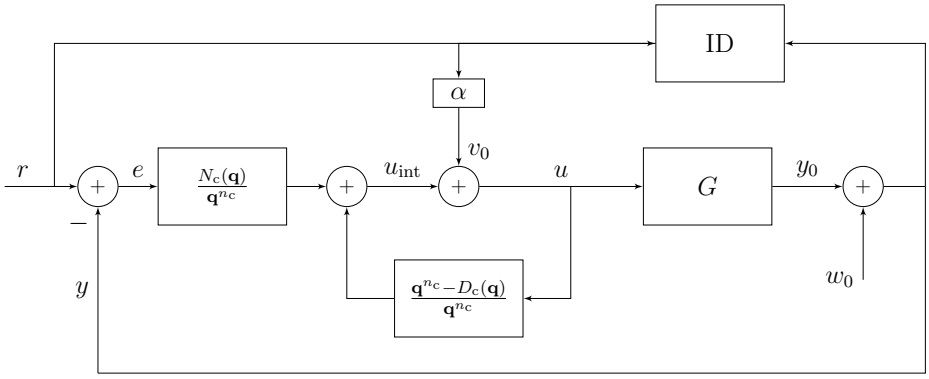


Figure 6.6: Intercalated auxiliary indirect closed-loop identification from v_0 and r to y.

The plant output y is given by

$$\begin{aligned} y(k) &= T(\mathbf{q})r(k) + \Gamma_{\text{int}}(\mathbf{q})v_0(k) + S(\mathbf{q})w_0(k), \\ &= (T(\mathbf{q}) + \alpha\Gamma_{\text{int}}(\mathbf{q}))r(k) + S(\mathbf{q})w_0(k), \end{aligned} \tag{6.73}$$

where

$$\Gamma_{\text{int}}(\mathbf{q}) = \Gamma(\mathbf{q})\frac{\mathbf{q}^{n_c}}{D_c(\mathbf{q})} = \frac{N(\mathbf{q})\mathbf{q}^{n_c}}{D(\mathbf{q})D_c(\mathbf{q}) + N(\mathbf{q})N_c(\mathbf{q})}, \tag{6.74}$$

and T and S are defined in (6.68) and (6.69), respectively. Note that Γ_{int} given by (6.74) is similar to (6.71) except that the numerator of (6.74) includes n_c zeros at 0 rather than the controller denominator. The transfer function from r to y is given by

$$G_0(\mathbf{q}) = T(\mathbf{q}) + \alpha\Gamma_{\text{int}}(\mathbf{q}) = \frac{N(\mathbf{q})(N_c(\mathbf{q}) + \alpha\mathbf{q}^{n_c})}{D(\mathbf{q})D_c(\mathbf{q}) + N(\mathbf{q})N_c(\mathbf{q})}, \tag{6.75}$$

and the noise transfer function $G_w(\mathbf{q}) = S(\mathbf{q})$. For each identification architecture, Table 6.1 summarizes the input to the estimated transfer function G_m, the estimated transfer function G_m, and the order of G_m.

Table 6.1: Summary of direct and indirect closed-loop identification architectures with inputs, estimated transfer function, and order of the estimated transfer function.

Architecture	**Input of G_m**	G_m	**Order of G_m**
DCL	$G_c(\mathbf{q})e(k)$	$G(\mathbf{q})$	n
ADCL/S	$G_c(\mathbf{q})e(k) + v_0(k)$	$G(\mathbf{q})$	n
ADCL/I	$G_c(\mathbf{q})e(k) + \frac{\mathbf{q}^{n_c}}{D_c(\mathbf{q})}v_0(k)$	$G(\mathbf{q})$	n
ICL	$r(k)$	$T(\mathbf{q})$	$n + n_c$
AICL/S	$r(k)$	$T(\mathbf{q}) + \alpha\Gamma(\mathbf{q})$	$n + n_c$
AICL/I	$r(k)$	$T(\mathbf{q}) + \alpha\Gamma_{\text{int}}(\mathbf{q})$	$n + n_c$

6.6 Numerical Investigation of Direct Closed-Loop Identification Architectures

In this section, we investigate the accuracy of the NMP-zero estimates obtained from LS and PEM for the direct closed-loop architectures. Two examples are considered, both of which involve a third-order plant with one minimum-phase zero and one NMP zero. The first plant is asymptotically stable with a pole near the unit circle, while the second plant is unstable.

Example 6.2 (Asymptotically stable, NMP plant.)**.** Consider the asymptotically stable, NMP plant

$$G(\mathbf{q}) = \frac{(\mathbf{q} - 0.6)(\mathbf{q} - 1.5)}{(\mathbf{q} - 0.1)(\mathbf{q} - 0.5)(\mathbf{q} - 0.98)}, \tag{6.76}$$

where the true NMP zero is $z_{\mathrm{NMP}} = 1.5$, and the controller is given by

$$G_{\mathrm{c}}(\mathbf{q}) = \frac{-0.5174\mathbf{q}^2 + 0.3315\mathbf{q} - 0.02795}{\mathbf{q}^3 - 0.2265\mathbf{q}^2 + 0.6855\mathbf{q} - 0.5142}. \tag{6.77}$$

The exogenous inputs r, w_0, and v_0 (when applicable) are independent zero-mean white noise signals whose standard deviations are listed in Table 6.2. For each architecture, the signal w_0 is adjusted so that the signal-to-noise ratio defined by

$$\mathrm{SNR}_{\mathrm{dB}} \triangleq 20 \log_{10} \frac{\mathrm{RMS}(y)}{\mathrm{RMS}(w_0)}, \tag{6.78}$$

where $\mathrm{RMS}(x)$ represents the root mean square of the sampled signal x, is fixed to be 31.5 dB for all simulations. This noise level corresponds to an SNR of 40.

For the direct closed-loop architectures and both identification algorithms, Fig. 6.7 shows the mean absolute value of the error in the estimated NMP zero and the corresponding standard deviation averaged over 2000 independent realizations of the

Table 6.2: Direct closed-loop architecture exogenous-signal standard deviations for Example 6.2. The standard deviation of the noise w_0 is adjusted so that the signal-to-noise ratio (6.78) is 31.5 dB for each simulation.

Architecture	**Command** r	**Auxiliary Signal** v_0	**Noise** w_0
DCL	1.2	–	0.024
ADCL/S	1.2	0.5	0.034
ADCL/I	1.2	0.5	0.032

inputs for increasing number of samples ℓ. The NMP-zero estimate is found by taking the minimum of the difference between the real part of the roots of the estimated transfer function numerator and the true zero z_{NMP}. Note that, since the input u is correlated with the noise w_0, the LS estimates are not consistent for each of the three architectures. In contrast, the accuracy of the PEM estimates improves as the number of samples increases, thus suggesting consistency. For both identification algorithms, the standard deviation decreases in a similar fashion, with PEM at a lower value.

Figure 6.7: Simulation statistics for Example 6.2 for the direct closed-loop architectures for both identification algorithms. The upper plot shows the absolute value of the mean NMP error over 2000 independent realizations of the inputs over varying number of samples. The lower plot shows the standard deviation of the NMP error.

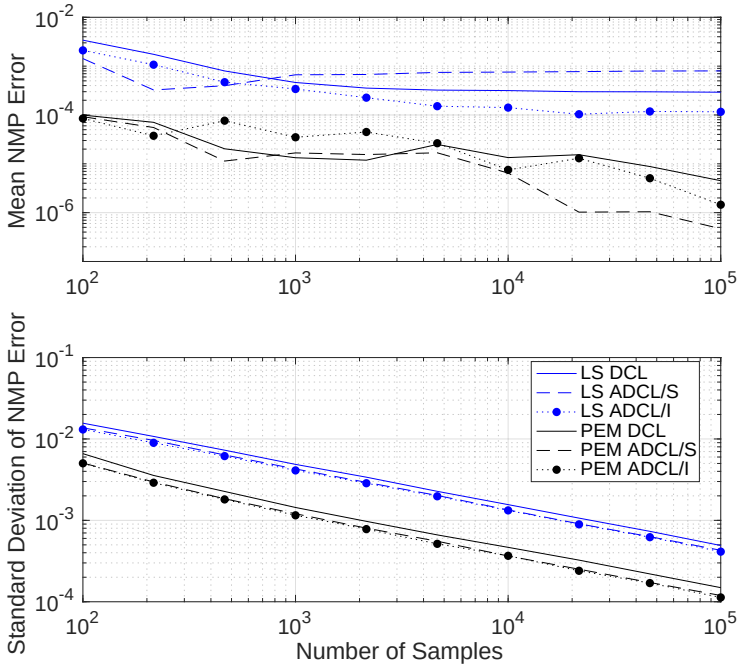


Table 6.3 compares the accuracy of the LS estimates, where the NMP-zero errors are averaged over $\ell = 10^5$ samples, with the error based on the analytical bias expression (6.54). Note that, because of the correlation between the input u and the noise w_0, the NMP-zero error based on (6.54) is inaccurate by an order of magnitude.

Table 6.3: Direct closed-loop architecture comparison between the LS NMP zero error estimates with $\ell = 10^5$ and the NMP error based on (6.54) for Example 6.2

Architecture	**Averaged NMP-Zero Error**	**NMP-Zero Error Based on** (6.54)
DCL	2.92×10^{-4}	4.54×10^{-3}
ADCL/S	7.98×10^{-4}	3.61×10^{-3}
ADCL/I	1.16×10^{-4}	3.63×10^{-3}

Example 6.6.2: *Unstable, NMP plant.* Consider the unstable, NMP plant

$$G(\mathbf{q}) = \frac{(\mathbf{q} - 0.6)(\mathbf{q} - 1.5)}{(\mathbf{q} - 0.1)(\mathbf{q} - 0.5)(\mathbf{q} - 1.3)}, \tag{6.79}$$

where the true NMP zero location $z_{\text{NMP}} = 1.5$, and the stabilizing controller is

$$G_{\text{c}}(\mathbf{q}) = \frac{-6.357\mathbf{q}^2 + 3.827\mathbf{q} - 0.3191}{\mathbf{q}^3 + 0.1312\mathbf{q}^2 + 6.943\mathbf{q} - 4.419}. \tag{6.80}$$

The input to the system, r, w_0, and v_0 (where applicable), are all independent zero mean white noise signals whose standard deviations are listed in Table 6.4. As in the previous example, w_0 is adjusted in order so that the signal-to-noise ratio is a constant 31.5 dB for all simulations.

For all three direct closed-loop architectures and both identification algorithms, Fig. 6.8 shows the absolute value of the mean errors in the estimated NMP zero

Table 6.4: Direct closed-loop architecture input signal standard deviations for Example 6.2. The nosie w_0 is regulated so that the signal-to-noise ratio (6.78) remains at 31.5 dB for each simulation.

Architecture	**Command** r	**Auxiliary Signal** v_0	**Noise** w_0
DCL	0.134	–	0.033
ADCL/S	0.134	0.5	0.147
ADCL/I	0.134	0.5	0.040

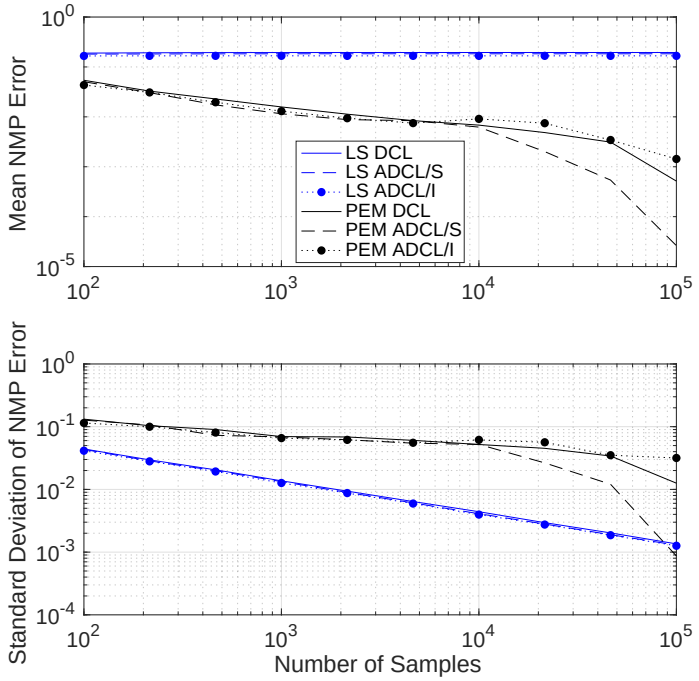
and the standard deviations based on 2000 independent realizations of the inputs for various values of ℓ. Note that, as in the last example, the LS result does not yield consistency for each of the three architectures since the input u is correlated with the noise w_0, whereas PEM has a trend of consistency. For both identification algorithms, the standard deviation decreases as the number of samples increases.

In all three direct closed-loop architectures, the input u is correlated with the noise signal w_0. Table 6.3 provides the mean of NMP-zero error obtained from LS averaged over 2000 independent realizations of the inputs at $\ell = 10^5$ samples and using the results based on (6.54). Note that, due to the correlation between the input u and the noise w_0, the NMP error based on (6.54) is inaccurate.

Table 6.5: Comparison for the direct closed-loop architectures of the LS NMP-zero error estimates with $\ell = 10^5$ and the NMP error based on (6.54) for Example 6.6.2.

Architecture	**Averaged NMP-Zero Error**	**NMP-Zero Error Based on** (6.54)
DCL	0.194	0.353
ADCL/S	0.183	0.333
ADCL/I	0.167	0.309

Figure 6.8: Simulation statistics for Example 6.6.2 for all three direct closed-loop architectures for both identification algorithms. The upper plot shows the absolute value of the mean NMP error over 2000 independent realizations of the inputs over varying number of samples. The lower plot shows the standard deviation of the NMP error.



6.7 Numerical Investigation of Indirect Closed-Loop Identification Architectures

In this section, the same two examples that are used in the previous section are used to illustrate the consistency in computing the NMP zero using both LS and PEM identification algorithms for all three indirect closed-loop architectures.

Example 6.7.1: *Asymptotically stable, NMP plant.* We consider the asymptotically stable NMP plant and controller in Example 6.1 The inputs into the system, r and w_0, are both independent zero-mean white noise signals and for the AICL/S and the AICL/I architectures, the value of $\alpha = 0.5$, such that $v_0(k) = 0.5r(k)$. As in the previous examples, the standard deviation of w_0 is adjusted so that the signal-to-noise ratio is a constant 31.5 dB for all simulations shown in Table 6.6.

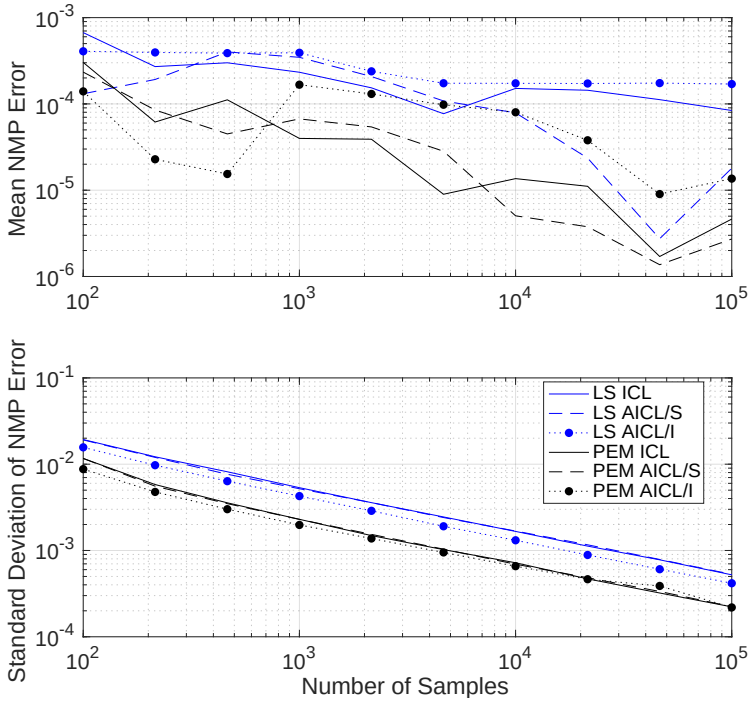
For the indirect closed-loop architectures and both identification algorithms, Fig.

Table 6.6: Indirect closed-loop architecture input signal standard deviations for Example 6.7.1. The nosie w_0 is regulated so that the signal-to-noise ratio (6.78) remains at 31.5 dB for each simulation.

Architecture	**Command** r	**Noise** w_0
ICL	1.2	0.0235
AICL/S	1.2	0.0341
AICL/I	1.2	0.0267

6.9 shows the absolute value of the mean errors in the estimated NMP zero and the standard deviations based on 2000 independent realizations of the inputs for various values of ℓ. Note that the LS result does not yield consistency for each of the three architectures. Although the input signal r and w_0 are uncorrelated and both the model and noise transfer functions have the same denominator, the matrix $\mathcal{H}_w N_w$ is not zero, and thus i) of Theorem 3.3 is not satisfied. For both identification algorithms, the standard deviation decreases as the number of samples increases.

Figure 6.9: Simulation statistics for Example 6.7.1 for all three indirect closed-loop architectures for both identification algorithms. The upper plot shows the absolute value of the mean NMP error over 2000 independent realizations of the inputs over varying number of samples. The lower plot shows the standard deviation of the NMP error.



In all of the indirect closed-loop architectures, the input r is uncorrelated with the noise signal w_0. Table 6.7 provides the mean of the NMP-zero error from LS averaged over 2000 independent realizations of the inputs at $\ell = 10^5$ samples and using the results based on (6.54). Note that, since the input r and noise w_0 are uncorrelated, the NMP error based on (6.54) agrees with the bias provided by the analytical expression given by Theorem 3.3 as $\ell \to \infty$.

Table 6.7: Comparison of indirect closed-loop identification architecture between the LS NMP zero error estimates with $\ell = 10^5$ and the NMP error based on (6.54) for Example 6.7.1.

Architecture	**Averaged NMP-Zero Error**	**NMP-Zero Error Based on** (6.54)
ICL	0.84×10^{-4}	1.28×10^{-4}
AICL/S	1.80×10^{-5}	2.76×10^{-5}
AICL/I	1.70×10^{-4}	1.63×10^{-4}

Example 6.7.2: *Unstable, NMP plant.* We consider the same unstable, NMP plant given in (6.79) and the stabilizing controller in (6.80) for the use in the comparison for all of the three indirect architectures. The inputs into the system, r and w_0, are both independent zero-mean white noise signals and for the AICL/S and the AICL/I architectures, the value of $\alpha = 0.5$, such that $v_0(k) = 0.5r(k)$. As in the previous examples, the standard deviation of w_0 is adjusted so that the signal-to-noise ratio is a constant 31.5 dB for all simulations shown in Table 6.8.

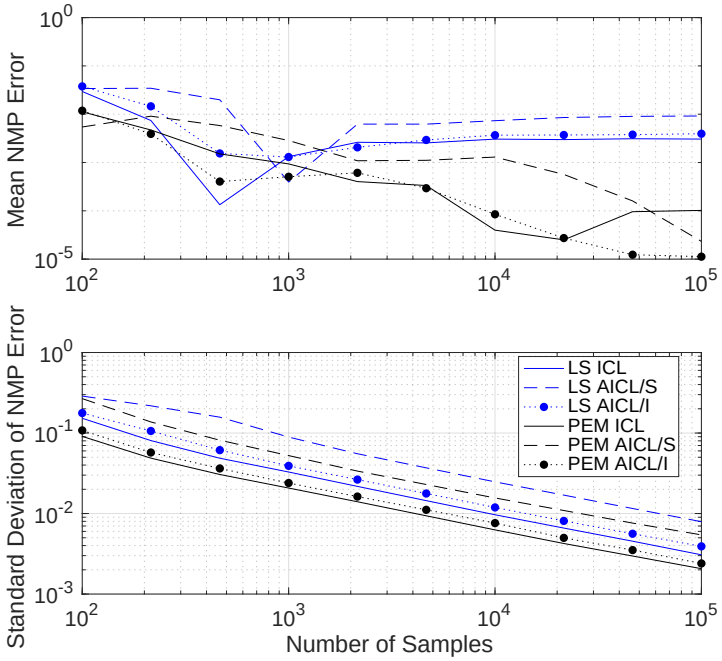
Table 6.8: Indirect closed-loop architecture input signal standard deviations for Example 6.7.2. The nosie w_0 is regulated so that the signal-to-noise ratio (6.78) remains at 31.5 dB for each simulation.

Architecture	**Command** r	**Noise** w_0
ICL	0.134	0.0334
AICL/S	0.134	0.0383
AICL/I	0.134	0.0322

For all three indirect closed-loop architectures and both identification algorithms,

Fig. 6.10 shows the absolute value of the mean errors in the estimated NMP zero and the standard deviations based on 2000 independent realizations of the inputs for various values of ℓ. Note that, for all of the three architectures, the LS estimates are not consistent. Although the input signal r and w_0 are uncorrelated and both the model and noise transfer functions have the same denominator, the matrix $\mathcal{H}_w N_w$ is not zero, and thus i) of Theorem 3.3 is not satisfied. For both identification algorithms, the standard deviation decreases as $O(\ell^{-1/2})$ as the number of samples increases.

Figure 6.10: Simulation statistics for Example 6.7.2 for all three indirect closed-loop architectures for both identification algorithms. The upper plot shows the absolute value of the mean NMP error over 2000 independent realizations of the inputs over varying number of samples. The lower plot shows the standard deviation of the NMP error.



In all three indirect closed-loop architectures, the input r is uncorrelated with the noise signal w_0. Table 6.9 provides the averages of the NMP-zero errors using the LS algorithm for 2000 independent realizations of the inputs at $\ell = 10^5$ samples and using the results based on (6.54). Note that, since the input r and noise w_0 are uncorrelated, the NMP-zero error based on (6.54) agrees with the bias provided by

the analytical expression given by Theorem 3.3 as $\ell \to \infty$.

Table 6.9: Indirect closed-loop architecture comparison between the LS NMP zero error estimates with $\ell = 10^5$ and the NMP error based on (6.54) for Example 6.7.2.

Architecture	**Averaged NMP-Zero Error**	**NMP-Zero Error Based on** (6.54)
ICL	3.05×10^{-3}	3.26×10^{-3}
AICL/S	9.19×10^{-3}	9.44×10^{-3}
AICL/I	3.94×10^{-3}	3.95×10^{-3}

6.8 Conclusions and Future Research

The objective of this chapter was to numerically investigate the effectiveness of architectures for closed-loop identification. Three direct closed-loop identification architectures and three indirect closed-loop identification architectures were considered. These architectures included standard cases with and without auxiliary signals as well as two novel architectures involving intercalated injection of the auxiliary signal. Infinite impulse response models were fit using least squares (LS) estimation, which provided a baseline method, and prediction error methods (PEM), which account for noise correlation. To simplify the study, the plant order was assumed to be known; for the indirect architectures, the auxiliary signal was chosen to be a multiple of the command; and errors-in-variables noise was not considered. The signal-to-noise ratio was normalized across all architectures. Motivated by adaptive control, the performance metric was chosen to be the accuracy of the estimate of the nonminimum-phase (NMP) zero of the plant. Two examples were considered, both of which were third-order and NMP. One plant was asymptotically stable, and the other was unstable.

As expected, for all of the architectures and for both examples, the least squares estimates exhibited bias, whereas the PEM estimates indicated consistency. In comparing architectures, the numerical results do not allow definitive conclusions, but some observations can be made. Based on PEM for direct closed-loop identification, standard injection of the auxiliary signal appears to be advantageous; direct closed-loop and auxiliary direct closed-loop with intercalated injection provide roughly the same accuracy. For indirect closed-loop identification, the conclusions are less clear. For the asymptotically stable plant, standard injection of the auxiliary signal appears to be advantageous, whereas, for the unstable plant, intercalated injection appears advantageous.

In view of the practical importance of closed-loop identification, further investigation is warranted. A more detailed study would include consideration of instrumental variables as an alternative to PEM, uncertainty in the plant order, EIV noise, and higher order plants with multiple NMP zeros.

CHAPTER 7

Retrospective Cost Adaptive Control and Closed-Loop Identification for Target Model Construction

7.1 Introduction

The complex aerodynamics encountered by flight vehicles typically requires some form of gain scheduling, where the feedback control law is tailored to the current flight condition. By using wind tunnel data to parameterize the force and moment coefficients as functions of angle of attack, sideslip, and velocity, the feedback control gains can be smoothly modulated to provide the desired closed-loop response over a wide flight envelope [97, 98].

Although this approach is often highly successful, errors in the aerodynamic forces and moments as well as unsteady effects can lead to degradation in the desired closed-loop performance. In order to overcome these effects, robust controllers can be gain scheduled [47, 99, 100], where, for each flight condition, the robust controller must account for the worst-case variation of the aerodynamic effects. This approach depends on knowledge of the possible variations of the forces and moments and, assuming that these variations can be modeled, may sacrifice performance under nominally modeled conditions.

As an alternative approach, adaptive control methods have been widely applied to aerospace systems [5]. Although there is no precise definition of an adaptive controller, an adaptive controller can be viewed as a robust control law that modifies its gains in response to the actual plant dynamics, thereby overcoming the loss in performance associated with robust control.

A key distinction between adaptive control and gain-scheduled control is the fact the gain-scheduled controller gains change according to the autopilot logic. This protocol facilitates flight certification and performance guarantees. In contrast, the gains of an adaptive controller can change "on the fly" in response to the actual aerodynamics encountered by the vehicle, whether or not these are modeled or even modelable. Consequently, it is challenging to validate the flight performance of an adaptive autopilot. These points are discussed in detail in [101–103].

To overcome these challenges, the present chapter investigates a hybrid approach, called *semi-adaptive control*, where the baseline controller is gain-scheduled and an adaptive controller is used to assist the baseline controller when its performance is degraded for any reason. To illustrate this approach, we consider the well-known benchmark problem of a planar three-degree-of-freedom (3DOF) missile [12, 48]. Assuming accurately modeled aerodynamics scheduled on Mach number and angle of attack, a three-loop autopilot (3LA) is designed in [99] to provide satisfactory and reliable performance over a broad flight envelope. For the adaptive component of the semi-active controller, we consider retrospective cost adaptive control (RCAC). The theoretical development of RCAC is detailed in [8, 24, 104], and application to the NASA GTM Model is presented in [11, 105]. In preliminary studies, RCAC was applied to the 3DOF missile in [12, 13, 37].

The contents within the present chapter goes beyond [12, 13, 37] by taking advantage of recent developments described in [36]. In particular, it is shown in [36] that the retrospective cost function used by RCAC is based on the residual of a fit

between a specific closed-loop transfer function (called the *intercalated transfer function*) and the user-specified target model. The intercalated transfer function arises due to the way in which the virtual controller perturbation, due to the adaptation, is injected into the closed-loop system. Since plant zeros are invariant under feedback, it immediately becomes clear why plant nonminimum-phase (NMP) zeros must be reproduced in the target model: if these zeros are not included in the target model, RCAC may cancel them.

For the 3DOF missile, it is possible to compute the NMP zeros as a function of angle of attack and Mach number at each linearized flight condition. Although this could be done, the estimates of the NMP zeros depend on the available model and thus maybe erroneous due to model uncertainty. Therefore, we take an alternative approach, where the NMP zeros of the missile are estimated online and the estimates are used to update the target model at each time step. The contribution of this chapter is thus a detailed study of the feasibility of semi-adaptive control, where RCAC is used along with concurrent estimation of the plant NMP zeros in order to update the target model. This approach is demonstrated using a fully nonlinear simulation of the 3DOF missile.

Chapter Nomenclature

m	=	mass, kg
I_{yy}	=	moment of inertia, kg-m^2
S_{ref}	=	reference area, m^2
d_{ref}	=	reference length, m
d_{IMU}	=	distance of the IMU from the CG along the body frame x-axis, m
U	=	inertial velocity component along the body frame x-axis of the CG, m/s

W	=	inertial velocity component along the body frame z-axis of the CG, m/s
V	=	magnitude of the velocity, m/s
M	=	Mach number
α	=	angle of attack, rad
γ	=	flight path angle, rad
θ_{m}	=	pitch angle, rad
q	=	pitch rate, rad/s
F_x	=	force about the CG along the body frame x-axis, N
C_x	=	aerodynamic force coefficient along the body frame x-axis
F_z	=	force about the CG along the body frame z-axis, N
C_z	=	aerodynamic force coefficient along the body frame z-axis
M_y	=	moment about the body frame y-axis, N-m
C_m	=	aerodynamic moment coefficient along the body frame y-axis
$\bar{q}$	=	dynamic pressure, $\mathrm{N/m^2}$
ρ	=	air density, $\mathrm{kg/m^3}$
δ_{a}	=	actuated fin deflection angle, rad
a	=	altitude dependent speed of sound, m/s
ρ	=	altitude dependent air density, $\mathrm{kg/m^3}$
q_{meas}	=	measured pitch rate, rad/s
$A_{z,\mathrm{meas}}$	=	measured normal acceleration from the inertial measurement unit, $\mathrm{m/s^2}$

7.2 The Adaptive Servo Problem

Consider the discrete-time, linear time-invariant system

$$x(k+1) = Ax(k) + Bu(k), \tag{7.1}$$

$$y_0(k) = Cx(k) + Du(k) \tag{7.2}$$

$$y(k) = y_0(k) + w(k), \tag{7.3}$$

$$e(k) = y(k) - r(k), \tag{7.4}$$

where $x(k) \in \mathbb{R}^n$ is the state, $y(k) \in \mathbb{R}$ is the measurement, $u(k) \in \mathbb{R}$ is the control input, $r(k) \in \mathbb{R}$ is the reference command, $w(k) \in \mathbb{R}$ is the sensor noise, and $e(k) \in \mathbb{R}$ is the performance variable. The output of the system y_0 can be written in terms of the forward-shift operator $\mathbf{q}$ as

$$y_0(k) = G(\mathbf{q})u(k), \tag{7.5}$$

where

$$G(\mathbf{q}) \triangleq C(\mathbf{q}I_n - A)^{-1}B + D = \frac{N(\mathbf{q})}{D(\mathbf{q})} \tag{7.6}$$

Furthermore, the controller has the form

$$u(k) = G_{\mathrm{c},k}(\mathbf{q})e(k) = \frac{N_{\mathrm{c},k}(\mathbf{q})}{D_{\mathrm{c},k}(\mathbf{q})}e(k), \tag{7.7}$$

where $G_{\mathrm{c},k}(\mathbf{q}) = N_{\mathrm{c},k}(\mathbf{q})/D_{\mathrm{c},k}(\mathbf{q})$ is updated at each time step based on the performance metric $e(k)$. Figure 7.1 illustrates the adaptive servo problem.

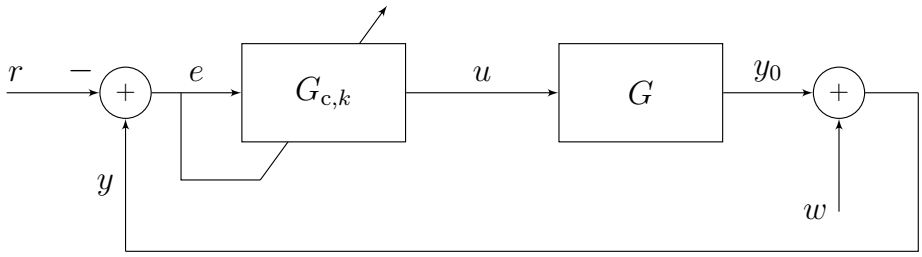


Figure 7.1: The adaptive servo problem.

7.3 The Retrospective Cost Adaptive Control Algorithm

In this section, the retrospective cost adaptive control (RCAC) algorithm is presented for application on the adaptive servo problem shown in Figure 7.1. The system, G is assumed to be single-input, single-output (SISO) but extensions to RCAC operating on multiple-input, multiple-output systems are discussed in [8, 24, 106]. RCAC minimizes the measured error e, which may be corrupted by noise.

7.3.1 The Controller Structure

The controller $G_{c,k}$ is constructed as a strictly proper time-series dynamic compensator of order n_c, such that the control $u(k)$ is given as

$$u(k) = \sum_{i=1}^{n_c} P_i(k)u(k-i) + \sum_{i=1}^{n_c} Q_i(k)e(k-i), \tag{7.8}$$

where $P_i(k)$, $Q_i(k) \in \mathbb{R}$ are the controller coefficients. The transfer function of the controller from e to u is given by

$$G_{c,k}(\mathbf{q}) = \left(\mathbf{q}^{n_c} - P_1(k)\mathbf{q}^{n_c-1} - \cdots - P_{n_c}(k)\right)^{-1} \left(Q_1(k)\mathbf{q}^{n_c-1} + \cdots + Q_{n_c}(k)\right). \tag{7.9}$$

We focus in SISO controllers, and hence $G_{c,k}$ is written as

$$G_{c,k}(\mathbf{q}) = \frac{Q_1(k)\mathbf{q}^{n_c-1} + \cdots + Q_{n_c}(k)}{\mathbf{q}^{n_c} - P_1(k)\mathbf{q}^{n_c-1} - \cdots - P_{n_c}(k)} = \frac{N_{c,k}(\mathbf{q})}{D_{c,k}(\mathbf{q})}. \tag{7.10}$$

Note that the controller coefficients in (7.8) are time-dependent, and thus $G_{\mathrm{c},k}$ is a linear time-varying controller. The controller (7.8) is expressed as

$$u(k) = \phi(k)\theta(k), \tag{7.11}$$

where the regressor vector $\phi(k)$ and controller coefficient vector $\theta(k)$ is defined as

$$\phi(k) \triangleq \begin{bmatrix} u(k-1) & \cdots & u(k-n_{\mathrm{c}}) & e(k-1) & \cdots & e(k-n_{\mathrm{c}}) \end{bmatrix} \in \mathbb{R}^{1\times l_\theta}, \tag{7.12}$$

$$\theta(k) \triangleq \begin{bmatrix} P_1(k) & \cdots & P_{n_{\mathrm{c}}}(k) & Q_1(k) & \cdots & Q_{n_{\mathrm{c}}}(k) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{l_\theta}, \tag{7.13}$$

where $l_\theta \triangleq 2n_{\mathrm{c}}$. As explained in Section 7.3.3, the controller can be written as

$$u(k) = \left(1 - \frac{D_{\mathrm{c},k}(\mathbf{q})}{\mathbf{q}^{n_{\mathrm{c}}}}\right) u(k) + \frac{N_{\mathrm{c},k}(\mathbf{q})}{\mathbf{q}^{n_{\mathrm{c}}}} e(k), \tag{7.14}$$

and a block diagram representation is shown in Figure 7.2.

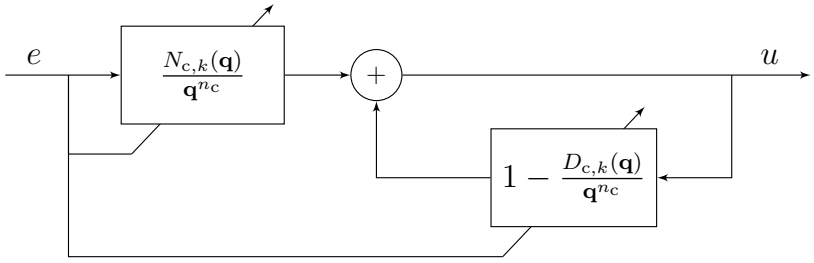


Figure 7.2: Alternative representation of the adaptive controller architecture.

7.3.2 The Retrospective Performance Variable

The retrospective performance variable is defined as

$$\hat{z}(k,\hat{\theta}) \triangleq z(k) - G_{\mathrm{f}}(\mathbf{q})[u(k) - \phi(k)\hat{\theta}], \tag{7.15}$$

where $z(k) = e(k)$ is the performance variable, $\hat{\theta} \in \mathbb{R}^{l_\theta}$, and $G_{\mathrm{f}}(\mathbf{q})$ is a filter specified below. The rationale underlying (7.15) is to replace the control $u(k)$ with $\phi(k)\hat{\theta}^*$, where $\hat{\theta}^*$ is the vector of retrospectively optimized controller coefficients obtained by minimizing the retrospective cost function in Section 7.3.4. The updated controller coefficient are given by $\theta(k+1) = \hat{\theta}^*$, and the control implemented at step $k+1$ is

$$u(k+1) = \phi(k+1)\theta(k+1). \tag{7.16}$$

The strictly proper filter G_{f}, which is of order n_{f} and has the form

$$G_{\mathrm{f}}(\mathbf{q}) \stackrel{\triangle}{=} D_{\mathrm{f}}^{-1}(\mathbf{q})N_{\mathrm{f}}(\mathbf{q}), \tag{7.17}$$

$$= \frac{N_1\mathbf{q}^{n_{\mathrm{f}}-1} + \cdots + N_{n_{\mathrm{f}}}}{\mathbf{q}^{n_{\mathrm{f}}} + D_1\mathbf{q}^{n_{\mathrm{f}}-1} + \cdots + D_{n_{\mathrm{f}}}}, \tag{7.18}$$

and is constructed in Section 7.4 based on required modeling information. For reasons given in Section 7.3.3, G_{f} is referred to as the *target model*. Using (7.17), equation (7.15) implies that

$$D_{\mathrm{f}}(\mathbf{q})\hat{z}(k,\hat{\theta}) = D_{\mathrm{f}}(\mathbf{q})z(k) - N_{\mathrm{f}}(\mathbf{q})[u(k) - \phi(k)\hat{\theta}]. \tag{7.19}$$

Defining

$$\hat{z}_{\mathrm{f}}(k,\hat{\theta}) \stackrel{\triangle}{=} D_{\mathrm{f}}(\mathbf{q})\hat{z}(k,\hat{\theta}), \quad z_{\mathrm{f}}(k) \stackrel{\triangle}{=} D_{\mathrm{f}}(\mathbf{q})z(k), \tag{7.20}$$

$$\phi_{\mathrm{f}}(k) \stackrel{\triangle}{=} N_{\mathrm{f}}(\mathbf{q})\phi(k), \quad u_{\mathrm{f}}(k) \stackrel{\triangle}{=} N_{\mathrm{f}}(\mathbf{q})u(k), \tag{7.21}$$

(7.19) can be written as

$$\hat{z}_{\mathrm{f}}(k,\hat{\theta}) = z_{\mathrm{f}}(k) - u_{\mathrm{f}}(k) + \phi_{\mathrm{f}}(k)\hat{\theta}. \tag{7.22}$$

7.3.3 The Target Model

By minimizing a cost function based on the retrospective performance variable (7.15), the controller coefficient vector $\hat{\theta}$ is determined such that $G_{\mathrm{f}}(\mathbf{q})[u(k)-\phi(k)\hat{\theta}]$ provides the best fit to the performance variable $z(k)$. In other words, $\hat{\theta}$ is chosen such that the transfer function from $u(k)-\phi(k)\hat{\theta}$ to $z(k)$ matches $G_{\mathrm{f}}(\mathbf{q})$. In terms of the optimal controller coefficient vector $\hat{\theta}^*$, (7.15) is written as

$$\hat{z}(k,\hat{\theta}) \triangleq z(k) - G_{\mathrm{f}}(\mathbf{q})[u(k)-\phi(k)\hat{\theta}^*]. \tag{7.23}$$

Defining

$$\bar{u}(k) \triangleq u(k) - u^*(k), \tag{7.24}$$

where $u^*(k) = \phi(k)\hat{\theta}^*$, (7.23) can be rewritten as

$$\hat{z}(k,\hat{\theta}) = z(k) - G_{\mathrm{f}}(\mathbf{q})\bar{u}(k). \tag{7.25}$$

It follows from (7.24) that

$$u(k) = u^*(k) + \bar{u}(k), \tag{7.26}$$

and thus the controller (7.8) can be written as

$$u^*(k) = \sum_{i=1}^{n_{\mathrm{c}}} P_i^*(k)u^*(k-i) + \sum_{i=1}^{n_{\mathrm{c}}} P_i^*(k)\bar{u}(k-i) + \sum_{i=1}^{n_{\mathrm{c}}} Q_i^*(k)e(k-i). \tag{7.27}$$

Hence

$$u^*(k) = \left(1 - \frac{D_{\mathrm{c},k}^*(\mathbf{q})}{\mathbf{q}^{n_{\mathrm{c}}}}\right)[u^*(k)+\bar{u}(k)] + \frac{N_{\mathrm{c},k}^*(\mathbf{q})}{\mathbf{q}^{n_{\mathrm{c}}}}e(k), \tag{7.28}$$

where

$$D^*_{\mathrm{c},k}(\mathbf{q}) = \mathbf{q}^{n_\mathrm{c}} - P^*_1(k)\mathbf{q}^{n_\mathrm{c}-1} - \cdots - P^*_{n_\mathrm{c}}(k), \tag{7.29}$$

$$N^*_{\mathrm{c},k}(\mathbf{q}) = Q^*_1(k)\mathbf{q}^{n_\mathrm{c}-1} + \cdots + Q^*_{n_\mathrm{c}}(k). \tag{7.30}$$

In (7.26), $u(k)$ is written as the sum of the pseudo control input $u^*(k)$ and the virtual control perturbation $\bar{u}(k)$. Note that neither of these signals is explicitly used in RCAC. Figure 7.3 represents the controller (7.28) within the adaptive servo problem.

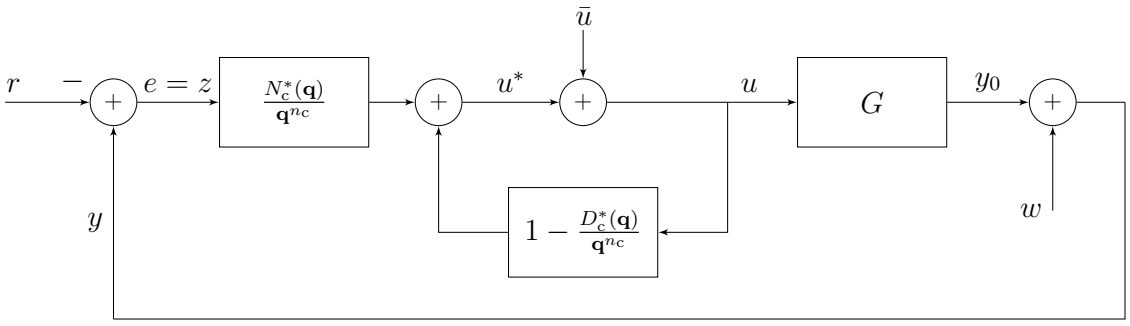


Figure 7.3: The adaptive servo problem based on the controller (7.28).

It can be seen from Figure 7.3 that the closed-loop transfer functions from the inputs r, $\bar{u}$, and w to z are given by

$$z(k) = -\tilde{G}_{zr}(\mathbf{q})r(k) + \tilde{G}_{z\bar{u}}(\mathbf{q})\bar{u}(k) + \tilde{G}_{zw}(\mathbf{q})w(k), \tag{7.31}$$

where

$$\tilde{G}_{zr}(\mathbf{q}) = \tilde{G}_{zw}(\mathbf{q}) \triangleq \frac{D(\mathbf{q})D_\mathrm{c}(\mathbf{q})}{D(\mathbf{q})D_\mathrm{c}(\mathbf{q}) - N(\mathbf{q})N_\mathrm{c}(\mathbf{q})}, \tag{7.32}$$

$$\tilde{G}_{z\bar{u}}(\mathbf{q}) \triangleq \frac{N(\mathbf{q})\mathbf{q}^{n_\mathrm{c}}}{D(\mathbf{q})D_\mathrm{c}(\mathbf{q}) - N(\mathbf{q})N_\mathrm{c}(\mathbf{q})}. \tag{7.33}$$

As discussed in [36], (7.25) is the residual of the fit between z and the output of the target model G_f with the input $\bar{u}$. However, (7.31) shows the actual transfer function

$\tilde{G}_{z\bar{u}}$ from $\bar{u}$ to z. Since $\tilde{G}_{z\bar{u}}$ arises from the injection of the virtual control perturbation $\bar{u}$ between the numerator and denominator of $G_{\mathrm{c},k}$, $\tilde{G}_{z\bar{u}}$ is the *intercalated transfer function*. Note that the zeros of $\tilde{G}_{z\bar{u}}$ include the zeros of G. Therefore, in order to avoid inadvertent cancellation of NMP zeros, the numerator N_{f} of the target model G_{f} is constructed to include the NMP zeros of G, as discussed in Section 7.4.

7.3.4 The Retrospective Cost Function

The retrospective cost function (7.22) is optimized over a sliding window of data. Define the windows of data

$$Z(k) \triangleq \begin{bmatrix} z(k) \\ \vdots \\ z(k-p_{\mathrm{c}}+1) \end{bmatrix} \in \mathbb{R}^{p_{\mathrm{c}}}, \quad \tilde{Z}(k) \triangleq \begin{bmatrix} z(k-1) \\ \vdots \\ z(k-p_{\mathrm{n}}) \end{bmatrix} \in \mathbb{R}^{p_{\mathrm{n}}}, \tag{7.34}$$

$$\tilde{U}(k) \triangleq \begin{bmatrix} u(k-1) \\ \vdots \\ u(k-p_{\mathrm{n}}) \end{bmatrix} \in \mathbb{R}^{p_{\mathrm{n}}}, \quad \tilde{\Phi}(k) \triangleq \begin{bmatrix} \phi(k-1) \\ \vdots \\ \phi(k-p_{\mathrm{n}}) \end{bmatrix} \in \mathbb{R}^{p_{\mathrm{n}} \times l_\theta}, \tag{7.35}$$

where $p_{\mathrm{c}} \geq 1$ is the window size, and $p_{\mathrm{n}} \triangleq p_{\mathrm{c}} + n_{\mathrm{f}} - 1$. Furthermore, define the windowed retrospective performance variable as

$$\hat{Z}_{\mathrm{f}}(k,\hat{\theta}) \triangleq Z(k) + \bar{D}\tilde{Z}(k) + \bar{N}[\tilde{U}(k) - \tilde{\Phi}(k)\hat{\theta}], \tag{7.36}$$

where $\hat{Z}_{\mathrm{f}}(k,\hat{\theta}) \in \mathbb{R}^{p_{\mathrm{c}}}$, and

$$\bar{D} \triangleq \begin{bmatrix} D_1 & \cdots & D_{n_{\mathrm{f}}} & 0 & \cdots & \cdots & 0 \\ 0 & D_1 & \cdots & D_{n_{\mathrm{f}}} & \ddots & \ddots & \vdots \\ 0 & 0 & D_1 & \cdots & D_{n_{\mathrm{f}}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & D_1 & \cdots & D_{n_{\mathrm{f}}} \end{bmatrix} \in \mathbb{R}^{p_{\mathrm{c}} \times p_{\mathrm{n}}}, \tag{7.37}$$

$$\bar{N} \triangleq \begin{bmatrix} N_1 & \cdots & N_{n_{\mathrm{f}}} & 0 & \cdots & \cdots & 0 \\ 0 & N_1 & \cdots & N_{n_{\mathrm{f}}} & \ddots & \ddots & \vdots \\ 0 & 0 & N_1 & \cdots & N_{n_{\mathrm{f}}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & N_1 & \cdots & N_{n_{\mathrm{f}}} \end{bmatrix} \in \mathbb{R}^{p_{\mathrm{c}} \times p_{\mathrm{n}}}, \tag{7.38}$$

are Toeplitz matrices. We define the retrospective cost function as

$$J(k,\hat{\theta}) \triangleq \hat{Z}_{\mathrm{f}}^{\mathrm{T}}(k,\hat{\theta})\hat{Z}_{\mathrm{f}}(k,\hat{\theta}) + (\hat{\theta} - \theta(k-1))^{\mathrm{T}} R_{\delta} (\hat{\theta} - \theta(k-1)), \tag{7.39}$$

where the positive-definite matrix $R_{\delta} \in \mathbb{R}^{l_\theta \times l_\theta}$ is the learning-rate weighting. Substituting (7.36) into (7.39) yields

$$J(k,\hat{\theta}) = \hat{\theta}^{\mathrm{T}}(k) A_{\theta}(k) \hat{\theta}(k) + 2\hat{\theta}^{\mathrm{T}}(k) b_{\theta}(k) + c_{\theta}(k), \tag{7.40}$$

where

$$A_\theta(k) \triangleq \tilde{\Phi}(k)^{\mathrm{T}} \bar{N}^{\mathrm{T}} \bar{N} \tilde{\Phi}(k) + R_\delta, \tag{7.41}$$

$$b_\theta(k) \triangleq \tilde{\Phi}(k)^{\mathrm{T}} \bar{N}^{\mathrm{T}} \Psi(k) - R_\delta \theta(k-1), \tag{7.42}$$

$$c_\theta(k) \triangleq \Psi(k)^{\mathrm{T}} \Psi(k) + \theta(k-1)^{\mathrm{T}} R_\delta \theta(k-1), \tag{7.43}$$

$$\Psi(k) \triangleq Z(k) + \bar{D} \tilde{Z}(k) - \bar{N} \tilde{U}(k). \tag{7.44}$$

If $A_\theta(k)$ is positive definite, then (7.40) has a unique global minimizer $\hat{\theta}^*$ given by

$$\hat{\theta}^* \triangleq -A_\theta(k)^{-1} b_\theta(k). \tag{7.45}$$

7.4 Target-Model Construction

As explained in [36], RCAC matches a specific closed-loop transfer function $\tilde{G}_{z\bar{u}}$ to G_{f}. Consequently, RCAC may cancel NMP zeros that are not included in G_{f}. Since zeros are invariant under feedback, the NMP zeros of G also appear as NMP zeros of $\tilde{G}_{z\bar{u}}$. We thus estimate the NMP zeros of G, and then use this information to construct the numerator of the target model N_{f}. The estimation is performed online by identifying G during closed-loop operation.

7.4.1 Direct Closed-Loop Identification

Although G is operating inside a feedback loop, we perform direct identification of G, where the input and output of G are used for regression. Alternative architectures can be considered for closed-loop identification, see [107].

Figure 7.4 shows the signals used to perform the direct identification of G during closed-loop operation, which is the input u and measured output y. An external exogenous input v is used to enhance persistency and thus accuracy of the identification. The estimated model is an infinite impulse response (IIR) model of order

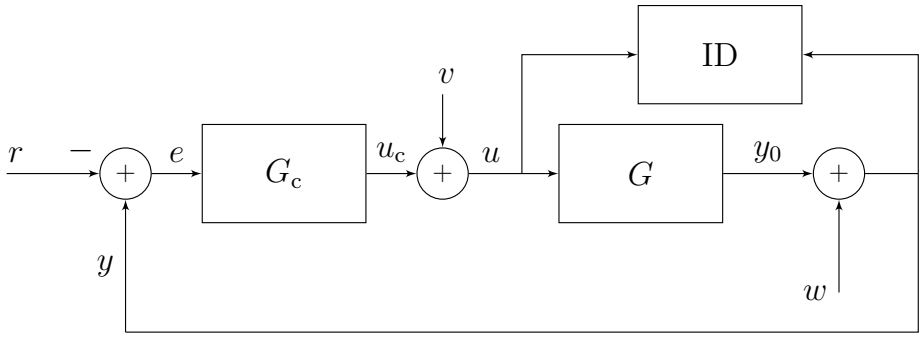


Figure 7.4: Direct closed-loop identification illustrating the signals used for regression.

$\hat{n} \geq n$, which is written as

$$\hat{y}(k) = \hat{G}(\mathbf{q})u(k) = \frac{\hat{N}(\mathbf{q})}{\hat{D}(\mathbf{q})}u(k). \tag{7.46}$$

The estimate of G can thus be written as

$$\hat{y}(k) = \sum_{i=1}^{\hat{n}} b_{\hat{n}-i}u(k-i) - \sum_{i=1}^{\hat{n}} a_{\hat{n}-i}y(k-i) = \theta_{\mathrm{ID}}\phi_{\mathrm{ID}}(k), \tag{7.47}$$

where

$$\theta_{\mathrm{ID}} = [b_{\hat{n}-1} \quad \cdots \quad b_0 \quad a_{\hat{n}-1} \quad \cdots a_0], \tag{7.48}$$

$$\phi_{\mathrm{ID}}(k) = [u(k-1) \quad \cdots \quad u(k-\hat{n}) \quad -y(k-1) \quad \cdots \quad -y(k-\hat{n})]^{\mathrm{T}}, \tag{7.49}$$

and the coefficients b_i represent the coefficients of the estimated numerator. A batch sliding window least squares algorithm is used for estimation with window size $p_c \geq 2\hat{n}$. The identification window size is chosen to be the same as the RCAC window size for convenience. Define

$$Y(k) \triangleq [y(k) \quad \cdots \quad y(k-p_c+1)] \in \mathbb{R}^{1\times p_c}, \tag{7.50}$$

$$\Phi_{\mathrm{ID}}(k) \triangleq [\phi_{\mathrm{ID}}(k) \quad \cdots \quad \phi_{\mathrm{ID}}(k-p_c+1)] \in \mathbb{R}^{2\hat{n}\times p_c}, \tag{7.51}$$

such that the least squares solution is

$$\theta_{\mathrm{ID}} = Y(k)\Phi_{\mathrm{ID}}^{\mathrm{T}}(k)(\Phi_{\mathrm{ID}}(k)\Phi_{\mathrm{ID}}^{\mathrm{T}}(k))^{-1}. \tag{7.52}$$

In order to extract the necessary information from the estimated system, we first define

$$\hat{N}(\mathbf{q}) = b_{\hat{n}-1}\mathbf{q}^{\hat{n}-1} + \cdots + b_0, \tag{7.53}$$

which is the numerator of the estimated system transfer function.

The information needed to construct the numerator N_{f} of the target model is the leading numerator coefficient $b_{\hat{n}-1}$ and all NMP zeros. Let $\hat{N}_{\mathrm{NMP}}(\mathbf{q})$ be a monic polynomial whose roots consist of the roots of $\hat{N}(\mathbf{q})$ whose magnitude is greater than 1. Then, the numerator of the target model is constructed as

$$N_{\mathrm{f}}(\mathbf{q}) = b_{\hat{n}-1}\hat{N}_{\mathrm{NMP}}(\mathbf{q}). \tag{7.54}$$

7.4.2 Adaptive Control with Concurrent Target-Model Construction

The adaptive control with concurrent target model construction works in two steps. First, the numerator of the target model N_{f} is constructed by performing the direct identification discussed in the previous section. The denominator of the target model D_{f} is constructed by the user and the roots are the desired locations of the closed-loop poles. Second, the coefficients of the adaptive controller are found by minimizing the retrospective cost function, as shown in Section 7.3.4.

7.5 Longitudinal Missile Model

In this section, RCAC is applied to the planar, three-degree-of-freedom, nonlinear longitudinal missile model shown in Figure 7.5. This model is studied in [12, 13, 37, 48] and is available in MATLAB [108]. To intercept the target, the missile is equipped with a guidance section that provides the autopilot with a normal acceleration command $A_{z,\mathrm{cmd}}$ that the missile must follow to reach its target.

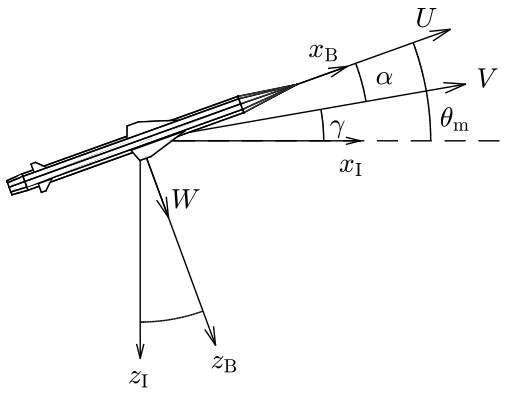


Figure 7.5: Missile kinematics, where $(\cdot)_{\mathrm{B}}$ represents the body frame and $(\cdot)_{\mathrm{I}}$ represents the inertial frame. The velocity components and angles relative to the inertial frame are shown.

The nonlinear dynamics and linear kinematics of the missile are described in the body frame as

$$m\dot{U} + mqW = F_x, \tag{7.55}$$

$$m\dot{W} - mqU = F_z, \tag{7.56}$$

$$I_{yy}\dot{Q} = M_y, \tag{7.57}$$

$$\dot{\theta}_{\mathrm{m}} = q, \tag{7.58}$$

where $U(0) = U_0$, $W(0) = W_0$, $q(0) = q_0$, and $\theta_{\mathrm{m}}(0) = \theta_{\mathrm{m}0}$. Assuming a flat Earth and neglecting gravity, the forces and moment about the missile's center of gravity

(CG) are

$$F_x = \bar{q} S_{\text{ref}} C_x, \tag{7.59}$$

$$F_z = \bar{q} S_{\text{ref}} C_z, \tag{7.60}$$

$$M_y = \bar{q} S_{\text{ref}} d_{\text{ref}} C_m, \tag{7.61}$$

where $\bar{q} = \frac{1}{2}\rho V^2$ is the dynamic pressure and the aerodynamic coefficients

$$C_x = a_x, \tag{7.62}$$

$$C_z = a_z \alpha^3 + b_z \alpha |\alpha| + c_z \left(2 - \frac{M}{3}\right) \alpha + d_z \delta_{\text{a}}, \tag{7.63}$$

$$C_m = a_m \alpha^3 + b_m \alpha |\alpha| + c_m \left(\frac{8}{3} M - 7\right) \alpha + d_m \delta_{\text{a}} + e_m q \tag{7.64}$$

are based on the force and moment data given in [48] and summarized in Table 7.2.

Normal and Axial Force Data				
$a_z = 19.373$	$b_z = -31.023$	$c_z = -9.717$	$d_z = -1.948$	$a_x = -0.300$
Pitch Moment Data				
$a_m = 40.440$	$b_m = -64.015$	$c_m = 2.922$	$d_m = -11.803$	$e_m = -1.719$

Table 7.2: Aerodynamic data

Using the relations

$$V^2 = U^2 + W^2, \qquad M = \frac{V}{a}, \qquad \tan\alpha = \frac{W}{U}, \qquad \gamma = \theta - \alpha, \tag{7.65}$$

(7.55)-(7.58) can be rewritten as

$$\dot{M} = \frac{1}{2m}\rho M^2 a S_{\text{ref}} \left(C_x \cos\,\alpha + C_z \sin\,\alpha\right), \tag{7.66}$$

$$\dot{\alpha} = \frac{1}{2m}\rho M a S_{\text{ref}} \left(C_z \cos\,\alpha - C_x \sin\,\alpha\right) + q, \tag{7.67}$$

$$\dot{\gamma} = \frac{1}{2m}\rho M a S_{\text{ref}} \left(C_z \cos\,\alpha - C_x \sin\,\alpha\right), \tag{7.68}$$

$$\dot{q} = \frac{1}{2I_{yy}}\rho M^2 a^2 S_{\text{ref}} d_{\text{ref}} C_m. \tag{7.69}$$

The pitch-rate measurement q_{meas}, normal-acceleration measurement $A_{z,\text{meas}}$, and longitudinal-acceleration measurement $A_{x,\text{meas}}$ are assumed to be provided by an on-board inertial measurement unit (IMU) and thus are available for feedback. The normal acceleration is given by

$$A_z = \frac{1}{m}F_z - \dot{q} d_{\text{IMU}}, \tag{7.70}$$

where d_{IMU} is the distance from the CG to the IMU. It is assumed that the IMU is forward of the missile CG, and thus $d_{\text{IMU}} > 0$. The actuator dynamics are represented by

$$\ddot{\delta}_{\text{a}} = -\omega_{\text{a}}^2 \delta_{\text{a}} - 2\zeta\omega_{\text{a}}\dot{\delta}_{\text{a}} + \omega_{\text{a}}^2 \delta_{\text{req}}, \tag{7.71}$$

as well as magnitude- and rate-saturation nonlinearities, where δ_{req} is the actuator setting requested by the autopilot. Table 7.3 provides the missile parameters. A standard atmospheric model is assumed, where the air density ρ and speed of sound a are altitude-dependent.

Variable	m	S_{ref}	d_{ref}	I_{yy}	d_{IMU}	ω_a	ζ_a
Value	204.022	0.041	0.228	247.436	0.500	314.160	0.700
Units	kg	m^2	m	kg-m^2	m	rad/s	-

Table 7.3: Missile parameters.

7.5.1 Three-Loop Autopilot

The three-loop autopilot (3LA) is a multi-input, single-output, proportional-integral control law that is gain scheduled on angle-of-attack and Mach number and is tuned for robust performance at an altitude of 3.0 km [99]. As shown in Figure 7.6, the 3LA receives a normal acceleration command $A_{z,\mathrm{cmd}}$ from the guidance section and uses the measured normal acceleration $A_{z,\mathrm{meas}}$ and the measured pitch rate q_{meas} from the IMU for feedback. The values K_{a}, K, K_{I}, and K_{g} are the scheduled 3LA gains based on angle-of-attack and Mach number. The output $\delta_{\mathrm{req,3LA}} = \delta_{\mathrm{req}}$ of the 3LA is the requested actuator setting, as discussed in the previous section.

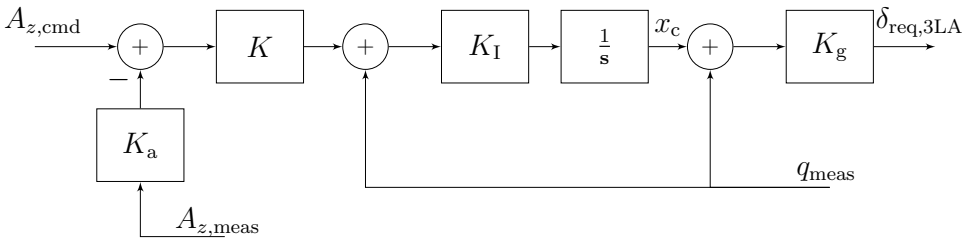


Figure 7.6: The missile three-loop autopilot architecture.

The 3LA is written as

$$\dot{x}_{\mathrm{c}} = B_{\mathrm{c},y} y_{\mathrm{meas}} + B_{\mathrm{c},A} A_{z,\mathrm{cmd}}, \tag{7.72}$$

$$\delta_{\mathrm{req,3LA}} = C_{\mathrm{c}} x_{\mathrm{c}} + D_{\mathrm{c}} y_{\mathrm{meas}}, \tag{7.73}$$

where the scalar state x_{c} is shown in Fig. 7.6, $y_{\mathrm{meas}} = [A_{z,\mathrm{meas}} \;\; q_{\mathrm{meas}}]^{\mathrm{T}}$, and

$$B_{\mathrm{c},y} = [-K_{\mathrm{a}} K K_{\mathrm{I}} \;\; K_{\mathrm{I}}], \qquad B_{\mathrm{c},A} = K K_{\mathrm{I}}, \qquad C_{\mathrm{c}} = K_{\mathrm{g}}, \qquad D_{\mathrm{c}} = [0 \;\; K_{\mathrm{g}}]. \tag{7.74}$$

7.5.2 Simultaneous Identification and Adaptive Control of the Linearized Missile

To illustrate RCAC, the missile is linearized about the trim condition corresponding to M_0, α_0, and q_0. The linearized dynamics are given by

$$\dot{x}_{\rm m} = A_{\rm m} x_{\rm m} + B_{\rm m} \delta_{\rm a}, \tag{7.75}$$

$$y_{\rm meas} = C_{\rm m} x_{\rm m} + D_{\rm m} \delta_{\rm a}, \tag{7.76}$$

$$A_{z,\rm meas} = C_{{\rm m},A} x_{\rm m} + D_{{\rm m},A} \delta_{\rm a}, \tag{7.77}$$

$$q_{\rm meas} = C_{{\rm m},q} x_{\rm m} + D_{{\rm m},q} \delta_{\rm a}, \tag{7.78}$$

where $x_{\rm m} = [\alpha \;\; q]^{\rm T}$, $y_{\rm meas} = [A_{z,\rm meas} \;\; q_{\rm meas}]^{\rm T}$, and

$$A_{\rm m} = \begin{bmatrix} \frac{\partial \dot{\alpha}}{\partial \alpha} & 1 \\ \frac{\partial \dot{q}}{\partial \alpha} & \frac{\partial \dot{q}}{\partial q} \end{bmatrix}, \quad B_{\rm m} = \begin{bmatrix} \frac{\partial \dot{\alpha}}{\partial \delta_{\rm a}} \\ \frac{\partial \dot{q}}{\partial \delta_{\rm a}} \end{bmatrix}, \quad D_{\rm m} = \begin{bmatrix} D_{{\rm m},A} \\ D_{{\rm m},q} \end{bmatrix}, \tag{7.79}$$

$$C_{\rm m} = \begin{bmatrix} C_{{\rm m},A} \\ C_{{\rm m},q} \end{bmatrix}, \quad C_{{\rm m},A} = \begin{bmatrix} \dfrac{\partial A_{z,\rm meas}}{\partial \alpha} & \dfrac{\partial A_{z,\rm meas}}{\partial q} \end{bmatrix}, \tag{7.80}$$

$$D_{{\rm m},A} = \frac{\partial A_{z,\rm meas}}{\partial \delta_{\rm a}}, \quad D_{{\rm m},q} = 0, \quad C_{{\rm m},q} = [0 \;\; 1], \tag{7.81}$$

where

$$\frac{\partial \dot{\alpha}}{\partial \alpha} = \frac{1}{2m}\rho a M_0 S_{\text{ref}}(C_{z_{\alpha_0}}\cos\,\alpha_0 - C_z \sin\,\alpha_0 - C_x \cos\,\alpha_0), \tag{7.82}$$

$$\frac{\partial \dot{\alpha}}{\partial \delta_{\text{a}}} = \frac{1}{2m}\rho a M_0 S_{\text{ref}} C_{z_{\delta_{\text{a}}}} \cos\,\alpha_0, \tag{7.83}$$

$$\frac{\partial \dot{q}}{\partial \alpha} = \frac{1}{2I_{yy}}\rho a^2 M_0^2 S_{\text{ref}} d_{\text{ref}} C_{m_{\alpha_0}}, \tag{7.84}$$

$$\frac{\partial \dot{q}}{\partial q} = \frac{1}{2I_{yy}}\rho a^2 M_0^2 S_{\text{ref}} d_{\text{ref}} C_{m_{q_0}}, \tag{7.85}$$

$$\frac{\partial \dot{q}}{\partial \delta_{\text{a}}} = \frac{1}{2I_{yy}}\rho a^2 M_0^2 S_{\text{ref}} d_{\text{ref}} C_{m_{\delta_{\text{a}}}}, \tag{7.86}$$

$$\frac{\partial A_{z,\text{meas}}}{\partial \alpha} = \frac{1}{2m}\rho a^2 M_0^2 S_{\text{ref}} C_{z_{\alpha_0}}, \tag{7.87}$$

$$\frac{\partial A_{z,\text{meas}}}{\partial q} = -\frac{1}{2I_{yy}}\rho a^2 M_0^2 S_{\text{ref}} d_{\text{ref}} C_{m_{q_0}} d_{\text{IMU}}, \tag{7.88}$$

$$\frac{\partial A_{z,\text{m}}}{\partial \delta_{\text{a}}} = \frac{1}{2m}\rho a^2 M_0^2 S_{\text{ref}} C_{z_{\delta_{\text{a}}}}. \tag{7.89}$$

The aerodynamic derivatives are given by

$$C_{z_{\alpha_0}} = 3a_z\alpha_0^2 + 2b_z|\alpha_0| + c_z\left(2 - \frac{M_0}{3}\right), \tag{7.90}$$

$$C_{m_{\alpha_0}} = 3a_m\alpha_0^2 + 2b_m|\alpha_0| + c_m\left(\frac{8}{3}M_0 - 7\right), \tag{7.91}$$

$$C_{z_{\delta_{\text{a}}}} = d_z, \qquad C_{m_{q_0}} = e_m, \qquad C_{m_{\delta_{\text{a}}}} = d_m. \tag{7.92}$$

In state space form, the linear actuator dynamics in (7.71) can be written as

$$\dot{x}_{\text{a}} = A_{\text{a}}x_{\text{a}} + B_{\text{a}}\delta_{\text{req}}, \tag{7.93}$$

$$\delta_{\text{a}} = C_{\text{a}}x_{\text{a}}, \tag{7.94}$$

where $x_{\rm a} = [\dot{\delta}_{\rm a} \;\; \delta_{\rm a}]^{\rm T}$ and

$$A_{\rm a} = \begin{bmatrix} -2\zeta_a\omega_{\rm a} & -\omega_{\rm a}^2 \\ 1 & 0 \end{bmatrix}, \quad B_{\rm a} = \begin{bmatrix} \omega_{\rm a}^2 \\ 0 \end{bmatrix}, \quad C_{\rm a} = [0 \;\; 1]. \tag{7.95}$$

In terms of the adaptive servo problem shown in Figure 7.1, G consists of the combined actuator and missile dynamics, which are given by

$$\begin{bmatrix} \dot{x}_{\rm a} \\ \dot{x}_{\rm m} \end{bmatrix} = \begin{bmatrix} A_{\rm a} & 0_{2\times2} \\ B_{\rm m}C_{\rm a} & A_{\rm m} \end{bmatrix} \begin{bmatrix} x_{\rm a} \\ x_{\rm m} \end{bmatrix} + \begin{bmatrix} B_{\rm a} \\ 0_{2\times1} \end{bmatrix} \delta_{\rm req}, \tag{7.96}$$

$$A_{z,\rm meas} = \begin{bmatrix} [0 \;\; D_{{\rm m},A}] & C_{{\rm m},A} \end{bmatrix} \begin{bmatrix} x_{\rm a} \\ x_{\rm m} \end{bmatrix}, \tag{7.97}$$

where the input $\delta_{\rm req}$ is the requested actuator setting and the output $A_{z,\rm meas}$ is the measured normal acceleration. The resulting closed-loop equations are given by

$$\begin{bmatrix} \dot{x}_{\rm a} \\ \dot{x}_{\rm m} \\ \dot{x}_{\rm c} \end{bmatrix} = \begin{bmatrix} A_{\rm a} + B_{\rm a}D_{\rm c}D_{\rm m}C_{\rm a} & B_{\rm a}D_{\rm c}C_{\rm m} & B_{\rm a}C_{\rm c} \\ B_{\rm m}C_{\rm a} & A_{\rm m} & 0 \\ B_{{\rm c},y}D_{\rm m}C_{\rm a} & B_{{\rm c},y}C_{\rm m} & 0 \end{bmatrix} \begin{bmatrix} x_{\rm a} \\ x_{\rm m} \\ x_{\rm c} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ B_{{\rm c},A} \end{bmatrix} A_{z,\rm cmd}, \tag{7.98}$$

$$A_{z,\rm ref} = \begin{bmatrix} [0 \;\; D_{{\rm m},A}] & C_{{\rm m},A} & 0 \end{bmatrix} \begin{bmatrix} x_{\rm a} \\ x_{\rm m} \\ x_{\rm c} \end{bmatrix}. \tag{7.99}$$

Figure 7.7 shows how RCAC is combined with the linearized missile model G, which consists of the linearized missile and actuator dynamics (7.96) and (7.97) with input $\delta_{\rm req} = \delta_{\rm req,RCAC}$ and output $A_{z,\rm meas}$. The goal of the setup in Figure 7.7 is to use RCAC to adaptively place closed-loop poles that match those corresponding to those of the 3LA. To do this, the loop is first closed with the 3LA, and then the

resulting closed-loop poles are used to construct $D_{\rm f}$. In particular, with $D_{\rm f}$ chosen based on the 3LA simulation, the direct identification technique in Section 7.4.1 is used to estimate G described by (7.96), (7.97). The estimate of G is then used to construct the numerator $N_{\rm f}$ of the target model based on (7.54). To enhance the accuracy of the identification, an external perturbation signal $\delta_{\rm pert}$ is added to the controller output.

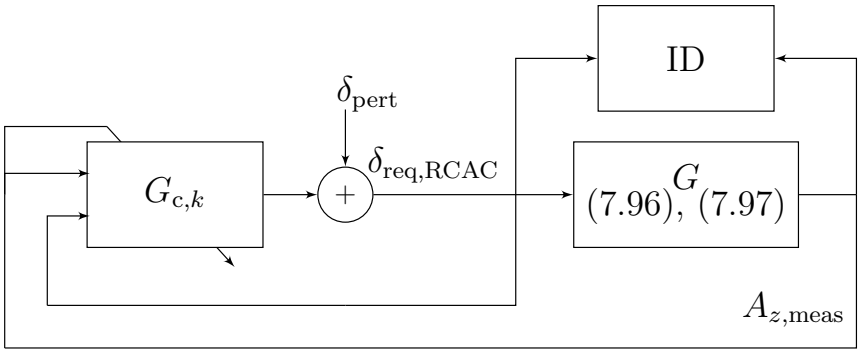


Figure 7.7: RCAC combined with the linear missile model in the absence of commands. The goal is to use RCAC to place closed-loop poles that match those arising from the 3LA. The same setup is used to demonstrate the use of direct identification of G to construct the numerator $N_{\rm f}$ of the target model $G_{\rm f}$.

For this example, the adaptive controller has order $n_{\rm c} = 5$ and is written as

$$\delta_{\rm req,RCAC}(k) = \frac{Q_1(k)\mathbf{q^4} + \cdots + Q_5(k)}{\mathbf{q}^5 - P_1(k)\mathbf{q}^4 - \cdots - P_5(k)} A_{z,\rm meas}(k). \tag{7.100}$$

The model is linearized at the flight condition $M_0 = 2.0$, $\alpha_0 = 0.0$ rad, and $q_0 = 0.0$ rad/s. For this flight condition, the gains for the 3LA are $K_{\rm a} = 0.945$, $K = 0.027$, $K_{\rm I} = 22.556$, and $K_{\rm g} = 2.125$. Equations (7.98) and (7.99) are discretized at 1000 Hz using zero-order hold, and the resulting discrete-time equations are used to construct the denominator

$$D_{\rm f}(\mathbf{q}) = \mathbf{q}^5 - 4.551\mathbf{q}^4 + 8.296\mathbf{q}^3 - 7.574\mathbf{q}^2 + 3.464\mathbf{q} - 0.635, \tag{7.101}$$

of the target model, whose order is $n_{\rm f} = 5$. The roots of $D_{\rm f}$, which are the target

closed-loop poles, are $[0.856 \;\; 0.865 \pm 0.142\jmath \;\; 0.982 \pm 0.021\jmath]$.

In order to evaluate the accuracy of the identification, G which is represented by (7.96) and (7.97), is discretized at a sampling rate of 1000 Hz using the zero-order hold operation which yields

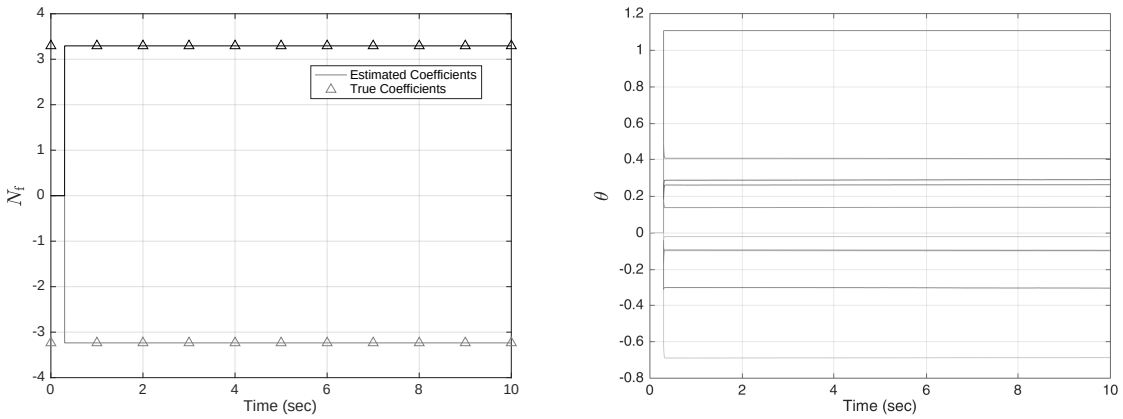
$$A_{z,\text{meas}}(k) = \frac{-3.235\mathbf{q}^3 + 3.639\mathbf{q}^2 + 2.357\mathbf{q} - 2.757}{\mathbf{q}^4 - 3.551\mathbf{q}^3 + 4.739\mathbf{q}^2 - 2.823\mathbf{q} + 0.635}\delta_{\text{req}}(k), \tag{7.102}$$

with poles located at $[0.993 \pm 6.326 \times 10^{-4}\jmath \;\; 0.782 \pm 0.178\jmath]$ and zeros located at $[-0.863 \;\; 0.969 \;\; 1.018]$. Note that there is one NMP zero located at 1.018. Thus, accurate identification of G and proper extraction of the leading numerator coefficient and NMP zero location yields

$$N_{\text{f}}(\mathbf{q}) = -3.235\mathbf{q}^4 + 3.293\mathbf{q}^3. \tag{7.103}$$

In Figure 7.7, the external perturbation signal δ_{pert} is zero-mean white noise with standard deviation 1.0×10^{-7} rad. To identify G, a 6th-order IIR model with a sliding window least squares algorithm with window size $p_{\text{c}} = 300$ time steps is used, as discussed in Section 7.4.1. Note that this window size equates to a data buffer of 0.3 s. The leading coefficient and all NMP zeros are extracted from the numerator of the identified model and are used to update the numerator N_{f} of the target model G_{f} according to (7.54). Figure 7.8a shows the estimated and true coefficients for the target model numerator N_{f}. At time 0.3 s, the data buffer is full and the estimate of N_{f} is provided for the RCAC target model. RCAC is also run with the same sliding window of size $p_{\text{c}} = 300$ with the learning-rate weight $R_\delta = 1.0 \times 10^{-17} I_{10}$. Figure 7.8b shows the evolution of the coefficients of the adaptive controller, where, at time 0.3 s, the buffer is full and RCAC is controlling the linearized missile dynamics. Figure 7.9 compares the desired closed-loop pole locations with the closed-loop pole locations obtained by RCAC at the final simulation time of 10.0 s. Note that the adapted

closed-loop poles are located near the desired closed-loop pole locations, with the additional closed-loop poles located near the origin.



(a) Coefficients of N_f. (b) RCAC controller coefficients $\theta(k)$.

Figure 7.8: Evolution of the estimated polynomial coefficients used in the target model numerator N_f and the adaptive controller coefficients $\theta(k)$ for the linear missile example.

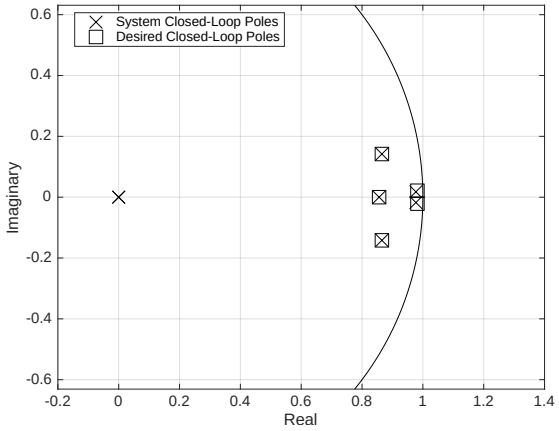


Figure 7.9: Comparison of the desired and actual closed-loop pole locations. The closed-loop system is evaluated using the controller coefficients at the final time 10.0 s.

7.5.3 Simultaneous Identification and Adaptive Control of the Nonlinear Missile

In order to construct the semi-adaptive autopilot, 3LA is combined with RCAC by applying the augmented control law

$$\delta_\mathrm{req}(k) = \delta_\mathrm{req,3LA}(k) + \delta_\mathrm{req,RCAC}(k) + \delta_\mathrm{pert}(k), \tag{7.104}$$

where $\delta_{\text{req,3LA}}$ is the gain-scheduled 3LA contribution, $\delta_{\text{req,RCAC}}$ is the adaptive contribution, and δ_{pert} is an external perturbation signal for enhancing persistency. This augmentation is designed to improve the performance of the closed-loop system when the performance of the 3LA is degraded by modeling errors. The adaptive controller is provided with the reference normal acceleration command $A_{z,\text{ref}}$ through the scheduled transfer function G_{ref}. Note that, although G_{ref} is represented by (7.98) and (7.99), it is updated at each time step as a function of α and M. The scheduled G_{ref} also updates the denominator D_{f} of the target model with the desired closed-loop polynomial. The numerator N_{f} of the target model G_{f} is updated through LS direct identification with a sliding window of size $p_{\text{c}} = 300$ and a 10th-order IIR model, as discussed in Section 7.4.1. Figure 7.10 shows how RCAC augments the 3LA. The sensor noise w is zero-mean white noise with standard deviation of 1.0×10^{-6} for both the acceleration $A_{z,\text{meas}}$ and the pitch-rate q_{meas}.

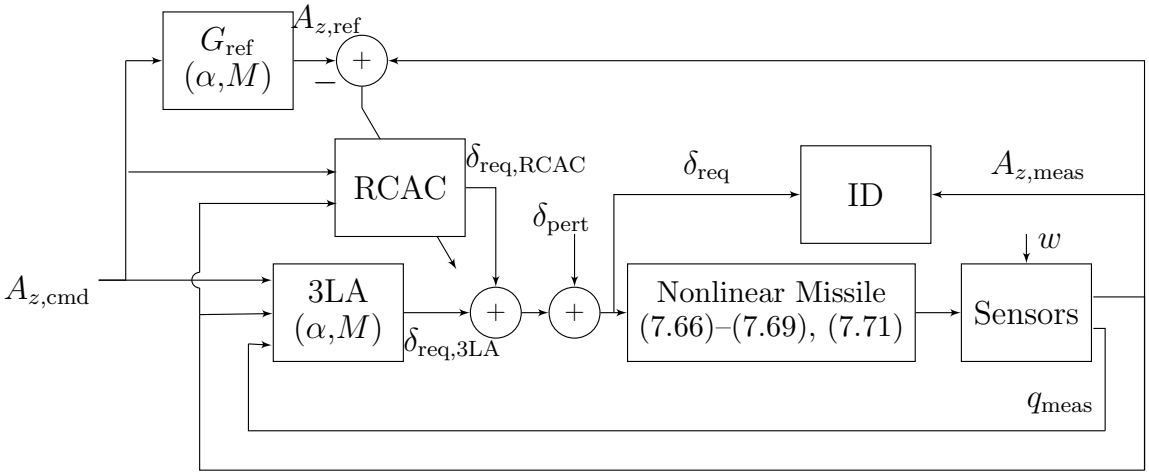


Figure 7.10: Block diagram of the semi-adaptive control law for the nonlinear missile model consisting of the 3LA augmented with RCAC.

For the following examples, the multi-input, single-output adaptive controller is

finite impulse response (FIR), of order $n_c = 5$, and written as

$$\delta_{\text{req,RCAC}}(k) = \sum_{i=1}^{n_c} P_{1,i}(k) A_{z,\text{des}}(k-i) + \sum_{i=1}^{n_c} P_{2,i}(k) A_{z,\text{meas}}(k-i), \tag{7.105}$$

The performance variable is $z(k) = A_{z,\text{meas}}(k) - A_{z,\text{ref}}(k)$. In terms of the retrospective performance variable (7.22), $u(k) = \delta_{\text{req}}(k) - \delta_{\text{req,3LA}}(k)$, so that the RCAC optimizes based only on the control contribution of the adaptive control effort. The retrospective performance variable is optimized over sliding window of $p_c = 300$ with a learning-rate weight $R_\delta = 1.0 \times 10^4 I_{10}$.

The examples given below compare the performance of the 3LA and the semi-adaptive autopilot. The first example assumes no model error, while the second example considers uncertainty in the pitch moment coefficient and control authority.

Example 7.1 (Nominal Performance)**.** In this example, we compare the 3LA with the semi-adaptive autopilot under no uncertainty. The model is initialized at $M(0) = 3$, $\alpha(0) = 0$, $q(0) = 0$, and $\theta(0) = 0$. The external perturbation signal δ_{pert} is zero-mean white noise with standard deviation 1.0×10^{-5} rad. Figure 7.11 compares the desired and measured normal acceleration of the 3LA and the semi-adaptive autopilot. Note that the 3LA contributes the majority of the control signal. Since G_{ref} is a linearized gain scheduled transfer function and the missile dynamics are nonlinear, small errors arise between the reference normal acceleration command $A_{z,\text{ref}}$ and the measured normal acceleration $A_{z,\text{meas}}$, which leads to a small adaptive controller contribution, as shown by the adaptive controller coefficients in Figure 7.12b.

Figure 7.12a compares the target model numerator coefficients based on linearization of the nominal model at each time step versus on-line identification. Note that, at time 0.3 s, the on-line identification starts and converges to the nominal linearized coefficients. ■

Example 7.2 (Performance Under Uncertainty.)**.** We now compare the 3LA with the

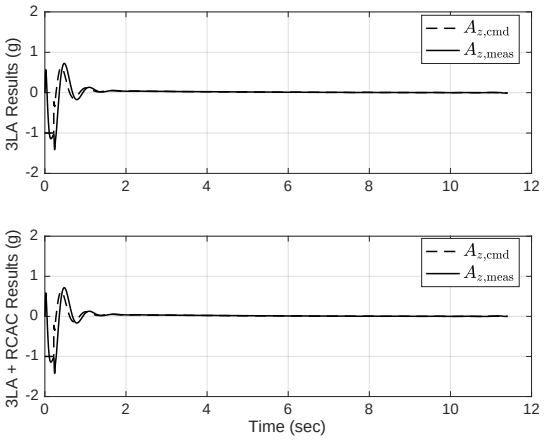
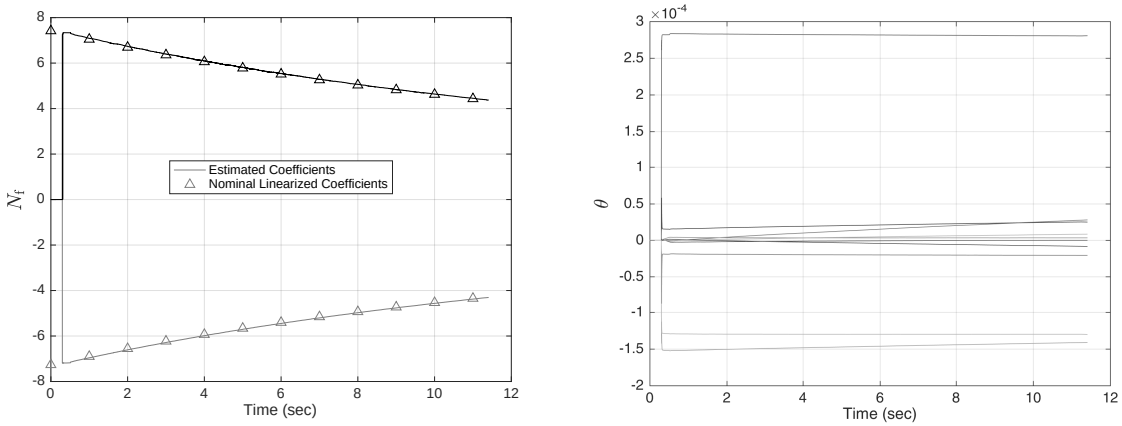


Figure 7.11: Comparison of the 3LA and the semi-adaptive autopilot for the nominal nonlinear model for Example 7.1.



(a) Coefficients of N_{f}.

(b) RCAC controller coefficients $\theta(k)$.

Figure 7.12: Evolution of the estimated polynomial coefficients used in the target model numerator N_{f} and the adaptive controller coefficients $\theta(k)$ for Example 7.1.

semi-adaptive autopilot in the case where the missile aerodynamics are uncertain. In particular, we consider the model errors

$$C_{m_q} = 2.0e_m, \qquad C_{z_{\delta_c}} = 0.9d_z. \tag{7.106}$$

The model is initialized at $M(0) = 3$, $\alpha(0) = 0$, $q(0) = 5$ rad/s, and $\theta(0) = 0$. The external perturbation signal δ_{pert} is zero-mean white noise with standard deviation 1.0×10^{-5} rad. Figure 7.13 compares the desired and measured normal acceleration of the 3LA with the response of the semi-adaptive autopilot. Note that, due to the

uncertainty in the nonlinear missile model, the 3LA exhibits oscillatory behavior at the beginning of the flight and as well as at the terminal phase. For the semi-adaptive autopilot, the acceleration is initially oscillatory but damps out as the flight progresses due to the adaptive contribution to the control effort and is not as aggressive during the terminal phase. Note that the adaptive controller coefficients shown in Figure 7.14b change rapidly at the beginning of the flight and converge as the simulation progresses.

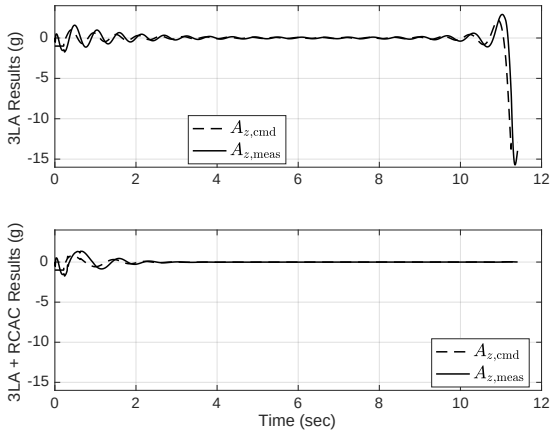
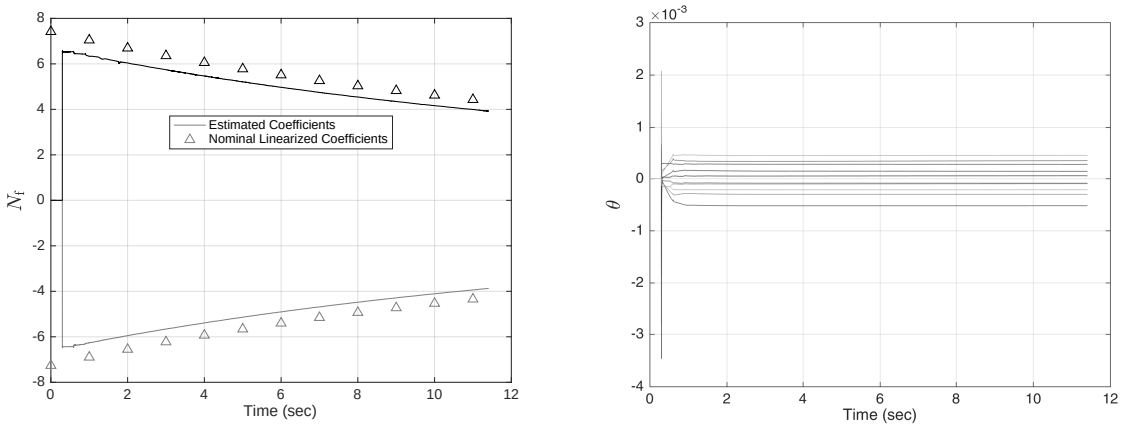


Figure 7.13: Comparison of the 3LA with the semi-adaptive autopilot for Example 7.2, which assumes uncertainty in the nonlinear model.

Figure 7.14a compares the target model numerator coefficients based on linearization of the nominal model at each time step versus the on-line identification. Note that at time 0.3 s, the on-line identification starts and converges to values that differ from the nominal linearized coefficients due to the control effectiveness $C_{z_{\delta_c}}$ differing from the nominal model. ■

7.6 Conclusions

The contents within this chapter represents a first step in demonstrating that adaptive control can assist a gain-scheduled autopilot by compensating for uncertainty in the underlying model. To achieve this objective, RCAC was combined in a



(a) Coefficients of N_f.

(b) RCAC controller coefficients $\theta(k)$.

Figure 7.14: Evolution of the estimated polynomial coefficients used in the target model numerator N_f and the adaptive controller coefficients $\theta(k)$ for Example 7.2.

semi-adaptive control architecture with a baseline gain-scheduled three-loop autopilot for a planar three-degree-of-freedom missile model. To facilitate the use of RCAC without depending on the aerodynamic model, concurrent identification was used to estimate the leading numerator coefficient and nonminimum-phase (NMP) zero. Since the model used for identification is linear, this approach is arguably incompatible with the nonlinear missile dynamics. Nevertheless, the concurrent identification was able to estimate the NMP zeros of the instantaneously linearized missile dynamics under off-nominal conditions, and RCAC was able to use this data to the extent that cancellation of NMP zeros was avoided. Future work will stress this technique to a much greater extent by flying the missile under higher Mach numbers and angles of attack. The ultimate test of this approach is to apply it to a fully nonlinear 6DOF missile model and, ultimately, flight tests.

CHAPTER 8

Conclusions

This dissertation presented advances in the retrospective cost adaptive control (RCAC) algorithm. In chapter 2, the basic RCAC algorithm was presented. The adaptive controller can be used for stabilization, command following, and disturbance rejection for multi-input, multi-output, linear, time -invariant, minimum phase, nonminimum-phase, discrete-time systems. The require minimal modeling information needed includes the plant transfer function leading numerator coefficient and any nonminimum-phase (NMP) zeros. We applied RCAC to the NASA GTM model under unanticipated and unknown changes to the aerodynamics of the aircraft. Specifically, we used a single RCAC block within the simulation that controls the aircraft to a desired steady level flight by commanding five actuation channels (left and right engines, aileron, elevator, and rudder). The goal is to examine the evolution of the RCAC controller gains in response to changes in the aerodynamic coefficients from the linearized aircraft dynamics. To show this, simulations in the Simulink GTM were run, all with the desire for the aircraft to fly in a straight and level flight configuration. At a certain time (unknown to RCAC) the aerodynamic coefficients changed. Presented were three different aerodynamic parameter modifications, including an icing example, a sudden increase in drag, and a sudden decrease in lift. For all three examples, only a single tuning was used for each RCAC block.

Results in chapter 2 showed that RCAC is able to adapt its gains in order to compensate for the unknown time-varying aerodynamic perturbations while maintaining the desired performance. The icing example showed that both the elevator and engines were used to compensate for an increase of 700% drag and decrease of 30% in lift production over a 500 sec interval. In the drag example, RCAC overcame a increase in the aerodynamic drag coefficient by over 200% by increasing the engine thrust and using minimal elevator deflections. The gains evolved substantially to compensate for the amount of drag induced. Finally, we considered an example where the lift decreased by 40%. Given the constraints imposed by the performance variable on the airspeed, RCAC used the elevator to increase the angle of attack to maintain altitude. Future research focus on additional aerodynamic stability derivative changes as well as a method for relaxing constraints to maintain safe level flight.

In chapter 3, we extended [12, 13] in the RCAC formulation by including a variable forgetting factor and Kalman Filter. This extension allowed for continual parameter estimation in the adaptive controller update, a feature that is essential to controlling a system with nonlinear dynamics. Additionally, an inner-loop/outer-loop control architecture is used to adaptively adjust the pitch-rate command based on the normal acceleration command. The adaptive pitch-rate command loop is appropriate due to the inability to infer such a command when given an arbitrary normal acceleration command. Results show that the CFF, although effective, leads to aggressive adaptation that may lead to instabilities. We also showed that both RCAC/VFF and RCAC/KF allow for gain adapting through the entire flight and were able to track the normal acceleration command as well as the 3LA. The adaptive controller excelled when an aerodynamic coefficient modification was introduced to the system as well as having a lower bandwidth actuator.

Chapter 4 proposed a technique that minimizes the innovations based on retrospective optimization of the process noise covariance. We showed that minimizing the

cumulative innovations is equivalent to minimizing the cumulative state-estimation error for the parameter estimation problem under certain assumptions. This technique is applied to system identification problems where the parameters to be estimated can be time-varying and thus have an unknown Q value. We compared our technique to the standard Kalman filter and IAKF. Results show that when applied to time-varying parameter estimation problem, this technique performs as well if not better than IAKF.

The cost function associated with RCAC was used for concurrent optimization of the controller gains and the target model due to its natural biquadratic structure in chapter 5. A modified version of the alternating convex search (ACS) and the MATLAB fminsearch algorithm demonstrated the ability to concurrently optimize these coefficients for a command-following problem. The novel element in this chapter is the ability to minimize the cost function without prior knowledge of the plant transfer function, including NMP zero locations. Future work will focus on improving the computational efficiency of the concurrent optimization along with guarantees of global convergence.

The objective of chapter 6 was to numerically investigate the effectiveness of architectures for closed-loop identification. Three direct closed-loop identification architectures and three indirect closed-loop identification architectures were considered. These architectures included standard cases with and without auxiliary signals as well as two novel architectures involving intercalated injection of the auxiliary signal. Infinite impulse response models were fit using least squares (LS) estimation, which provided a baseline method, and prediction error methods (PEM), which account for noise correlation. To simplify the study, the plant order was assumed to be known; for the indirect architectures, the auxiliary signal was chosen to be a multiple of the command; and errors-in-variables noise was not considered. The signal-to-noise ratio was normalized across all architectures. Motivated by adaptive control, the perfor-

mance metric was chosen to be the accuracy of the estimate of the nonminimum-phase (NMP) zero of the plant. Two examples were considered, both of which were third-order and NMP. One plant was asymptotically stable, and the other was unstable.

As expected, for all of the architectures and for both examples, the least squares estimates exhibited bias, whereas the PEM estimates indicated consistency. In comparing architectures, the numerical results do not allow definitive conclusions, but some observations can be made. Based on PEM for direct closed-loop identification, standard injection of the auxiliary signal appears to be advantageous; direct closed-loop and auxiliary direct closed-loop with intercalated injection provide roughly the same accuracy. For indirect closed-loop identification, the conclusions are less clear. For the asymptotically stable plant, standard injection of the auxiliary signal appears to be advantageous, whereas, for the unstable plant, intercalated injection appears advantageous.

In view of the practical importance of closed-loop identification, further investigation is warranted. A more detailed study would include consideration of instrumental variables as an alternative to PEM, uncertainty in the plant order, EIV noise, and higher order plants with multiple NMP zeros.

The contents within chapter 7 represents a first step in demonstrating that adaptive control can assist a gain-scheduled autopilot by compensating for uncertainty in the underlying model. To achieve this objective, RCAC was combined in a semi-adaptive control architecture with a baseline gain-scheduled three-loop autopilot for a planar three-degree-of-freedom missile model. To facilitate the use of RCAC without depending on the aerodynamic model, concurrent identification was used to estimate the leading numerator coefficient and nonminimum-phase (NMP) zero. Since the model used for identification is linear, this approach is arguably incompatible with the nonlinear missile dynamics. Nevertheless, the concurrent identification was able to estimate the NMP zeros of the instantaneously linearized missile dynamics under

off-nominal conditions, and RCAC was able to use this data to the extent that cancellation of NMP zeros was avoided. Future work will stress this technique to a much greater extent by flying the missile under higher Mach numbers and angles of attack. The ultimate test of this approach is to apply it to a fully nonlinear 6DOF missile model and, ultimately, flight tests.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] K. J. Åström and B. Wittenmark, *Adaptive Control*. Courier Corporation, 2013.

[2] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Courier Corporation, 2012.

[3] M. Krstic, P. V. Kokotovic, and I. Kanellakopoulos, *Nonlinear and Adaptive Control Design*. John Wiley & Sons, Inc., 1995.

[4] I. Landau, "From Robust Control to Adaptive Control," *Contr. Eng. Prac.*, vol. 7, no. 9, pp. 1113–1124, 1999.

[5] E. Lavretsky and K. A. Wise, "Robust Adaptive Control," in *Robust and Adaptive Control*. Springer, 2013, pp. 317–353.

[6] R. Venugopal and D. S. Bernstein, "Adaptive Disturbance Rejection Using ARMARKOV System Representations," in *Proc. 36th IEEE Conf. Dec. Contr.,*, vol. 2, Dec 1997, pp. 1884–1889.

[7] J. B. Hoagg, M. A. Santillo, and D. S. Bernstein, "Discrete-Time Adaptive Command Following and Disturbance Rejection with Unknown Exogenous Dynamics," *IEEE Trans. Autom. Contr.*, vol. 53, no. 4, pp. 912–928, 2008.

[8] J. B. Hoagg and D. S. Bernstein, "Retrospective Cost Model Reference Adaptive Control for Nonminimum-Phase Systems," *AIAA J. Guid. Contr. Dyn.*, vol. 35, no. 6, pp. 1767–1786, 2012.

[9] A. M. D'Amato, E. D. Sumer, K. Mitchell, A. Morozov, J. B. Hoagg, and D. S. Bernstein, "Adaptive Output Feedback Control of the NASA GTM Model with Unknown Nonminimum-Phase zeros," in *AIAA Guid. Nav. Contr. Conf.*, Portland, OR, Aug 2011.

[10] M.-J. Yu, Y. Rahman, E. M. Atkins, I. Kolmanovsky, and D. Bernstein, "Minimal Modeling Adaptive Control of the NASA Generic Transport Model with Unknown Control-Surface Faults," in *AIAA Guid. Nav. Contr. Conf.*, 2013,AIAA-2013-4693.

[11] A. Ansari and D. S. Bernstein, "Retrospective Cost Adaptive Control of Generic Transport Model under Uncertainty and Failure," *J. of Aero. Inform. Sys.*, vol. 14, pp. 123–174, 2017.

[12] A. D'Amato and D. Bernstein, "Adaptive Control of a Seeker-Guided 2D Missile with Unmodeled Aerodynamics," in *AIAA Gui. Nav. Contr. Conf.*, Minneapolis, MN, August 2012, AIAA-2012-4617-516.

[13] R. J. Fuentes, J. B. Hoagg, B. J. Anderton, A. M. D'Amato, and D. S. Bernstein, "Investigation of Cumulative Retrospective Cost Adaptive Control for Missile Application," in *AIAA Guid. Nav. Contr. Conf.*, Toronto, August 2010, AIAA 2010-7577.

[14] C. Hide, T. Moore, and M. Smith, "Adaptive Kalman Filtering for Low-Cost INS/GPS," *J. of Nav.*, vol. 56, no. 01, pp. 143–152, 2003.

[15] A. Almagbile, J. Wang, and W. Ding, "Evaluating the Performances of Adaptive Kalman Filter Methods in GPS/INS Integration," *J. of G.P.S.*, vol. 9, no. 1, pp. 33–40, 2010.

[16] A. Mohamed and K. Schwarz, "Adaptive Kalman Filtering for INS/GPS," *J. of Geo.*, vol. 73, no. 4, pp. 193–203, 1999.

[17] R. K. Mehra, "Approaches to Adaptive Filtering," *IEEE Trans. Autom. Contr.*, vol. 17, no. 5, pp. 693–698, 1972.

[18] L. Guo, "Estimating Time-Varying Parameters by the Kalman Filter Based Algorithm: Stability and Convergence," *IEEE Trans. on Autom. Contr.*, vol. 35, no. 2, pp. 141–147, 1990.

[19] M. A. Zagrobelny and J. B. Rawlings, "Identification of Disturbance Covariances Using Maximum Likelihood Estimation," No. 2014-02, Tech. Rep., 2014.

[20] A. Ilchmann, *Non-Identifier-Based High-Gain Adaptive Control*. Springer-Verlag New York, Inc., 1993.

[21] K. J. Åström, "Direct Methods for Nonminimum Phase Systems," in *Proc. Conf. Dec. Contr., Albuquerque, NM*, no. 19, 1980, pp. 611–615.

[22] G. C. Goodwin and K. Sin, "Adaptive Control of Nonminimum Phase Systems," *IEEE Trans. Autom. Contr.*, vol. 26, no. 2, pp. 478–483, 1981.

[23] Y. Rahman, K. F. Aljanaideh, E. D. Sumer, and D. S. Bernstein, "Adaptive Control of Aircraft Lateral Motion with an Unknown Transition to Nonmimimum-Phase Dynamics," in *Amer. Contr. Conf.* IEEE, 2014, pp. 2359–2364.

[24] M. A. Santillo and D. S. Bernstein, "Adaptive Control Based on Retrospective Cost Optimization," *AIAA J. Guid. Contr. Dyn.*, vol. 33, no. 2, pp. 289–304, 2010.

[25] J. Gorski, F. Pfeuffer, and K. Klamroth, "Biconvex Sets and Optimization with Biconvex Functions: A Survey and Extensions," *Mathematical Methods of Operations Research*, vol. 66, no. 3, pp. 373–407, 2007.

[26] C. Ling, J. Nie, L. Qi, and Y. Ye, "Biquadratic Optimization Over Unit Spheres and Semidefinite Programming Relaxations," *SIAM J. Opt.*, vol. 20, no. 3, pp. 1286–1310, 2009.

[27] J. G. VanAntwerp and R. D. Braatz, "A Tutorial on Linear and Bilinear Matrix Inequalities," *J. Proc. Contr.*, vol. 10, no. 4, pp. 363–385, 2000.

[28] H. D. Tuan and P. Apkarian, "Low Nonconvexity-Rank Bilinear Matrix Inequalities: Algorithms and Applications in Robust Controller and Structure Designs," *IEEE Trans. Autom. Contr.*, vol. 45, no. 11, pp. 2111–2117, 2000.

[29] K.-C. Goh, M. Safonov, and J. Ly, "Robust Synthesis via Bilinear Matrix Inequalities," *Int. J. Robust Nonlin. Contr.*, vol. 6, no. 9-10, pp. 1079–1095, 1996.

[30] U. Forssell and L. Ljung, "Closed-Loop Identification Revisited," *Automatica*, vol. 35, pp. 1215–1241, 1999.

[31] J. B. Hoagg and D. S. Bernstein, "Nonminimum-Phase Zeros: Much to Do about Nothing," *IEEE Contr. Sys. Mag.*, vol. 27, pp. 45–57, June 2007.

[32] M. A. Santillo and D. S. Bernstein, "Adaptive Control Based on Retrospective Cost Optimization," *AIAA J. Guid. Contr. Dyn.*, vol. 33, pp. 289–304, 2010.

[33] J. B. Hoagg and D. S. Bernstein, "Retrospective Cost Model Reference Adaptive Control for Nonminimum-Phase Systems," *AIAA J. Guid. Contr. Dyn.*, vol. 35, pp. 1767–1786, 2012.

[34] Y. Rahman, A. Xie, J. B. Hoagg, and D. S. Bernstein, "A Tutorial and Overview of Retrospective-Cost-Based Adaptive Control," in *Proc. Amer. Contr. Conf.*, Boston, MA, July 2016, pp. 3386–3409.

[35] Y. Rahman and D. S. Bernstein, "Adaptive Control of Plants That Are Practically Impossible to Control by Fixed-Gain Control Laws," in *Proc. Conf. Dec. Contr.*, Las Vegas, NV, December 2016, pp. 383–388.

[36] Y. Rahman, A. Xie, and D. S. Bernstein, "Retrospective Cost Adaptive Control: Pole Placement, Frequency Response, and Connections with LQG Control," *IEEE Contr. Sys. Mag.*, 2017, to appear.

[37] F. M. Sobolic, G. Cruz, and D. S. Bernstein, "An Inner-Loop/Outer-Loop Architecture for an Adaptive Missile Autopilot," in *Proc. Amer. Contr. Conf.*, Chicago, IL., July 2015, pp. 850–855.

[38] J. Gertler, *Fault Detection and Diagnosis in Engineering Systems*. Marcel Dekker, 1998.

[39] D. Dasgupta, K. Krishna-Kumar, D. Wong, and M. Berry, "Negative Selection Algorithm for Aircraft Fault Detection," in *Artificial Immune Systems*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, vol. 3239, pp. 1–13.

[40] R. Isermann, *Fault-Diagnosis Systems*. Springer, 2006.

[41] M. Blanke, *Diagnosis and Fault-Tolerant Control*, ser. Engineering Online Library. Springer, 2003.

[42] J. Roskam, *Airplane Flight Dynamics and Automatic Flight Controls*, ser. Airplane Flight Dynamics and Automatic Flight Controls. Darcorporation, 2003, no. pt. 2.

[43] P. Ioannou and B. Fidan, "Adaptive Control Tutorial. SIAM Society for Industrial & Applied Mathematics,," ISBN 0-89871-615-2, Tech. Rep., 2006.

[44] T. Fortescue, L. Kershenbaum, and B. Ydstie, "Implementation of Self-Tuning Regulators with Variable Forgetting Factors," *Automatica*, vol. 17, no. 6, pp. 831 – 835, 1981.

[45] A. Cordero and D. Mayne, "Deterministic Convergence of a Self-Tuning Regulator with Variable Forgetting Factor," *Control Theory and Applications, IEE Proceedings D*, vol. 128, no. 1, pp. 19–23, January 1981.

[46] C. Schumacher and P. P. Khargonekar, "Missile Autopilot Designs Using H_∞ Control with Gain Scheduling and Dynamic Inversion," *J. Guid. Contr. Dyn.*, vol. 21, no. 2, pp. 234–243, 1998.

[47] H. Buschek, "Full Envelope Missile Autopilot Design Using Gain Scheduled Robust Control," *AIAA J. Guid. Contr. Dyn.*, vol. 22, no. 1, pp. 115–122, 1999.

[48] C. Mracek and J. Cloutier, "Full Envelope Missile Longitudinal Autopilot Design Using the State-Dependent Riccati Equation Method," in *AIAA Guid. Nav. Contr. Conf.*, August 1997, AIAA-97-3767, pp. 1697–1705.

[49] D. Lawrence, J. Kelly, and J. Evers, "Gain Scheduled Missile Autopilot Design Using a Control Signal Interpolation Technique," in *Guid. Nav. Contr. Conf.*, 1998, p. 4418.

[50] M. Xin and S. N. Balakrishnan, "Missile Autopilot Design Using a New Suboptimal Nonlinear Control Method," in *IEEE Proc. of 41st Aerospace Sciences Meeting and Exhibit*, 2003, pp. 577–584.

[51] E. Sumer, M. Holzel, A. D'Amato, and D. Bernstein, ""FIR-Based Phase Matching for Robust Retrospective-Cost Adaptive Control,"," in *Proc. Amer. Contr. Conf.*, June 2012, Montreal, Canada, pp. 2707–2712.

[52] R. G. Brown and P. Y. Hwang, *Introduction to Random Signals and Applied Kalman Filtering: with MATLAB Exercises and Solutions*, 3rd ed. New York: John Wiley and Sons, 1997.

[53] P. Axelsson, U. Orguner, F. Gustafsson, and M. Norrlöf, "ML Estimation of Process Noise Variance in Dynamic Systems," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 5609–5614, 2011.

[54] MATLAB, *version 7.9.0.529 (R2009b)*. Natick, Massachusetts: The MathWorks Inc., 2009.

[55] Y. Zhu and F. Butoyi, "Case Studies on Closed-Loop Identification for MPC," *Contr. Eng. Prac.*, vol. 10, no. 4, pp. 403–417, 2002.

[56] J. Langer and I. Landau, "Improvement of Robust Digital Control by Identification in the Closed Loop. Application to a 360 Flexible Arm," *Contr. Eng. Prac.*, vol. 4, no. 12, pp. 1637–1646, 1996.

[57] L. von Wangenheim, "Phase Margin Determination in a Closed-Loop Configuration," *Circ. Sys. Sig. Proc.*, vol. 31, no. 6, pp. 1917–1926, 2012.

[58] I. Gustavsson, L. Ljung, and T. Soderstrom, "Identification of Process in Closed Loop – Identifiability and Accuracy Aspects," *Automatica*, vol. 13, pp. 59–75, 1977.

[59] H. Hjalmarsson, M. Gevers, and F. De Bruyne, "For Model-Based Control Design, Closed-Loop Identification Gives Better Performance," *Automatica*, vol. 32, pp. 1659–1673, 1996.

[60] P. Van den Hof, "Closed-Loop Issues in System Identification," *Annual Reviews in Control*, vol. 22, pp. 173–186, 1998.

[61] H. Hjalmarsson, "From Experiment Design to Closed-Loop Control," *Automatica*, vol. 41, no. 3, pp. 393–438, 2005.

[62] L. Ljung, "Perspectives on System Identification," *Annual Reviews in Control*, vol. 34, no. 1, pp. 1–12, 2010.

[63] Z. Yakoub, M. Amairi, M. Chetoui, and M. Aoun, "On the Closed-Loop System Identification with Fractional Models," *Circ. Sys. Sig. Proc.*, vol. 34, no. 12, pp. 3833–3860, 2015.

[64] B. Shen, F. Ding, A. Alsaedi, and T. Hayat, "Gradient-Based Recursive Identification Methods for Input Nonlinear Equation Error Closed-Loop Systems," *Circ. Sys. Sig. Proc.*, vol. 36, no. 5, pp. 2166–2183, 2017.

[65] J. P. Epperlein, B. Bamieh, and K. J. Åström, "Thermoacoustics and the Rijke Tube: Experiments, Identification, and Modeling," *IEEE Contr. Sys. Mag.*, vol. 35, no. 2, pp. 57–77, 2015.

[66] D. Engelhart, T. Boonstra, R. Aarts, A. Schouten, and H. van der Kooij, "Comparison of Closed-Loop System Identification Techniques to Quantify Multi-Joint Human Balance Control," *Annual Reviews in Control*, vol. 41, pp. 58–70, 2016.

[67] E. De Vlugt, A. C. Schouten, and F. C. Van Der Helm, "Closed-Loop Multivariable System Identification for the Characterization of the Dynamic Arm Compliance Using Continuous Force Disturbances: A Model Study," *Journal of Neuroscience Methods*, vol. 122, no. 2, pp. 123–140, 2003.

[68] G. Van Baars and P. Bongers, "Closed Loop System Identification of an Industrial Wind Turbine System: Experiment Design and First Validation Results," in *Proc. IEEE Con. Dec. Contr.*, vol. 1. IEEE, 1994, pp. 625–630.

[69] R. S. Smith, "Closed-Loop Identification of Flexible Structures: An Experimental Example," *J. Guid. Contr. Dyn.*, vol. 21, no. 3, pp. 435–440, 1998.

[70] M. Östring, S. Gunnarsson, and M. Norrlöf, "Closed-Loop Identification of an Industrial Robot Containing Flexibilities," *Contr. Eng. Prac.*, vol. 11, no. 3, pp. 291–300, 2003.

[71] P. Van den Hof, "Closed-Loop Issues in System Identification," *Annual Reviews in Control*, vol. 22, pp. 173–186, 1998.

[72] D. E. Rivera, H. Lee, H. D. Mittelmann, and M. W. Braun, "High-Purity Distillation," *IEEE Contr. Sys. Mag.*, vol. 27, pp. 72–89, October 2007.

[73] B. Ho and R. E. Kalman, "Effective Construction of Linear State-Variable Models from Input/Output Functions," *Regelungstechnik*, vol. 14, no. 1-12, pp. 545–548, 1966.

[74] J. N. Juang and R. S. Pappa, "An Eigensystem Realization Algorithm for Modal Parameter Identififcation and Model Reduction," *J. Guid. Contr. Dyn.*, vol. 8, no. 5, pp. 620–627, 1985.

[75] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization," *SIAM Review*, vol. 52, no. 3, pp. 471–501, 2010.

[76] R. S. Smith, "Frequency Domain Subspace Identification Using Nuclear Norm Minimization and Hankel Matrix Realizations," *IEEE Trans. Autom. Contr.*, vol. 59, no. 11, pp. 2886–2896, 2014.

[77] K. F. Aljanaideh and D. S. Bernstein, "Closed-Loop Identification of Unstable Systems Using Noncausal FIR Models," *Int. J. Contr.*, vol. 90, no. 2, pp. 168–185, 2017.

[78] M. Galrinho, "Least Squares Methods for System Identification of Structured Models," Ph.D. dissertation, KTH Royal Institute of Technology, 2016.

[79] L. Ljung and B. Wahlberg, "Asymptotic Properties of the Least-Squares Method for Estimating Transfer Functions and Disturbance Spectra," *Advances in Applied Probability*, vol. 24, no. 02, pp. 412–440, 1992.

[80] L. Ljung, "On the Consistency of Prediction Error Identification Methods," *Mathematics in Science and Engineering*, vol. 126, pp. 121–164, 1976.

[81] P. Van den Hof and R. J. Schrama, "An Indirect Method for Transfer Function Estimation from Closed Loop Data," *Automatica*, vol. 29, no. 6, pp. 1523–1527, 1993.

[82] L. Ljung and U. Forssell, "An Alternative Motivation for the Indirect Approach to Closed-Loop Identification," *IEEE Trans. Autom. Contr.*, vol. 44, no. 11, pp. 2206–2209, 1999.

[83] T. Söderström and P. Stoica, *System Identification*. Prentice-Hall, Inc., 1988.

[84] L. Ljung, *System Identification: Theory for the User*. Prentice Hall PTR, USA, 1999.

[85] P. Joseph, J. Lewis, and J. Tou, "Plant Identification in the Presence of Disturbances and Application to Digital Adaptive Systems," *Trans. Amer. Inst. Elec. Eng., Part II: Applications and Industry*, vol. 80, no. 1, pp. 18–24, 1961.

[86] K. Wong and E. Polak, "Identification of Linear Discrete Time Systems Using the Instrumental Variable Method," *IEEE Trans. Autom. Contr.*, vol. 12, no. 6, pp. 707–718, 1967.

[87] T. Söderström and P. Stoica, "Instrumental Variable Methods for System Identification," *Cir. Sys. Sig. Proc.*, vol. 21, no. 1, pp. 1–9, 2002.

[88] P. Young, A. Jakeman, and R. McMurtrie, "An Instrumental Variable Method for Model Order Identification," *Automatica*, vol. 16, no. 3, pp. 281–294, 1980.

[89] T. Söderström and K. Mahata, "On Instrumental Variable and Total Least Squares Approaches for Identification of Noisy Systems," *Int. J. Contr.*, vol. 75, no. 6, pp. 381–389, 2002.

[90] M. Gilson and P. Van Den Hof, "Instrumental Variable Methods for Closed-Loop System Identification," *Automatica*, vol. 41, no. 2, pp. 241–249, 2005.

[91] T. Söderström, "Errors-in-Variables Methods in System Identification," *Automatica*, vol. 43, no. 6, pp. 939–958, 2007.

[92] C. T. Chou and M. Verhaegen, "Subspace Algorithms for the Identification of Multivariable Dynamic Errors-in-Variables Models," *Automatica*, vol. 33, no. 10, pp. 1857–1869, 1997.

[93] S. Van Huffel and P. Lemmerling, *Total Least Squares and Errors-in-Variables Modeling: Analysis, Algorithms and Applications*. Springer Science & Business Media, 2013.

[94] B. D. Anderson, "Identification of Scalar Errors-in-Variables Models with Dynamics," *Automatica*, vol. 21, no. 6, pp. 709–716, 1985.

[95] L. Ljung, *System Identification: Theory for the User, 2nd ed.* Upper Saddle River, NJ: Prentice-Hall Information and Systems Sciences, 1999.

[96] ——, *System Identification Toolbox: For Use with Matlab: Computation, Visualization, Programming: User's Guide, Version 5.* The Mathworks, 2001.

[97] D. P. White, J. G. Wozniak, and D. A. Lawrence, "Missile Autopilot Design Using a Gain Scheduling Technique," in *Proc. 26th SE Symp. on Sys. Theory.* IEEE, 1994, pp. 606–610.

[98] T. Richardson, P. Davison, M. Lowenberg, and M. di Bernardo, "Control of Nonlinear Aircraft Models Using Dynamic State-Feedback Gain Scheduling," in *AIAA Guid. Nav. Contr. Conf. and Ex.*, 2003, p. 5503.

[99] J. S. Shamma and J. R. Cloutier, "Gain-Scheduled Missile Autopilot Design Using Linear Parameter Varying Transformations," *AIAA J. Guid. Contr. Dyn.*, vol. 16, no. 2, pp. 256–263, 1993.

[100] R. A. Nichols, R. T. Reichert, and W. J. Rugh, "Gain Scheduling for H-infinity Controllers: A Flight Control Example," *IEEE Trans. on Contr. Sys. Tech.*, vol. 1, no. 2, pp. 69–79, 1993.

[101] N. Hovakimyan, C. Cao, E. Kharisov, E. Xargay, and I. M. Gregory, "$\mathcal{L}_1$ Adaptive Control for Safety-Critical Systems," *IEEE Contr. Sys. Mag.*, vol. 31, no. 5, pp. 54–104, 2011.

[102] Z. T. Dydek, A. M. Annaswamy, and E. Lavretsky, "Adaptive Control and the NASA X-15-3 Flight Revisited," *IEEE Contr. Sys.*, vol. 30, no. 3, pp. 32–48, 2010.

[103] J. S. Orr and C. J. Dennehy, "Analysis of the X-15 Flight 3-65-97 Divergent Limit-Cycle Oscillation," *J. of Aircraft*, pp. 1–14, 2016.

[104] Y. Rahman, A. Xie, J. B. Hoagg, and D. S. Bernstein, "A Tutorial and Overview of Retrospective Cost Adaptive Control," in *Proc. Amer. Contr. Conf.*, Boston, MA., July 2016, pp. 3386–3409.

[105] F. Sobolic and D. Bernstein, "Aerodynamic-Free Adaptive Control of the NASA Generic Transport Model," in *AIAA Guid. Nav. Contr. Conf.*, 2013,AIAA-2013-4999.

[106] E. D. Sumer and D. S. Bernstein, "Retrospective Cost Adaptive Control with Error-Dependent Regularization for MIMO Systems with Uncertain Nonminimum-Phase Transmission Zeros," in *AIAA Guid. Nav. Contr. Conf.*, Minneapolis, MN., August 2012, p. 4670.

[107] U. Forssell and L. Ljung, "Closed-Loop Identification Revisited," *Automatica*, vol. 35, no. 7, pp. 1215–1241, 1999.

[108] MATLAB, "Aerospace Toolbox: User's Guide (2016b)," https://www.mathworks.com/help/simulink/examples/designing-a-guidance-system-in-matlab-and-simulink.html, Natick, Massachusetts, 2016, accessed: 2017-04-23.