High Performance Control

T. T. Tay[1]

I. M. Y. Mareels[2]

J. B. Moore[3]

1997

1. Department of Electrical Engineering, National University of Singapore, Singapore.
2. Department of Electrical and Electronic Engineering, University of Melbourne, Australia.
3. Department of Systems Engineering, Research School of Information Sciences and Engineering, Australian National University, Australia.

Preface

The engineering objective of high performance control using the tools of optimal control theory, robust control theory, and adaptive control theory is more achievable now than ever before, and the need has never been greater. Of course, when we use the term *high performance control* we are thinking of achieving this in the real world with all its complexity, uncertainty and variability. Since we do not expect to always achieve our desires, a more complete title for this book could be “Towards High Performance Control”.

To illustrate our task, consider as an example a disk drive tracking system for a portable computer. The better the controller performance in the presence of eccentricity uncertainties and external disturbances, such as vibrations when operated in a moving vehicle, the more tracks can be used on the disk and the more memory it has. Many systems today are control system limited and the quest is for high performance in the real world.

In our other texts Anderson and Moore (1989), Anderson and Moore (1979), Elliott, Aggoun and Moore (1994), Helmke and Moore (1994) and Mareels and Polderman (1996), the emphasis has been on optimization techniques, optimal estimation and control, and adaptive control as separate tools. Of course, robustness issues are addressed in these separate approaches to system design, but the task of blending optimal control and adaptive control in such a way that the strengths of each is exploited to cover the weakness of the other seems to us the only way to achieve high performance control in uncertain and noisy environments.

The concepts upon which we build were first tested by one of us, John Moore, on high order NASA flexible wing aircraft models with flutter mode uncertainties. This was at Boeing Commercial Airplane Company in the late 1970s, working with Dagfinn Gangsaas. The engineering intuition seemed to work surprisingly well and indeed 180° phase margins at high gains was achieved, but there was a shortfall in supporting theory. The first global convergence results of the late 1970s for adaptive control schemes were based on least squares identification. These were harnessed to design adaptive loops and were used in conjunction with linear quadratic optimal control with frequency shaping to achieve robustness to flutter phase uncertainty. However, the blending of those methodologies in itself lacked theoretical support at the time, and it was not clear how to proceed to systematic designs with guaranteed stability and performance properties.

A study leave at Cambridge University working with Keith Glover allowed time for contemplation and reading the current literature. An interpretation of the Youla-Kučera result on the class of all stabilizing controllers by John Doyle gave a clue. Doyle had characterized the class of stabilizing controllers in terms of a stable filter appended to a standard linear quadratic Gaussian LQG controller design. But this was exactly where our adaptive filters were placed in the designs we developed at Boeing. Could we improve our designs and build a complete theory now? A graduate student Teng Tiow Tay set to work. Just as the first simulation studies were highly successful, so the first new theories and new algorithms seemed very powerful. Tay had also initiated studies for nonlinear plants, conveniently characterizing the class of all stabilizing controllers for such plants.

At this time we had to contain ourselves not to start writing a book right away. We decided to wait until others could flesh out our approach. Iven Mareels and his PhD student Zhi Wang set to work using averaging theory, and Roberto Horowitz and his PhD student James McCormick worked applications to disk drives. Meanwhile, work on Boeing aircraft models proceeded with more conservative objectives than those of a decade earlier. No aircraft engineer will trust an adaptive scheme that can take over where off-line designs are working well. Weiyong Yan worked on more aircraft models and developed nested-loop or iterated designs based on a sequence of identification and control exercises. Also Andrew Paice and Laurence Irlicht worked on nonlinear factorization theory and functional learning versions of the results. Other colleagues Brian Anderson and Robert Bitmead and their coworkers Michel Gevers and Robert Kosut and their PhD students have been extending and refining such design approaches. Also, back in Singapore, Tay has been applying the various techniques to problems arising in the context of the disk drive and process control industries.

Now is the time for this book to come together. Our objective is to present the practice and theory of high performance control for real world environments. We proceed through the door of our research and applications. Our approach specializes to standard techniques, yet gives confidence to go beyond these. The idea is to use prior information as much as possible, and on-line information where this is helpful. The aim is to achieve the performance objectives in the presence of variations, uncertainties and disturbances. Together the off-line and on-line approach allows high performance to be achieved in realistic environments.

This work is written for graduate students with some undergraduate background in linear algebra, probability theory, linear dynamical systems, and preferably some background in control theory. However, the book is complete in itself, including appropriate appendices in the background areas. It should appeal to those wanting to take only one or two graduate level semester courses in control and wishing to be exposed to key ideas in optimal and adaptive control. Yet students having done some traditional graduate courses in control theory should find that the work complements and extends their capabilities. Likewise control engineers in industry may find that this text goes beyond their background knowledge and that it will help them to be successful in their real world controller designs.

Acknowledgements

This work was partially supported by grants from Boeing Commercial Airplane Company, and the Cooperative Research Centre for Robust and Adaptive Systems. We wish to acknowledge the typesetting and typing support of James Ashton and Marita Rendina, and proof reading support of PhD students Andrew Lim and Jason Ford.

Contents

List of Figures

List of Tables

CHAPTER **1**

Performance Enhancement

1.1 Introduction

Science has traditionally been concerned with describing nature using mathematical symbols and equations. Applied mathematicians have traditionally been studying the sort of equations of interest to scientists. More recently, engineers have come onto the scene with the aim of manipulating or controlling various processes. They introduce (additional) control variables and adjustable parameters to the mathematical models. In this way, they go beyond the traditions of science and mathematics, yet use the tools of science and mathematics and, indeed, provide challenges for the next generation of mathematicians and scientists.

Control engineers, working across all areas of engineering, are concerned with adding *actuators* and *sensors* to engineering systems which they call *plants*. They want to monitor and control these plants with controllers which process information from both desired responses (commands) and sensor signals. The controllers send control signals to the actuators which in turn affect the behavior of the plant. They are concerned with issues such as actuator and sensor selection and location. They must concern themselves with the underlying processes to be controlled and work with relevant experts depending on whether the plant is a chemical system, a mechanical system, an electrical system, a biological system, or an economic system. They work with block diagrams, which depict actuators, sensors, processors, and controllers as separate blocks. There are directed arrows interconnecting these blocks showing the direction of information flow as in Figure 1.1. The directed arrows represent signals, the blocks represent functional operations on the signals. Matrix operations, integrations, and delays are all represented as blocks. The blocks may be (matrix) transfer functions or more general time-varying or nonlinear operators.

Control engineers talk in terms of the *controllability* of a plant (the effectiveness of actuators for controlling the process), and the *observability* of the plant (the effectiveness of sensors for observing the process). Their big concept is that

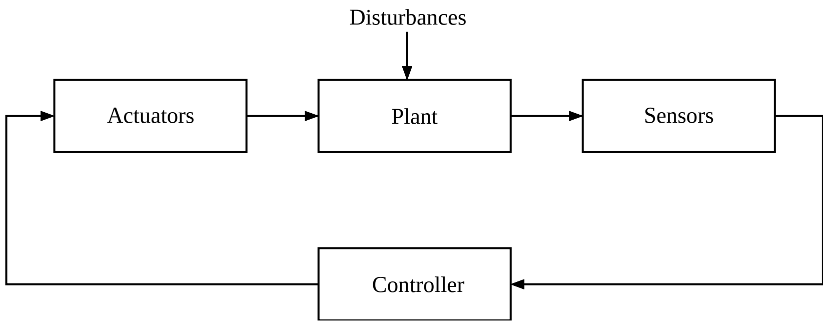


FIGURE 1.1. Block diagram of feedback control system

of *feedback*, and their big challenge is that of *feedback controller* design. Their territory covers the study of *dynamical systems* and *optimization*. If the plant is not performing to expectations, they want to detect this under-performance from sensors and suitably process this sensor information in *controllers*. The controllers in turn generate performance enhancing feedback signals to the actuators. How do they do this?

The approach to controller design is to first understand the physics or other scientific laws which govern the behavior of the plant. This usually leads to a mathematical model of the process, termed a *plant model*. There are invariably aspects of plant behavior which are not captured in precise terms by the plant model. Some uncertainties can be viewed as *disturbance signals*, and/or *plant parameter variations* which in turn are perhaps characterized by probabilistic models. *Unmodeled dynamics* is a name given to dynamics neglected in the plant model. Such are sometimes characterized in frequency domain terms. Next, *performance measures* are formulated in terms of the plant model and taking account of uncertainties. There could well be hard *constraints* such as limits on the controls or states. Control engineers then apply mathematical tools based in optimization theory to achieve their design of the control scheme. The design process inevitably requires compromises or trade-offs between various conflicting performance objectives. For example, achieving *high performance* for a particular set of conditions may mean that the controller is too finely tuned, and so can not yet cope with the contingencies of everyday situations. A racing car can cope well on the race track, but not in city traffic.

The designer would like to improve performance, and this is done through increased feedback in the control scheme. However, in the face of disturbances or plant variations or uncertainties, increasing feedback in the frequency bands of high uncertainty can cause instability. Feedback can give us high performance for the plant model, and indeed insensitivity to small plant variations, but poor performance or even instability of the actual plant. The term controller *robustness* is used to denote the ability of a controller to cope with these real world uncertainties. Can *high performance* be achieved in the face of uncertainty and change? This is the challenge taken up in this book.

1.2 Beyond Classical Control

Many control tasks in industry have been successfully tackled by very simple analog technology using classical control theory. This theory has matched well the technology of its day. Classical three-term-controllers are easy to design, are robust to plant uncertainties and perform reasonably well. However, for improved performance and more advanced applications, a more general control theory is required. It has taken a number of decades for digital technology to become the norm and for modern control theory, created to match this technology, to find its way into advanced applications. The market place is now much more competitive so the demands for high performance controllers at low cost is the driving force for much of what is happening in control. Even so, the arguments between classical control and modern control persist. Why?

The classical control designer should never be underestimated. Such a person is capable of achieving good trade-offs between performance and robustness. Frequency domain concepts give a real feel for what is happening in a process, and give insight as to what happens loop-by-loop as they are closed carefully in sequence. An important question for a modern control person (with a sophisticated optimization armory of Riccati equations and numerical programming packages and the like) to ask is: How can we use classical insights to make sure our modern approach is really going to work in this situation? And then we should ask: Where does the adaptive control expert fit into this scene? Has this expert got to fight both the classical and modern notions for a niche?

This book is written with a view to blending insights and methods from classical, optimal, and adaptive control so that each contributes at its point of strength and compensates for the weakness of others, so as to achieve both *robust control* and *high performance control*. Let us examine these strengths and weaknesses in turn, and then explore some design concepts which are perhaps at the interface of all three methods, called *iterated design, plug-in controller design, hierarchical design* and *nested controller design*.

Some readers may think of optimal control for linear systems subject to quadratic performance indices as classical control, since it is now well established in industry, but we refer to such control here as optimal control. Likewise, self-tuning control is now established in industry, but we refer to this as adaptive control.

Classical Control

The strength of classical control is that it works in the frequency domain. Disturbances, unmodeled dynamics, control actions, and system responses all predominate in certain frequency bands. In those frequency bands where there is high phase uncertainty in the plant, feedback gains must be low. Frequency characteristics at the unity gain cross-over frequency are crucial. Controllers are designed to shape the frequency responses so as to achieve stability in the face of plant uncertainty, and moreover, to achieve good performance in the face of this uncertainty. In other words, a key objective is *robustness*.

It is then not surprising that the classical control designer is comfortable working with transfer functions, poles and zeros, magnitude and phase frequency responses, and the like.

The plant models of classical control are linear and of low order. This is the case even when the real plant is obviously highly complex and nonlinear. A small signal analysis or identification procedure is perhaps the first step to achieve the linear models. With such models, controller design is then fairly straightforward. For a recent reference, see Ogata (1990).

The limitation of classical control is that it is fundamentally a design approach for a single-input, single-output plant working in the neighborhood of a single operating point. Of course, much effort has gone into handling multivariable plants by closing control loops one at a time, but what is the best sequence for this?

In our integrated approach to controller design, we would like to tackle control problems with the strengths of the frequency domain, and work with transfer functions where possible. We would like to achieve high performance in the face of uncertainty. The important point for us here is that we do not design *frequency shaping filters* in the first instance for the control loop, as in classical designs, but rather for formulating performance objectives. The optimal multivariable and adaptive methods then systematically achieve controllers which incorporate the frequency shaping insights of the classical control designer, and thereby the appropriate frequency shaped filters for the control loop.

Optimal Control

The strength of optimal control is that powerful numerical algorithms can be implemented off-line to design controllers to optimize certain performance objectives. The optimization is formulated and achieved in the time domain. However, in the case of time-invariant systems, it is often feasible to formulate an equivalent optimization problem in the frequency domain. The optimization can be for multivariable plants and controllers.

One particular class of optimal control problems which has proved powerful and now ubiquitous is the so-called linear quadratic Gaussian (LQG) method, see Anderson and Moore (1989), and Kwakernaak and Sivan (1972). A key result is the *Separation Theorem* which allows decomposition of an optimal control problem for linear plants with Gaussian noise disturbances and quadratic indices into two subproblems. First, the optimal control of linear plants is addressed assuming knowledge of the internal variables (*states*). It turns out that the optimal solutions for a noise free (*deterministic*) setting and an additive white Gaussian plant driving noise setting are identical. The second task addressed is the estimation of the plant model's internal variables (states) from the plant measurements in a noise (*stochastic*) setting. The Separation Theorem then tells us that the best design approach is to apply the *Certainty Equivalence Principle*, namely to use the state estimates in lieu of the actual states in the feedback control law. Remarkably, under the relevant assumptions, optimality is achieved. This task decomposition allows the designer to focus on the effectiveness of actuators and sensors separately, and indeed to address areas of weakness one at a time. Certainly, if a state feedback design does not deliver performance, then how can any output feedback controller? If a state estimator achieves poor state estimates, how can internal variables be controlled effectively? Unfortunately, this Separation Principle does not apply for general nonlinear plants, although such a principle does apply when working with so-called *information states* instead of state estimates. Information states are really the totality of knowledge about the plant states embedded in the plant observations.

Of course, in replacing states by state estimates there is some loss. It turns out that there can be severe loss of robustness to phase uncertainties. However, this loss can be recovered, at least to some extent, at the expense of optimality of the original performance index, by a technique known as *loop recovery* in which the feedback system sensitivity properties for state feedback are recovered in the case of state estimate feedback. This is achieved by working with colored fictitious noise in the nominal plant model, representing plant uncertainty in the vicinity of the so-called cross-over frequency where loop gains are near unity. There can be "total" sensitivity recovery in the case of minimum phase plants.

There are other optimal methods which are in some sense a more sophisticated generalization of the LQG methods, and are potentially more powerful. They go by such names as H_∞ and ℓ_1 optimal control. These methods in effect do not perform the optimization over only one set of input disturbances but rather the optimization is performed over an entire class of input disturbances. This gives rise to a so-called worst case control strategy and is often referred to as robust controller design, see for example, Green and Limebeer (1994), and Morari and Zafiriou (1989).

The inherent weakness of the optimization approach is that although it allows incorporation of a class of robustness measures in a performance index, it is not clear how to best incorporate *all* the robustness requirements of interest into the performance objectives. This is where classical control concepts come to the rescue, such as in the loop recovery ideas mentioned above, or in appending other *frequency shaping filters* to the nominal model. The designer should expect a trial-and-error process so as to gain a feel for the particular problem in terms of the trade-offs between performance for a nominal plant, and robustness of the controller design in the face of plant uncertainties. Thinking should take place both in the frequency domain and the time domain, keeping in mind the objectives of robustness and performance. Of course, any trial-and-error experiment should be executed with the most advanced mathematical and software tools available and not in an ad hoc manner.

Adaptive Control

The usual setting for adaptive control is that of low order single-input, single-output plants as for classical design. There are usually half a dozen or so parameters to adjust on-line requiring some kind of *gradient search* procedure, see for example Goodwin and Sin (1984) and Mareels and Polderman (1996). This setting is just as limited as that for classical control. Of course, there are cases where tens of parameters can be adapted on-line, including cases for multivariable plants, but such situations must be tackled with great caution. The more parameters to learn, the slower the learning rate. The more inputs and outputs, the more problems can arise concerning uniqueness of parameterization. Usually, the so-called *input/output representations* are used in adaptive control, but these are notoriously sensitive to parameter variations as model order increases. Finally, naively designed adaptive schemes can let you down, even catastrophically.

So then, what are the strengths of adaptive control, and when can it be used to advantage? Our position is that taken by some of the very first adaptive control designers, namely that adaptive schemes should be designed to augment robust off-line-designed controllers. The idea is that for a prescribed range of plant variations or uncertainties, the adaptive scheme should only improve performance over that of the robust controller. Beyond this range, the adaptive scheme may do well with enough freedom built into it, but it may cause instability. Our approach is to eliminate risk of failure, by avoiding too difficult a design task or using either a too simple or too complicated adaptive scheme. Any adaptive scheme should be a reasonably simple one involving only a few adaptive gains so that adaptations can be rapid. It should fail softly as it approaches its limits, and these limits should be known in advance of application.

With such adaptive controller augmentations for robust controllers, it makes sense for the robust controller to focus on stability objectives over the known range of possible plant variations and uncertainties, and for the adaptive or *self-tuning scheme* to beef up performance for any particular situation or setting. In this way performance can be achieved along with robustness without the compromises usually expected in the absence of adaptations or on-line calculations.

A key issue in adaptive schemes is that of control signal excitation for associated *on-line identification* or *parameter adjustment*. The terms *sufficiently exciting* and *persistence of excitation* are used to describe signals in the adaptation context. Learning objectives are in conflict with control objectives, so that there must be a balance in applying excitation signals to achieve a stable, robust, and indeed high performance adaptive controller. This balancing of conflicting interests is termed *dual control*.

1.3 Robustness and Performance

With the lofty goal of achieving high performance in the face of disturbances, plant variations and uncertainties, how do we proceed? It is crucial in any controller design approach to first formulate a plant model, characterize uncertainties and disturbances, and quantify measures of performance. This is a starting point. The best next step is open to debate. Our approach is to work with the class of stabilizing controllers for a nominal plant model, search within this class for

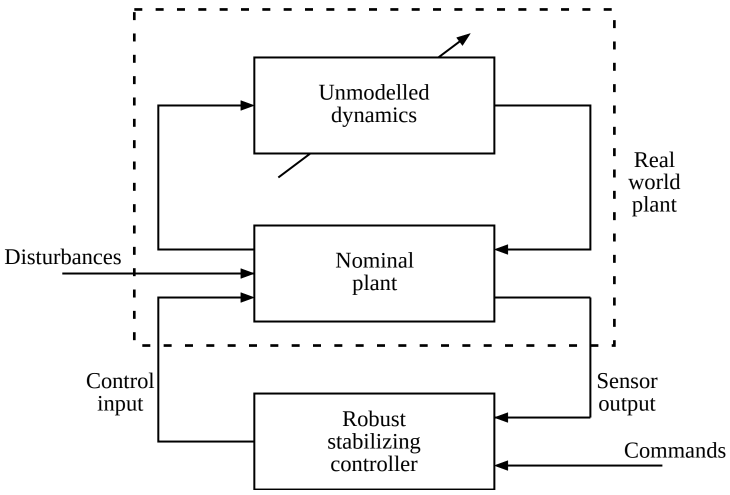


FIGURE 3.1. Nominal plant, robust stabilizing controller

a robust controller which stabilizes the plant in the face of its uncertainties and variations, and then tune the controller on-line to enhance controller performance, moment by moment, adapting to the real world situation. The adaptation may include reidentification of the plant, it may reshape the nominal plant, requantify the uncertainties and disturbances and even shift the performance objectives.

The situation is depicted in Figures 3.1 and 3.2. In Figure 3.1, the real world plant is viewed as consisting of a nominal plant and unmodeled dynamics driven by a control input and disturbances. There are sensor outputs which in turn feed into a feedback controller driven also by commands. It should be both stabilizing for the nominal plant and robust in that it copes with the unmodeled dynamics and disturbances. In Figure 3.2 there is a further feedback control loop around the real world plant/robust controller scheme of Figure 3.1. The additional controller is termed a performance enhancement controller.

Nominal Plant Models

Our interest is in dynamical systems, as opposed to static ones. Often for maintaining a steady state situation with small control actions, real world plants can be approximated by *linear dynamical systems*. A useful generalization is to include random disturbances in the model so that they become *linear dynamical stochastic systems*. The simplest form of disturbance is linearly filtered white, zero mean, Gaussian noise. Control theory is most developed for such deterministic or stochastic plant models, and more so for the case of time-invariant systems. We build as much of our theory as possible for *linear, time-invariant, finite-dimensional dynamical systems* with the view to subsequent generalizations.

Control theory can be developed for either *continuous-time (analog) models*, or *discrete-time (digital) models*, and indeed some operator formulations do not

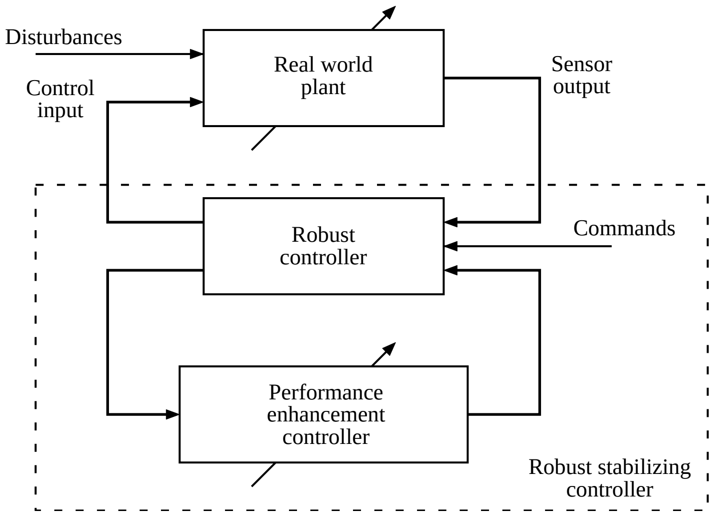


FIGURE 3.2. Performance enhancement controller

distinguish between the two. We select a discrete-time setting with the view to computer implementation of controllers. Of course, most real world engineering plants are in continuous time, but since analog-to-digital and digital-to-analog conversion are part and parcel of modern controllers, the discrete-time setting seems to us the one of most interest. We touch on sampling rate selection, inter-sample behavior and related issues when dealing with implementation aspects.

Most of our theoretical developments, even for the adaptive control loops, are carried out in a multivariable setting, that is, the signals are vectors.

Of course, the class of nominal plants for design purposes may be restricted as just discussed, but the expectation in so-called robust controller design is that the controller designed for the nominal plant also copes well with actual plants that are "near" in some sense to the nominal one. To achieve this goal, actual plant nonlinearities or uncertainties are often, perhaps crudely, represented as *fictitious noise disturbances*, such as is obtained from filtered white noise introduced into a linear system.

It is important that the plant model also include sensor and actuator dynamics. It is also important to append so-called *frequency shaping filters* to the nominal plant with the view to controlling the outputs of these filters, termed *derived variables* or *disturbance response* variables, see Figure 3.3. This allows us to more readily incorporate robustness measures into a performance index. This last point is further discussed in the next subsections.

Unmodeled Dynamics

A nominal model usually neglects what it cannot conveniently and precisely characterize about a plant. However, it makes sense to characterize what has been ne-

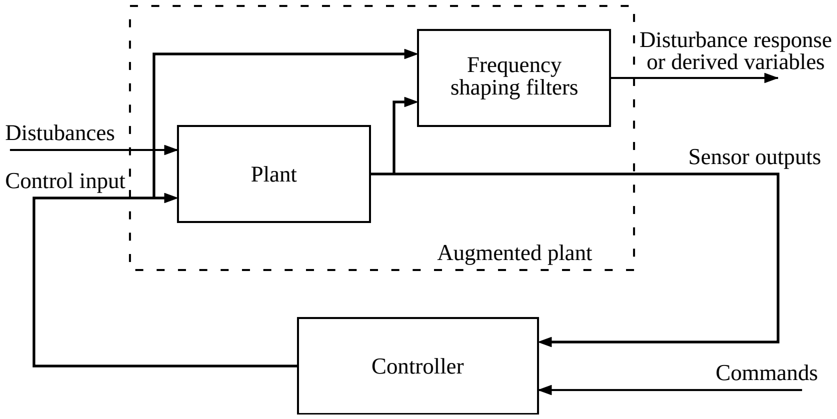


FIGURE 3.3. Plant augmentation with frequency shaped filters

glected in as convenient a way as possible, albeit loosely. Aerospace models, for example, derived from finite element methods are very high in order, and often too complicated to work with in a controller design. It is reasonable then at first to neglect all modes above the frequency range of expected significant control actions. Fortunately in aircraft, such neglected modes are stable, albeit perhaps very lightly damped in flexible wing aircraft. It is absolutely vital that these modes not be excited by control actions that could arise from controller designs synthesized from studies with low order models. The neglected dynamics introduce phase uncertainty in the low order model as frequency increases, and this fact should somehow be taken into account. Such uncertainties are referred to as *unmodeled dynamics*.

Performance Measures and Constraints

In an airplane flying in turbulence, wing root stress should be minimized along with other variables. But there is no sensor that measures this stress. It must be estimated from sensor measurements such as pitch measurements and accelerometers, and knowledge of the aircraft dynamics (kinematics and aerodynamics). This example illustrates that performance measures may involve internal (state) variables. Actually, it is often worthwhile to work with filtered versions of these state variables, and indeed with filtered control variables, and filtered output variables, since we may be interested in their behavior only in certain frequency bands. As already noted, we term all these relevant variables *derived variables* or *disturbance response* variables. Usually, there must be a compromise between control energy and performance in terms of these derived variables. Derived variables are usually generated by appropriate *frequency shaping filter* augmentations to a “first cut” plant model, as depicted in Figure 3.3. The resulting model is the nominal model of interest for controller design purposes.

In control theory, performance measures are usually designed for a *regulation* situation, or for a *tracking* situation. In regulation, ideally there should be a steady state situation, and if there is perturbance from this by external disturbances, then we would like to regulate to zero any disturbance response in the derived variables. The disturbance can be random such as when wind gusts impinge on an antenna, or *deterministic* such as when there is eccentricity in a disk drive system giving rise to periodic disturbances.

In tracking situations, there is some desired trajectory which should be followed by certain plant variables; again these can be derived variables rather than sensor measurements. Clearly, regulation is a special case of tracking.

In this text we consider first traditional performance measures for nominal plant models, such as are used in *linear quadratic Gaussian* (LQG) control theory, see Anderson and Moore (1989) and in the so-called H_∞ *control* and ℓ_1 *control* theories, see Francis (1987), Vidyasagar (1986) and Green and Limebeer (1994).

The LQG theory is derived based on penalizing control energy and plant energy of internal variables, termed *states*, in the presence of *white noise disturbances*. That is, there is a sum of squares index which is optimized over all control actions. In the linear quadratic Gaussian context it turns out that the optimal control signal is given from a feedback control law. The H_∞ theory is based on penalizing a sum of squares index in the presence of *worst case disturbances* in an appropriate class. The ℓ_1 theory is based on penalizing the worst peak in the response of internal states and/or control effort in the presence of worst case bounded disturbances. Such a theory is most appropriate when there are hard constraints on the control signals or states.

The Class of Stabilizing Controllers

Crucial to our technical approach is a characterization of the class of all stabilizing controllers in terms of a parameter termed Q. In fact, Q is not a parameter such as a gain or time constant, but is a stable (bounded-input, bounded-output) filter built into a stabilizing controller in a manner to be described in some detail as we proceed. This theory has been developed in a discrete-time setting by Kučera (1979) and in a continuous-time setting by Youla, Bongiorno and Jabr (1976a). Moreover, all the relevant input/output operators, (matrix transfer functions) of the associated closed-loop system turn out to be linear, or more precisely affine, in the operator (matrix transfer function) Q. In turn, this facilitates optimization over stable Q, or equivalently, over the class of stabilizing controllers for a nominal plant, see Vidyasagar (1985) and Boyd and Barratt (1991).

Our performance enhancement techniques work with finite-dimensional adaptive filters Q with parameters adjusted on line so as to minimize a sum of squares performance index. The effectiveness of this arrangement seems to depend on the initial stabilizing controller being a robust controller, probably because it exploits effectively all *a priori* knowledge of the plant.

A dual concept is the class of all plants stabilized by a given stabilizing controller which is parameterized in terms of stable "filter" S.

In our approach, depicted in Figure 3.4, the unmodeled dynamics of a plant

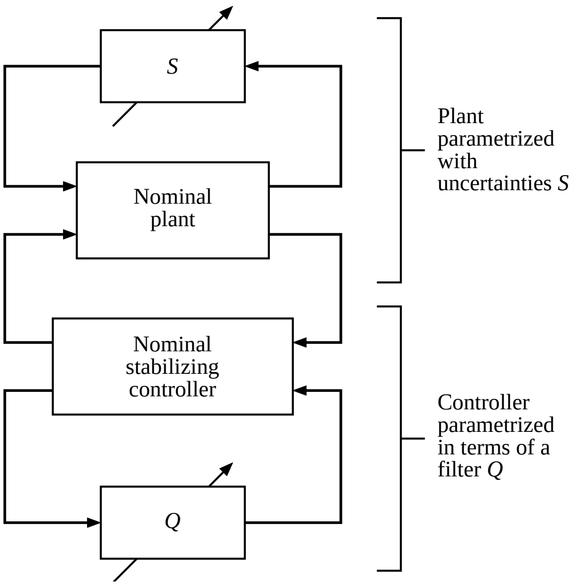

S
Nominal plant
Nominal stabilizing controller
Q
Plant parametrized with uncertainties S
Controller parametrized in terms of a filter Q

FIGURE 3.4. Plant/controller (Q, S) parameterization

model can be represented in terms of this “filter” S, which is zero when the plant model is a precise representation of the plant. Indeed with a plant parameterized in terms of S and a controller parameterized in terms of Q, the resulting closed-loop system turns out to be stable if and only if the nominal controller (with $Q = 0$) stabilizes the nominal plant (with $S = 0$) and Q stabilizes S, irrespective of whether or not Q and S are stable themselves, see Figure 3.5. This result is a basis for our controller performance (and robustness) analysis. With the original stabilizing controller being robust, it appears that although S is of high order, it can be approximated in many of our design examples as a low order, or at least a low gain system without too much loss. When this situation occurs, the proposed controllers involving only low order, possibly adaptive Q can achieve high performance enhancement.

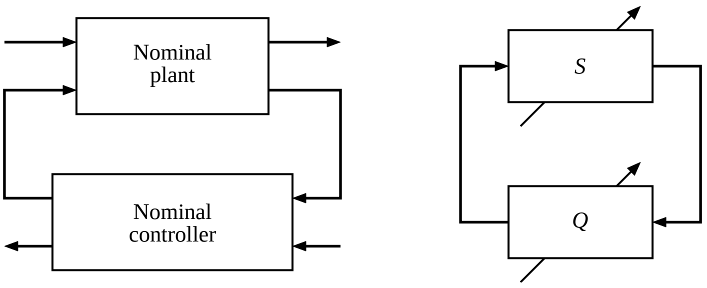


FIGURE 3.5. Two loops must be stabilizing

Adaptations

Adaptive schemes perform best when as much as possible *a priori information* about the plant is incorporated into the design of the adaptive algorithms. It is clear that a good way to achieve this is to first include such information into the off-line design of a fixed *robust controller*.

In *direct adaptive control,* the parameters for adaptation are those of a fixed structure controller, whereas in *indirect adaptive control,* the parameters for tuning are, in the first instance, those of some plant model. These plant parameters which are estimated on-line, are then used in a controller design law to construct on-line a controller with adaptive properties.

Adaptive schemes that work effectively in changing environments usually require suitably strong and rich *excitation* of control signals. Such excitation of itself is in fact against the control objectives. Of course, one could argue that if the plant is performing well at some time, there is no need for adaptation or of excitation signals. However the worst case scenario in such a situation is that the adjustable parameters adjust themselves on the basis of insufficient data and drift to where they cause instability. The instability may be just a burst, for the excitation associated with the instability could lead to appropriate adjustment of parameters to achieve stability. But then again, the instability may not lead to suitably rich signals for adequate learning and thus adequate control. This point is taken up again below. Such situations are completely avoided in our approach, because *a priori* constraints are set on the range of allowable adjustment, based on an *a priori* stability analysis.

One method to achieve a sufficiently rich excitation signal is to introduce random (bounded) noise, that is *stochastic excitation,* for then even the most "devious" adaptive controller will not cancel out the unpredictable components of this noise, as it could perhaps for predictable deterministic signals.

Should there be instability, then it is important that one does not rely on the signal build up itself as a source of excitation, since this will usually reflect only one unstable mode which dominates all other excitation, and allow estimation of only the one or two parameters associated with this mode, with other parameters perhaps drifting. It is important that the frequency rich (possibly random) excitation signal grows according to the instability. Only in this way can all modes be identified at similar rates and incipient instability be nipped in the bud by the consequent adaptive control action.

How does one analyze an adaptive scheme for performance? Our approach is to use *averaging analysis,* see Sanders and Verhulst (1985), Anderson, Bitmead, Johnson, Kokotovic, Kosut, Mareels, Praly and Riedle (1986), Mareels and Polderman (1996) and Solo and Kong (1995). This analysis looks at averaging out the effects of fast dynamics in the closed loop so as to highlight the effect of the relative slow adaptation process. This *time scale separation* approach is very powerful when adaptations are relatively slow compared to the dynamics of the system. Averaging techniques tell us that our adaptations can help performance of a robust controller. There is less guaranteed performance enhancement at the margins of robust controller stability.

Iterated Design

Of course, on-line adaptations using simple adaptive schemes are ideal when these work effectively, but perhaps as a result of implementing a possibly crude controller with limited *a priori* knowledge, direct or indirect adaptations may not be effective. The *a priori* knowledge should be updated using some identification in closed loop, not necessarily of the plant itself. The identification can then be used to refine the controller design. This interaction between identification and controller design, which is at the heart of on-line adaptive control, is sometimes best carried out in an iterative fashion based on a more complete data analysis than could be achieved by simple recursive schemes, see for example Anderson and Kosut (1991), Zang, Bitmead and Gevers (1991), Schrama (1992a) and Lee, Anderson, Kosut and Mareels (1993). Even so, we see such an off-line design approach as being in the spirit of adaptive control. Most importantly, we show that for a range of control objectives, one can proceed in an iterated identification control design manner, without losing the ability to reach an optimal design. That is, the iterated control design is able to incrementally improve the control performance at each iteration, given accurate models. Moreover, it can recover from any bad design at a later stage.

Nested or Recursive Design

Concepts related to those behind iterated design are that of a *plug-in controller* augmentation and that of *hierarchical design*. We are familiar with the classical design approach of adding controllers here and there in a complex system to enhance performance as experience with the system grows. Of course, the catch is that the most recent plug-in controller addition may counter earlier controller actions. Can we somehow proceed in a systematic manner?

Also, we are familiar with different control loops dealing with different aspects of the control task. The inner loop is for tight tracking, say, and an outer loop is for the determination of some desired (perhaps optimal) trajectory, and even further control levels may exist in a hierarchy to set control tasks at a more strategic level. Is there a natural way to embed control loops within control loops, and set up hierarchies of control?

Our position on plug-in controllers is that controller designs can be performed so that plug-in additions can take place in a systematic manner. With each addition, there is an optimization which takes the design a step in the direction of the overall goal. The optimizations for each introduced plug-in controller must not conflict with each other. It may be that the first controller focuses on one frequency band, and later controllers introduce focus on other bands as more information becomes available. It may be that the first controller is designed for robustness and the second to enhance performance for a particular setting, and a third “controller” is designed to switch in appropriate controllers as required. It may be that the first controller uses a frequency domain criterion for optimization and a second controller works in the time domain. Our experience is that a natural way to proceed is from a robust design, to adaptations for performance enhancement, and then to learning so as to build up a data base of experience for future decisions.

There is a key to a systematic approach to iterated design, plug-in controller design, and hierarchical control, which we term *recursive controller* or *nested controller* design. It exploits the mathematical concept of successive approximation by continued linear fraction expansions, for the control setting.

1.4 Implementation Aspects and Case Studies

Control theory has in the past been well in advance of practical implementation, whereas today the hardware and software technology is available for complex yet reliable controller design and implementation. Now it is possible for quite general purpose application software such as Rlab or commercially available packages MATLAB[*] and Xmath[†] to facilitate the controller design process, and even to implement the resultant controller from within the packages. For implementation in *microcontroller* hardware not supported by these packages, there are other software packages such as Matcom which can machine translate the generic algorithms into more widely supported computer languages such as C and Assembler. As the on-line digital technology becomes faster and the calculations are parallelized, the sampling rates for the signals and the controller algorithm complexity can be increased.

There will always be applications at the limits of technology. With the goal of increased system efficiency and thus increased controller performance in all operating environments, even simple processes and control tasks will push the limits of the technology as well as theory.

One aim in this book is to set the stage for practical implementation of high performance controllers. We explore various hardware and software options for the control engineer and raise questions which must be addressed in practical implementation. *Multirate sampling* strategies and sampling rate selection are discussed along with other issues of getting a controller into action.

Our aim in presenting controller design laboratory case studies is to show the sort of compromises that are made in real world implementations of advanced high performance control strategies.

1.5 Book Outline

In Chapter 2, a class of linear plant *stabilizing controllers* for a linear plant model is parameterized in terms of a stable filter, denoted Q. This work is based on a theory of *coprime factorization* and linear fractional representations. The idea is

[*]MATLAB® is a registered trademark of the MathWorks, Inc.
[†]Xmath® is a registered trademark of Integrated Systems, Inc.

introduced that any stabilizing controller can be augmented to include a physical stable filter Q, and this filter tuned either off-line or on-line to optimize performance. The notion of the class of *stabilizing regulators* to absorb classes of deterministic disturbances is also developed.

In Chapter 3, the controller *design environment* is characterized in terms of the uncertainties associated with any plant model, be they signal uncertainties, *structured* or *unstructured plant uncertainties*. The concept of frequency shaped uncertainties is developed as a dual theory to the *Q-parameterization* theory of Chapter 2. In particular, the class of plants stabilized by a given controller is studied via an S-parameterization. The need for plant model identification in certain environments is raised.

In Chapter 4, the notion of off-line optimizing a stabilizing controller design to achieve various performance objectives is introduced. One approach is that of optimal-Q filter selection. Various performance indices and methods to achieve optimality are studied such as those penalizing energy of the tracking error and control energy, or penalizing maximum tracking error subject to control limits, or penalizing peak spectral response.

Chapter 5 discusses the interaction between control via the Q-parameterization of all stabilizing controllers for a nominal plant model and identification via the S-parameterization of all plants stabilized by a controller. Two different schemes are presented. They differ in the way the identification proceeds. In the so-called iterated design, the same S parameterization is refined in recursive steps, followed by a control update step. In the so-called nested design, successive S parameters of the residual plant-model mismatch are identified. Each nested S parameter has a corresponding nested Q plug-in controller. Various control objectives are discussed. It is shown that the iterated and nested (Q, S) design framework is capable of achieving optimal control performance in a staged way.

In Chapter 6, a direct adaptive-Q method is presented. The premise is that the plant dynamics are well known but that the control performance of the nominal controller needs refinement. The adaptive method is capable of achieving optimal performance. It is shown that under very reasonable conditions the adaptive scheme improves the performance of the nominal controller. Particular control objectives we pay attention to are disturbance rejection and (unknown) reference tracking. The main difference from classical adaptive methods is that we assume from the outset that a stabilizing controller is available. The adaptive mechanism is only included for performance improvement. The adaptively controlled loop is analyzed using averaging techniques, exploiting the observation that the adaptation proceeds slowly as compared to the actual plant dynamics.

The direct adaptive scheme adjusts only a plug-in controller Q. This scheme can not handle significant model mismatch. To overcome this problem an indirect adaptive-Q method is introduced in Chapter 7. This method is an adaptive version of the nested (Q, S) design framework. An estimate for S is obtained on line. On the basis of this estimate we compute Q. The analysis is again performed using time scale separation ideas. The necessary tools for this are concisely developed in Appendix C.

In Chapter 8, the direct adaptive-Q scheme is applied for optimal control of nonlinear systems by means of *linearization techniques*. The idea is that in the real world setting these closed-loop controllers should achieve as close as possible the performance of an optimal open-loop control for the nominal plant. The concept of a *learning-Q* scheme is developed.

In Chapter 9, real-time controller implementation aspects are discussed, including the various hardware and software options for a controller designer. The role of the ubiquitous personal computer, *digital signal processing chip* and microcontrollers is discussed, along with the high level design and *simulation languages* and low level implementation languages.

In Chapter 10, some laboratory case studies are introduced. First, a *disk drive control system* is studied where sampling rates are high and compromises must be made on the complexity of the controller applied. Next, the control of a *heat exchanger* is studied. Since speed is not a critical factor, sampling rates can be low and the control algorithm design can be quite sophisticated. In a third simulation study, we apply the adaptive techniques developed to the model of a current commercial aircraft and show the potential for performance enhancement of a *flight control* system.

Finally, in the appendices, background results in *linear algebra*, *probability theory* and *averaging theory* are summarized briefly, and some useful computer programs are included.

1.6 Study Guide

The most recent and most fascinating results in the book are those of the last chapters concerning adaptive-Q schemes. Some readers with a graduate level background in control theory can go straight to these chapters. Other readers will need to build up to this material chapter by chapter. Certainly, the theory of robust linear control and adaptive control is not fully developed in the earlier chapters, since this is adequately covered elsewhere, but only the very relevant results summarized in the form of a user's guide.

Thus it is that advanced students could cover the book in a one semester course, whereas beginning graduate students may require longer, particularly if they wish to master robust and optimal control theory from other sources as well. Also, as an aid to the beginning student, some of the more technical sections are starred to indicate that the material may be omitted on first reading.

1.7 Main Points of Chapter

High performance control in the real world is our agenda. It goes beyond classical control, optimal control, robust control and adaptive control by blending the strengths of each. With today's software and hardware capabilities, there is a chance to realize significant performance gains using the tools of high performance control.

1.8 Notes and References

For a modern textbook treatment of classical control, we recommend Ogata (1990) and also Doyle, Francis and Tannenbaum (1992). A development of linear quadratic Gaussian control is given in Anderson and Moore (1989), and Kwakernaak and Sivan (1972). Robust control methods are studied in Green and Limebeer (1994) and Morari and Zafiriou (1989). For controller designs based on a factorization approach and optimizing over the class of stabilizing controllers, see Boyd and Barratt (1991) and Vidyasagar (1985). Adaptive control methods are studied in Mareels and Polderman (1996), Goodwin and Sin (1984) and Anderson et al. (1986). References to seminal material for the text, found only in papers, are given in the relevant chapters.

CHAPTER 2

Stabilizing Controllers

2.1 Introduction

In this chapter, our focus is on plant and controller descriptions. The minimum requirement is that any practical controller stabilizes or maintains the stability of the plant. We set the stage with various mathematical representations, also termed models, for the plants to be controlled. Our prime focus is on discrete-time, linear, finite-dimensional, dynamical system representations in terms of state space equations and (matrix) transfer functions; actually, virtually all of the results carry over to continuous time, and indeed time-varying systems as discussed in Chapter 8. A block partition notation for the system representations is introduced which allows for ready association of the transfer function with the state space description of the plant. It proves convenient to develop dexterity with the block partition notation. Manipulations such as concatenation, inverse, and feedback interconnection of systems are easily expressed using this formalism. Controllers are considered with the same dynamical system representations.

The key result in this chapter is the derivation of the class of all stabilizing linear controllers for a linear, time-invariant plant model. We show that all stabilizing controllers for the plant can be synthesized by conveniently parameterized augmentations to any stabilizing controller, called a nominal controller. The augmentations are parameterized by an arbitrary stable filter which we denote by Q. The class of all stabilizing linear controllers for the plant is generated as the stable (matrix) transfer function of the filter Q spans the class of all stable (matrix) transfer functions.

We next view any stabilizing controller as providing control action from two sources; the original stabilizing controller and the controller augmentations including the stable filter Q. This allows us to think of controller designs in two stages. The first stage is to design a nominal stabilizing controller, most probably from a nominal plant model. This controller needs only to achieve certain limited objectives such as *robust* stability, in that not only is the nominal plant stabilized by the controller, but all plants in a suitably large neighborhood of the nominal plant are also stabilized. This is followed by a second stage design, which aims to enhance the performance of the nominal controller in any particular environment. This is achieved with augmentations including a stable filter Q. This second stage could result in an on-line adaptation.

At center stage for the generation of the class of all stabilizing controllers for a plant are *coprime matrix fraction descriptions* of a dynamical system. These are introduced, and we should note that their nonuniqueness is a key in our development.

In the next section, we introduce a nominal plant model description with focus on a discrete-time linear model, possibly derived from a continuous-time plant, as for example is discussed in Chapter 8. The block partition notation is introduced. In Section 2.3, a definition of stability is introduced and the notion a stabilizing feedback controller is developed to include also controllers involving feedforward control. In Section 2.4, coprime factorizations, and the associated Bezout identity are studied. A method to obtain coprime factors via stabilizing feedback control design is also presented. In Section 2.5, the theory for the class of all stabilizing controllers, parameterized in terms of a stable Q filter, is developed. The case of *two-degree-of-freedom controllers* is covered by the theory. Finally, as in all chapters, the chapter concludes with notes and references on the chapter topics.

2.2 The Nominal Plant Model

In this section, we describe the various plant representations that are used throughout the book as a basis for the design of controllers. A feature of our presentation here is the block partition notation used to represent linear systems. The various elementary operations using the block partition notation are described.

The Plant Model

The control of a plant begins with a modeling of the particular physical process. The models can take various forms. They can range from a simple mathematical model parameterized by a gain and a rise time, used often in the design of simple classical controllers for industrial processes, to sophisticated nonlinear, time-varying, partial differential equation models as used for example in fluid flow models. In this book, we are not concerned with the physical processes and the derivation of the mathematical models for these. This aspect can be found in many excellent books and papers, and the readers are referred to Ogata (1990), Åstrom and Wittenmark (1984), Ljung (1987) and their references for a detailed exposition.

We start with a mathematical description of the plant. We lump the underlying dynamical processes, sensors and actuators together and assume that the mathematical model includes the modeling of all sensors and actuators for the plant. Of

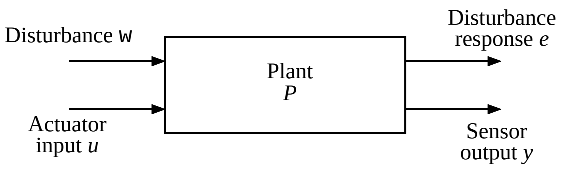


FIGURE 2.1. Plant

course the range of control signals for which the models are relevant should be kept in mind in the design. Actuators have limits on signal magnitudes as well as on rates of change of the control signal. What we have termed the plant is depicted in Figure 2.1. It is drawn as a two-by-two block with input variables w, u and output variables e, y which are functions of time and are vectors in general. Here the block P is an operator mapping the generalized input (w, u) to the generalized output (e, y). At this point in our development, the precise nature of the operator and the signal spaces linked by it are not crucial. More complete descriptions are introduced as we progress. Figure 2.1, in operator notation, is then

$$\begin{bmatrix} e \\ y \end{bmatrix} = [P] \begin{bmatrix} w \\ u \end{bmatrix}, \qquad P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}. \tag{2.1}$$

For these models, u is a variable subject to our control, and is termed the *control input*, and w is a variable not available for control purposes and often consists of disturbances and/or driving signals termed simply *disturbances*. This input w is sometimes referred to as an *exogenous input* or an *auxiliary input*. The *output* variable y is the collection of measurements taken from the sensors. The two-by-two block structure of Figure 2.1 includes an additional disturbance response vector e, also referred to as a derived variable. This vector e is useful in assessing performance. In selecting control signals u, we seek to minimize e in some sense. This response e is not necessarily measured, or measurable, since it may include internal variables of the plant dynamics. The disturbance response e will normally include a list of all the critical signals in the plant. Note that the commonly used arrangement of Figure 2.2 is a special case of Figure 2.1.

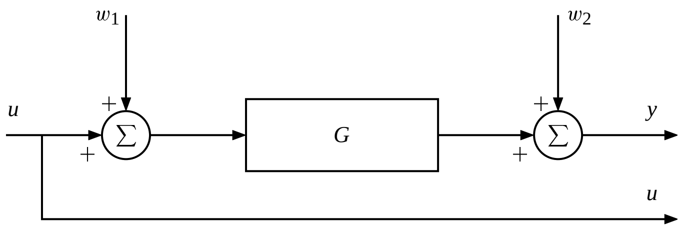


FIGURE 2.2. A useful plant model

$$\begin{bmatrix} e \\ y \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 0 \\ G & I \end{bmatrix} & \begin{bmatrix} I \\ G \end{bmatrix} \\ \begin{bmatrix} G & I \end{bmatrix} & G \end{bmatrix} \begin{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ u \end{bmatrix}. \tag{2.2}$$

Note that $e = [\,u' \; y'\,]'$ in this specialization, and w_1 and w_2 are both disturbance input vectors. Note that $P_{22} = G$.

Discrete-time Linear Model

The essential building block in our plant description is a multiple-input, multiple-output (MIMO) operator. In the case of linear, time invariant, discrete-time systems, denoted by W in operator notation, we use the following state space description:

$$\begin{aligned} W : x_{k+1} &= Ax_k + Bu_k; \qquad x_0, \\ y_k &= Cx_k + Du_k. \end{aligned} \tag{2.3}$$

Here $k \in \mathsf{Z}^+ = \{0, 1, 2, \ldots\}$ indicates sample time, $x_k \in \mathsf{R}^n$ is a state vector with initial value x_0, $u_k \in \mathsf{R}^p$ is the input vector and $y_k \in \mathsf{R}^m$ is the output vector. The coefficients, $A \in \mathsf{R}^{n\times n}$, $B \in \mathsf{R}^{n\times p}$, $C \in \mathsf{R}^{m\times n}$ and $D \in \mathsf{R}^{m\times p}$ are constant matrices. For reference material on matrices, see Appendix A, and on linear dynamical systems, see Appendix B.

It is usual to assume that the pair (A, B) is *controllable,* or at least *stabilizable*, and that the pair (C, A) is *observable* or at least is *detectable*, see definitions in Appendix B. In systems which are not stabilizable or detectable, there are unstable modes which are not controllable and/or observable. In practice, lack of controllability or observability could indicate the need for more actuators and sensors, respectively. Of particular concern is the case of unstable modes or perhaps lightly damped modes that are uncontrollable or unobservable, since these can significantly affect performance. Without loss of generality we assume B to be of full column rank and C full row rank*. In our discussion we will only work with systems showing the properties of detectability and stabilizability.

The transfer matrix function of the block W, can be written as

$$W(z) = C(zI - A)^{-1}B + D. \tag{2.4}$$

Here $W(\infty) = D$ and $W(z)$ is by definition a *proper* transfer function matrix. Thus $W \in R_p$, the class of *rational proper* transfer function matrices. When the coefficient D is a zero matrix, there is no direct feedthrough in the plant. In this case, $W(\infty) = 0$ and so we have $W \in R_{sp}$, the class of rational *strictly proper* transfer function matrices.

The transformation from the state space description to the matrix transfer function description is unique and is given by (2.4). However the transformation from

*A discussion of redundancy at input actuators or output sensors is outside the scope of this text the transfer function description to the state space description is not unique. The various transformations can be found in many references. The readers are referred to Kailath (1980), Chen (1984), Wolovich (1977) and Appendix B.

Block Partition Notation

Having introduced the basic building block, in the context of linear MIMO systems, we now concentrate on how to interconnect different blocks, as suggested in Figure 1.1.1.

For algebraic manipulations of systems described by matrix transfer functions or sets of state-space equations, it proves convenient to work with a block partitioned notation which can be easily related to their (matrix) transfer functions as well as their state space realizations. In this subsection, we present the notation and introduce some elementary operations. For definitions of controllability, observability, asymptotic stability and coordinate basis transformation for linear systems, the readers are referred to Appendix B.

Representation

Let us consider a dynamic system W with state space description given in (2.3). The (matrix) transfer function is given by (2.4) and in the block partition notation, the system with state equations (2.3) is written as

$$W : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]. \tag{2.5}$$

The two solid lines within the square brackets are used to demarcate the (A, B, C, D) matrices, or more generally the $(1, 1)$, $(1, 2)$, $(2, 1)$ and $(2, 2)$ subblocks. In this notation, the number of input and output variables are given by the number of columns and rows of the $(2, 2)$ subblock, respectively. In the case where (A, B) is controllable and (A, C) is observable, the representation of the system (2.5) is said to be a *minimal representation* and the dimension of the $(1, 1)$ subblock gives the *order* of the system.

Subpartitioning

With block partitioning of A, B, C, D as in the following state equations,

$$\begin{aligned} W : \qquad x_{k+1} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} x_k + \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} u_k, \\ y_k &= \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} x_k + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} u_k, \end{aligned} \tag{2.6}$$

the block partition notation for the system W is

$$W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} : \left[\begin{array}{cc|cc} A_{11} & A_{12} & B_{11} & B_{12} \\ A_{21} & A_{22} & B_{21} & B_{22} \\ \hline C_{11} & C_{12} & D_{11} & D_{12} \\ C_{21} & C_{22} & D_{21} & D_{22} \end{array}\right], \tag{2.7}$$

where W_{ij} denotes the state space equations for the subsystem with input j and output i. By inspection,

$$\begin{aligned} W_{11} &: \left[\begin{array}{cc|c} A_{11} & A_{12} & B_{11} \\ A_{21} & A_{22} & B_{21} \\ \hline C_{11} & C_{12} & D_{11} \end{array}\right], \qquad W_{12} : \left[\begin{array}{cc|c} A_{11} & A_{12} & B_{21} \\ A_{21} & A_{22} & B_{22} \\ \hline C_{11} & C_{12} & D_{12} \end{array}\right], \\ W_{21} &: \left[\begin{array}{cc|c} A_{11} & A_{12} & B_{11} \\ A_{21} & A_{22} & B_{21} \\ \hline C_{21} & C_{22} & D_{21} \end{array}\right], \qquad W_{22} : \left[\begin{array}{cc|c} A_{11} & A_{12} & B_{12} \\ A_{21} & A_{22} & B_{22} \\ \hline C_{21} & C_{22} & D_{22} \end{array}\right], \end{aligned} \tag{2.8}$$

and the associated state space equations of the subsystems W_{ij} can be written down immediately, for example

$$\begin{aligned} W_{12} : \qquad x_{k+1} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} x_k + \begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix} u_{2k}, \\ y_{1k} &= \begin{bmatrix} C_{11} & C_{12} \end{bmatrix} x_k + D_{12} u_{2k}. \end{aligned}$$

These state space realizations are not necessarily minimal.

Sums

Let us consider the parallel connection of two systems with the same input and output dimensions. For two systems, W_1 and W_2 given as

$$W_1 : \left[\begin{array}{c|c} A_1 & B_1 \\ \hline C_1 & D_1 \end{array}\right], \qquad W_2 : \left[\begin{array}{c|c} A_2 & B_2 \\ \hline C_2 & D_2 \end{array}\right], \tag{2.9}$$

their sum (parallel connection) in block partition notation is given by:

$$W_1 + W_2 : \left[\begin{array}{cc|c} A_1 & 0 & B_1 \\ 0 & A_2 & B_2 \\ \hline C_1 & C_2 & D_1 + D_2 \end{array}\right]. \tag{2.10}$$

This can be checked by examining the state-space model of the parallel connection. The order of the sum of the (matrix) transfer functions is the sum of the orders of each transfer function, in general. However in certain cases, as when $A_1 = A_2$, we can eliminate uncontrollable or unobservable modes in the parallel connection to achieve a reduced order minimal realization. (See problems at the end of the chapter for some discussion.)

Multiplication

Consider again two systems as given in (2.9), assuming that the output dimension of W_2 equals the input dimension of W_1. The product, or series connection, of the two systems W_1 and W_2 (W_1 after W_2) may be represented as:

$$W_1W_2 : \left[\begin{array}{cc|c} A_1 & B_1C_2 & B_1D_2 \\ 0 & A_2 & B_2 \\ \hline C_1 & D_1C_2 & D_1D_2 \end{array}\right]. \tag{2.11}$$

Again this representation is not minimal when there are pole/zero cancellations in W_1W_2, although it is minimal in the generic case; that is when A_1, A_2, B_1, B_2, C_1, C_2, D_1, D_2 have no special structure or element values. (Notice that $W_1W_2 \neq W_2W_1$. Also, even when W_1W_2 is defined, W_2W_1, may not be!).

Inverse

It is readily checked that under the condition that D_1^{-1} exists, the inverse of the system W_1 exists and is proper and is given by

$$W_1^{-1} : \left[\begin{array}{c|c} A_1 - B_1D_1^{-1}C_1 & -B_1D_1^{-1} \\ \hline D_1^{-1}C_1 & D_1^{-1} \end{array}\right]. \tag{2.12}$$

Notice that

$$W_1^{-1}W_1 = W_1W_1^{-1} : \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}. \tag{2.13}$$

Transpose

The transpose of the system W_1 is given by

$$W_1' : \left[\begin{array}{c|c} A_1' & C_1' \\ \hline B_1' & D_1' \end{array}\right]. \tag{2.14}$$

Stable Uncontrollable Modes

Consider the following system W with its associated state-space realization:

$$W : \left[\begin{array}{cc|c} A_{11} & A_{12} & B_1 \\ 0 & A_{22} & 0 \\ \hline C_1 & C_2 & D \end{array}\right], \tag{2.15}$$

where the eigenvalues of A_{22} are inside the unit circle, that is $|\lambda_i(A_{22})| < 1$, and so the modes associated with A_{22} are asymptotically stable. Clearly for the system W with its associated state-space realization, the states associated with A_{22} are not controllable from the inputs. Moreover, since the modes associated with A_{22} are asymptotically stable, the associated states decay exponentially to zero from arbitrary initial values. From a control point of view these modes may be ignored, simplifying the W representation to:

$$W_s : \left[\begin{array}{c|c} A_{11} & B_1 \\ \hline C_1 & D \end{array}\right]. \tag{2.16}$$

Stable Unobservable Modes

The "dual" case to the above is where there are stable unobservable modes. Consider the system with associated state-space realization given by

$$W : \left[\begin{array}{cc|c} A_{11} & 0 & B_1 \\ A_{21} & A_{22} & B_2 \\ \hline C_1 & 0 & D \end{array}\right], \tag{2.17}$$

where again $|\lambda_i(A_{22})| < 1$ for all i. In this case, the states associated with A_{22} can be excited from the inputs but they do not affect the states associated with A_{11} or the system output. These states are therefore not observable and if ignored allow the simpler representation.

$$W_s : \left[\begin{array}{c|c} A_{11} & B_1 \\ \hline C_1 & D \end{array}\right]. \tag{2.18}$$

The above simplifications are allowed from the control point of view. Obviously these unobservable/uncontrollable models do affect the internal system behavior and the transient behavior.

Coordinate Basis Transformation

For a linear system W of (2.5), let us consider the effect of transformations of the associated state-space variables. Thus consider transformations of the state vector x, input vector u and output vector y by nonsingular matrices T, S and R as follows.

$$\bar{x} = Tx, \qquad \bar{u} = Su, \qquad \bar{y} = Ry. \tag{2.19}$$

The system from $\bar{u}$ to $\bar{y}$, denoted $\bar{W}$, is then given by

$$\bar{W} : \left[\begin{array}{c|c} TAT^{-1} & TBS^{-1} \\ \hline RCT^{-1} & RDS^{-1} \end{array}\right]. \tag{2.20}$$

On many occasions in the book, cascade or parallel connections of subsystems using formulas given in (2.10) or (2.11) give rise to high dimensional systems W with stable uncontrollable or unobservable modes in their state-space realizations. The associated system descriptions can then be simplified by removing (or ignoring) such modes as in deriving W_S. That is, it is possible to find a transformation matrix, T (with $R = I$ and $S = I$) such that the transformed (matrix) transfer function is of the form (2.15) with stable uncontrollable modes, or (2.17) with stable unobservable modes. The stable uncontrollable or unobservable modes can then be removed by the appropriate line and column deletion to achieve the simpler system description W_S.

In fact, a number of the manipulations in our controller design approaches yield system descriptions that can be simplified by one of the following very simple transformations T.

$$\begin{bmatrix} I & I \\ 0 & I \end{bmatrix}, \quad \begin{bmatrix} I & -I \\ 0 & I \end{bmatrix}, \quad \begin{bmatrix} I & 0 \\ I & I \end{bmatrix}, \quad \begin{bmatrix} I & 0 \\ -I & I \end{bmatrix}. \tag{2.21}$$

There are short cuts when dealing with these simple transformations. The transformations can be performed using operations similar to the elementary rows and columns operations in the solution of a set of linear equations using the Gaussian elimination method. Let us work with an example.

Example. Consider the system W given as

$$W : \left[\begin{array}{cc|c} A_{11} & A_{12} & B_1 \\ A_{21} & A_{22} & B_2 \\ \hline C_1 & C_2 & D \end{array}\right], \tag{2.22}$$

where A_{11}, A_{12}, A_{21} and A_{22} are of the same dimension. Under the transformations $R = I,\ S = I$ and

$$T = \begin{bmatrix} I & I \\ 0 & I \end{bmatrix}, \qquad T^{-1} = \begin{bmatrix} I & -I \\ 0 & I \end{bmatrix}, \tag{2.23}$$

the system W can be represented in the transformed coordinate basis as

$$W : \left[\begin{array}{cc|c} A_{11} + A_{21} & A_{12} + A_{22} - A_{11} - A_{21} & B_1 + B_2 \\ A_{21} & A_{22} - A_{21} & B_2 \\ \hline C_1 & C_2 - C_1 & D \end{array}\right]. \tag{2.24}$$

The transformed system description is obtained as follows. First ignore the vertical and horizontal partitioning lines and view W as a 3 block by 3 block structure. Now add *block row 2* of the array to *block row 1* of the array to get the intermediate structure:

$$\begin{array}{cc|c} A_{11} + A_{21} & A_{12} + A_{22} & B_1 + B_2 \\ A_{21} & A_{22} & B_2 \\ \hline C_1 & C_2 & D \end{array}.$$

Leaving *block column 1* of the above intermediate array unchanged, we then subtract *block column 1* from the *block column 2* to get the final transformed system description of (2.24). In fact, we can generalize the procedure as follows.

For a system description with p inputs and m outputs expressed in the block partition form such that the A matrix of the associated state-space realization is an $(n \times n)$ matrix, then the input/output property of the system W is not changed by the following operations. Interpreting the block partition notation representation of the system as an $(n + m) \times (n + p)$ array, add (subtract) the ith row to the jth row of the array where $1 \leq i, j \leq n$, leaving the ith row unchanged, to give an intermediate array, and then follow by subtracting (adding) the jth column from the ith column of the intermediate array, leaving the jth column unchanged.

Main Points of Section

In this section, a two port operator description for the plant is introduced. Plant transfer function matrix and state space representations are given in the context of linear, time-invariant, discrete-time systems. A shorthand block partition notation is presented which allows ready manipulation of subsystems, including series and parallel connections and inverses.

2.3 The Stabilizing Controller

In this section, we first work with a controller and model of the plant in a closed-loop feedback control arrangement used for *regulation* of output variables to zero in the presence of disturbances. We then move on to the more general tracking case which has feedforward control as well as feedback control so that there is close tracking of external signals in the presence of disturbances.

The Closed-loop System

Consider the plant model shown in Figure 2.1. Notice that since P is a linear operator it can be block partitioned into four operators as

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}. \tag{3.1}$$

Let us introduce a feedback controller K in a feedback control arrangement as shown in Figure 3.1. The block P generates not only the plant sensor outputs y, but also the signal e, termed the disturbance response, which is used to evaluate the performance of the feedback system. The feedback controller seeks to minimize the disturbance response e in some sense. The feedback controller is sometimes referred to as a *one-degree-of-freedom controller*, in contrast to a *two-degree-of-freedom controller* with additional command inputs discussed subsequently.

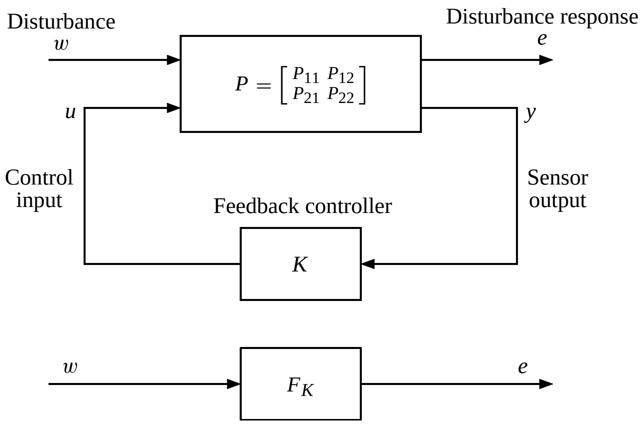


FIGURE 3.1. The closed-loop system

For the system of Figure 3.1, we have the following equations in operator notation:

$$\begin{bmatrix} e \\ y \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} w \\ u \end{bmatrix}, \qquad u = Ky. \tag{3.2}$$

Assuming *well-posedness* of the interconnection, or equivalently, that the relevant inverses exist, the disturbance response is given by:

$$e = F_K w, \qquad F_K = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}. \tag{3.3}$$

A Stabilizing Feedback Controller

Next we consider stability issues. We restrict ourselves in this chapter to the case where all operators correspond to time-invariant, multiple-input multiple-output, discrete-time, finite-dimensional systems. In this context all operators have corresponding proper rational transfer function matrices. In Chapter 7, the case of time-varying systems is seen to follow much of the development here when the transfer functions are generalized to time-varying operators.

The underlying stability concept of interest to us here is *bounded-input, bounded-output stability* (BIBO):

Definition. A system is called BIBO stable if any norm bounded input yields a norm bounded output. When the class of inputs is bounded in an ℓ_2 norm sense and leads to ℓ_2 norm bounded outputs, the system is said to be *BIBO stable in an ℓ_2 sense*. (Recall that ℓ_2 bounded signals in discrete-time are square summable, see also Appendix B.)

In the context of linear systems, BIBO stability in an ℓ_2 sense is equivalent to *small gain stability* and *asymptotic stability* of the state space description. Transfer functions which are BIBO stable in an ℓ_2 sense are said to belong to the H_∞ space and if rational to belong to the rational H_∞ space, denoted RH_∞. For further technical details on these spaces and stability issues, see Appendix B. Suffice it to say here: $H(z) \in RH_\infty$ if and only if every element of $H(z)$ is rational and has no pole in $|z| \geq 1$.

As a starting point, we use the representation as in Figure 3.2 to discuss stability.

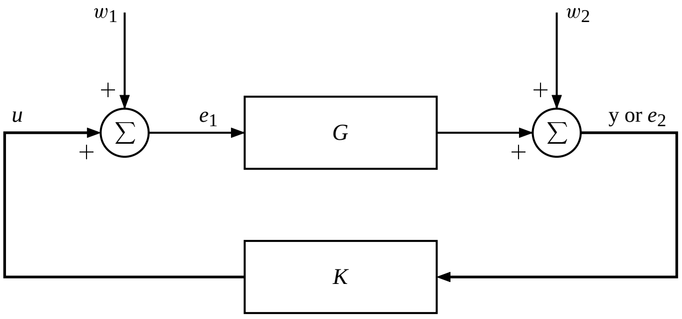


FIGURE 3.2. A stabilizing feedback controller

Let us consider the feedback control loop of Figure 3.2 with a feedback controller $K \in R_p$ applied to a plant $G \in R_p$. Noting that

$$\begin{bmatrix} I & -K \\ -G & I \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},$$

then we have under well-posedness, or equivalently assuming that the inverse exists,

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}. \tag{3.4}$$

Closed-loop stability, also termed *internal stability*, is defined as BIBO stability in an ℓ_2 sense, in that any bounded input to the closed-loop system will give rise to bounded-output signals everywhere within the loop. Here for linear, time-invariant systems, internal stability is identical to asymptotic stability of the state space description of the closed loop. Applying these notions to (3.4) leads to the following result.

Theorem 3.1. *A necessary and sufficient condition to ensure internal stability of the feedback control loop of Figure 3.2 is that*

$$\begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} \in RH_\infty. \tag{3.5}$$

Equivalently,

$$\begin{bmatrix} (I-KG)^{-1} & K(I-GK)^{-1} \\ G(I-KG)^{-1} & (I-GK)^{-1} \end{bmatrix} \in RH_\infty. \tag{3.6}$$

Definition. We say that K stabilizes G or (G, K) is a stabilizing pair if (3.5) or equivalently (3.6) holds.

We see that for K to stabilize G, there is a requirement that each of the four (matrix) transfer functions $(I-KG)^{-1}$, $K(I-GK)^{-1}$, $G(I-KG)^{-1}$ and $(I-GK)^{-1}$ be asymptotically stable. For the single-input, single-output case, where G and K are scalar transfer functions, the stability condition (3.6) is equivalent to the stability condition that $(I-GK)^{-1} \in RH_\infty$, and that there be no pole/zero cancellations in the open-loop transfer function KG, or GK.

For the system of Figure 3.1, internal stability will imply stability of the closed-loop (matrix) transfer function F_K of (3.3). Moreover, Theorem 3.1 is readily generalized by working with a rearranged version of Figure 3.1, namely Figure 3.3. Thus apply the results of Theorem 3.1 with G replaced by P and K replaced by $\left[\begin{smallmatrix} 0 & 0 \\ 0 & K \end{smallmatrix}\right]$. This leads to the following.

Theorem 3.2. *A necessary and sufficient condition to ensure internal stability of the closed-loop feedback control system of Figure 3.1, or equivalently Figure 3.3, is that*

$$\begin{bmatrix} I & -\begin{bmatrix} 0 & 0 \\ 0 & K \end{bmatrix} \\ -P & I \end{bmatrix}^{-1} \in RH_\infty, \tag{3.7}$$

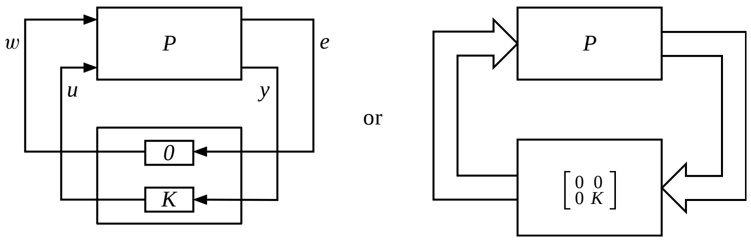


FIGURE 3.3. A rearrangement of Figure 3.1

or equivalently, with a partitioning of P as in (2.1)

$$\begin{bmatrix} I & -K \\ -P_{22} & I \end{bmatrix}^{-1} \in RH_\infty, \qquad P_{12}(I - KP_{22})^{-1} \begin{bmatrix} I & K \end{bmatrix} \in RH_\infty,$$

$$P \begin{bmatrix} I \\ K \end{bmatrix} (I - P_{22}K)^{-1} P_{21} \in RH_\infty. \tag{3.8}$$

We remark that a necessary condition for stability is that the pair (P_{22}, K) is stabilizing. To connect to earlier results we set $P_{22} = G$ so that this condition is the familiar condition that (G, K) is stabilizing. If in addition, $P_{11} \in RH_\infty$, and P_{21}, P_{12} belong to RH_∞ or P_{21} and/or P_{12} are identical to $G = P_{22}$, then this condition is also sufficient for stability.

A Stabilizing Feedforward/Feedback Controller

Consider the *feedforward/feedback controller* arrangement of Figure 3.4. Notice that in this arrangement, both the plant output y, and a reference signal d, possibly some desired output trajectory, are used to generate the control signal. Consider now the rearrangement of the scheme of Figure 3.4 as a feedback controller scheme for an augmented plant depicted in Figure 3.5. The original plant G and disturbance signal w_2 are augmented as follows:

$$G \to \begin{bmatrix} 0 \\ G \end{bmatrix} =: \square, \qquad w_2 \to \begin{bmatrix} d \\ w_2 \end{bmatrix}. \tag{3.9}$$

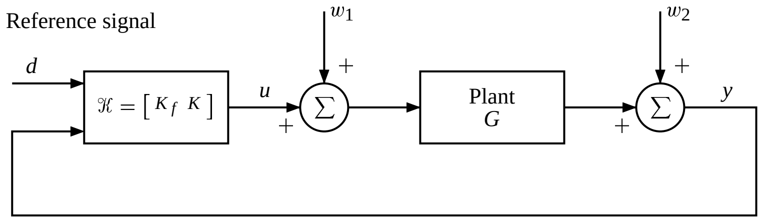
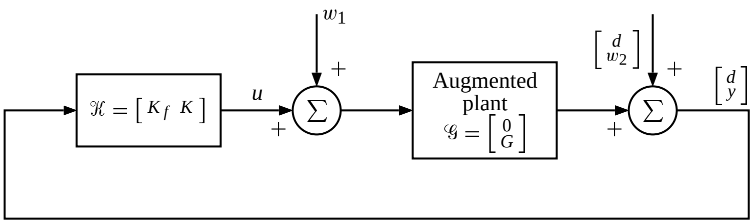


FIGURE 3.4. Feedforward/feedback controller



w_1
$+$
$\mathcal{K} = [\,K_f \;\; K\,]$
u
$+$
Σ
Augmented plant
$\mathcal{G} = \begin{bmatrix} 0 \\ G \end{bmatrix}$
$\begin{bmatrix} d \\ w_2 \end{bmatrix}$
$+$
$+$
Σ
$\begin{bmatrix} d \\ y \end{bmatrix}$

FIGURE 3.5. Feedforward/feedback controller as a feedback controller for an augmented plant

Under this augmentation, the *two-degree-of-freedom controller* for G is identical to the one-degree-of-freedom controller for the augmented plant □. Applying the earlier internal stability results for a one-degree-of-freedom controller to this augmented system leads to internal stability results for the two-degree-of-freedom system. Thus consider a two-degree-of-freedom controller $□ = [\,K_f \;\; K\,] \in R_p$ for the plant $□ \in R_p$. Then the controller □ internally stabilizes □ and thus □ if and only if

$$\begin{bmatrix} I & -□ \\ -□ & I \end{bmatrix}^{-1} = \begin{bmatrix} [I] & -\begin{bmatrix} K_f & K \end{bmatrix} \\ -\begin{bmatrix} 0 \\ G \end{bmatrix} & \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \end{bmatrix}^{-1} \in RH_\infty, \tag{3.10}$$

or equivalently, if and only if

$$\begin{bmatrix} (I-KG)^{-1} & (I-KG)^{-1}K_f & K(I-GK)^{-1} \\ 0 & I & 0 \\ G(I-KG)^{-1} & G(I-KG)^{-1}K_f & (I-GK)^{-1} \end{bmatrix} \in RH_\infty, \tag{3.11}$$

or equivalently, if and only if

$$\begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} \in RH_\infty, \qquad \begin{bmatrix} I \\ G \end{bmatrix} (I-KG)^{-1}K_f \in RH_\infty. \tag{3.12}$$

The first condition tells us that there must be internal stability of the feedback system consisting of G and the feedback controller K. The second condition tells us that any unstable modes of K_f must be contained in K. That is, in the event that K_f is unstable, it should be implemented along with K in the one block $□ = [\,K \;\; K_f\,]$ with a minimal state space representation. If K_f and K are implemented separately, then K_f must be stable. This is a well known result for the stability of two-degrees-of-freedom controllers, see Vidyasagar (1985), but the derivation here is more suited to our approach.

Main Points of Section

In this section we have considered stability properties when applying controllers to plants. Internal stability is linked to the stability of certain (matrix) transfer functions associated with the feedback loops. Both one-degree-of-freedom and two-degrees-of-freedom controller structures are examined. Conditions ensuring stability are identified.

2.4 Coprime Factorization

An important step towards the next section's objective of characterizing the class of all stabilizing controllers is the *coprime factorization* of the plant model and controller. For scalar models and controllers, factorization leads to the models and controllers being represented as the ratio of two stable transfer functions. This factorization is termed coprime when the two transfer functions have no common zeros in $|z| > 1$. Coprimeness excludes unstable pole/zero cancellations in the fractional representation. In the multivariable case, the plant model and nominal controller (matrix) transfer functions are factored into the product of a stable (matrix) transfer function and a (matrix) transfer function with a stable inverse. Coprimeness can be expressed as a full rank condition on the matrices in $|z| > 1$, see below. In this section, we discuss various ways to achieve coprime factorizations.

The Bezout Identity

Let us denote stable, coprime factorizations for the plant $G(z) \in R_p$ of (2.3) and a nominal controller $K(z) \in R_p$ as follows.

$$G = NM^{-1} = \tilde{M}^{-1}\tilde{N}; \qquad N, M, \tilde{N}, \tilde{M} \in RH_\infty, \tag{4.1}$$

$$K = UV^{-1} = \tilde{V}^{-1}\tilde{U}; \qquad U, V, \tilde{U}, \tilde{V} \in RH_\infty. \tag{4.2}$$

It turns out that there exist such factorizations for any plant and controller. We defer until later in this section the question of existence and construction of such factorizations in our setting. *Coprimeness* of the factors N and M means that $\begin{bmatrix} M \\ N \end{bmatrix}$ has full column rank in $|z| > 1$, or equivalently, that its left inverse exists in RH_∞. Coprimeness of $\tilde{M}$, $\tilde{N}$ requires correspondingly that $[\, \tilde{M} \;\; \tilde{N} \,]$ has full rank in $|z| > 1$, or equivalently, has a right inverse in RH_∞.

To illustrate the idea, consider the more familiar setting of scalar transfer functions. Let

$$G(z) = \frac{b(z)}{a(z)}, \tag{4.3}$$

where $b(z)$, $a(z)$ are polynomials in z. Assume that a has degree n, b is of degree less than or equal to n, and that a is monic in that the coefficient of z^n is unity.

Also, assume that $a(z)$, $b(z)$ have no common zeros. A coprime factorization as in (4.1) is then given as

$$G(z) = \left(\frac{b(z)}{z^n}\right)\left(\frac{a(z)}{z^n}\right)^{-1}. \tag{4.4}$$

By construction $b(z)/z^n,\ a(z)/z^n \in RH_\infty$.

In our setting here, we restrict attention to the situation where K stabilizes G and later give existence and constructive procedures for achieving the desired factorizations. First, we study in our context a set of equations known as the *Bezout* or *Diophantine* equations which are associated with coprimeness properties.

As a first step, examine the four closed-loop (matrix) transfer functions of (3.6), under the equalities of (4.1) and (4.2).

It is straightforward to see that

$$\begin{aligned}(I - KG)^{-1} &= (I - \tilde{V}^{-1}\tilde{U}NM^{-1})^{-1}\\ &= M(\tilde{V}M - \tilde{U}N)^{-1}\tilde{V},\end{aligned} \tag{4.5}$$

$$G(I - KG)^{-1} = N(\tilde{V}M - \tilde{U}N)^{-1}\tilde{V}, \tag{4.6}$$

Also, since $(I - KG)K = K(I - GK)$, then

$$\begin{aligned}K(I - GK)^{-1} &= (I - KG)^{-1}K\\ &= M(\tilde{V}M - \tilde{U}N)^{-1}\tilde{U},\end{aligned} \tag{4.7}$$

and

$$\begin{aligned}(I - GK)^{-1} &= I + GK(I - GK)^{-1}\\ &= I + G(I - KG)^{-1}K\\ &= I + N(\tilde{V}M - \tilde{U}N)^{-1}\tilde{U}.\end{aligned} \tag{4.8}$$

Thus

$$\begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} = \begin{bmatrix} M \\ N \end{bmatrix}(\tilde{V}M - \tilde{U}N)^{-1}\begin{bmatrix} \tilde{V} & \tilde{U} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}. \tag{4.9}$$

This result leads to the following lemma.

Lemma 4.1. *Consider the plant $G \in R_p$, and controller $K \in R_p$. Then K stabilizes G if and only if there exist coprime factorizations $G = NM^{-1} = \tilde{M}^{-1}\tilde{N}$, $K = UV^{-1} = \tilde{V}^{-1}\tilde{U}$ with $N, M, \tilde{N}, \tilde{M}, U, V, \tilde{U}, \tilde{V} \in RH_\infty$ such that either of the following* Bezout (Diophantine) *equations hold,*

$$\tilde{V}M - \tilde{U}N = I, \tag{4.10}$$

$$\tilde{M}V - \tilde{N}U = I, \tag{4.11}$$

or equivalently, under the lemma conditions the following double Bezout equation *holds*

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} M & U \\ N & V \end{bmatrix} = \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{4.12}$$

Proof. It is clear that if the stable pairs M, N and $\tilde{U}$, $\tilde{V}$ satisfy the Bezout equation (4.10), the right hand side of (4.9) is stable. This implies that the left hand side of (4.9) is stable, or equivalently, K stabilizes G. Conversely, if K stabilizes G, the left hand side of (4.9) is stable. Because M, N and $\tilde{U}$, $\tilde{V}$ are coprime there exists a left inverse for $\left[\begin{smallmatrix} M \\ N \end{smallmatrix}\right]$ and a right inverse for $[\,\tilde{V}\ \tilde{U}\,]$ in RH_∞. Hence we obtain the result that

$$(\tilde{V}M - \tilde{U}N)^{-1} = Z \in RH_\infty. \tag{4.13}$$

Now, define a new coprime factorization for $G : G = (NZ)(MZ)^{-1}$. Then (4.10) follows.

By interchanging the role G and K in the above arguments so that we view G as stabilizing K, we are led to the alternative dual condition to (4.10), namely (4.11). □

Corollary 4.2. *With* (4.1), (4.2) *holding, necessary and sufficient conditions for the pair* (G, K) *to be stabilizing are that*

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix}^{-1} \in RH_\infty, \quad or \quad \begin{bmatrix} M & U \\ N & V \end{bmatrix}^{-1} \in RH_\infty. \tag{4.14}$$

These conditions are equivalent to (3.5), (3.6).

The above corollary is transparent noting that (4.12) implies (4.14), and (4.14) leads to

$$\begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} = \begin{bmatrix} M & 0 \\ 0 & V \end{bmatrix} \begin{bmatrix} M & U \\ N & V \end{bmatrix}^{-1} \in RH_\infty,$$

or equivalently, that (G, K) is a stabilizing pair.

Normalized Coprime Factors

Normalized coprime factors (M, N) or $\left(\tilde{M}, \tilde{N}\right)$ are left or right coprime factors with the special normalization property

$$\begin{aligned} N^{-}N + M^{-}M &= I, \\ \tilde{N}\tilde{N}^{-} + \tilde{M}\tilde{M}^{-} &= I, \end{aligned} \tag{4.15}$$

for all $|z| = 1$. Here $M^-(z)$ denotes $M'(z^{-1})$ etc. Our developments do not feature these factorizations with their interesting properties as do those of McFarlane and Glover (1989). We note in passing that these factorizations may be obtained in principle from any coprime factors by a *spectral factorization*. Thus for $\tilde{N}\tilde{N}^- + \tilde{M}\tilde{M}^- = \tilde{Z}\tilde{Z}^-$ with $\tilde{Z}, \tilde{Z}^{-1} \in RH_\infty$ we have normalized left coprime factors $\big(\tilde{Z}^{-1}\tilde{N}, \tilde{Z}^{-1}\tilde{M}\big)$. Likewise for $N^-N + M^-M = Z^-Z$ with $Z, Z^{-1} \in RH_\infty$ we have normalized right coprime factors $\big(NZ^{-1}, MZ^{-1}\big)$.

State Estimate Feedback Controllers

Let us construct stable, coprime factorizations for a plant model and a special class of controllers known as state estimate feedback controllers. Consider a plant G with a state space description given as follows.

$$G : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]. \tag{4.16}$$

Under a *stabilizability* assumption on the pair (A, B), it is possible using standard methods to construct a constant *stabilizing state feedback gain* (matrix) F, in that $(A + BF)$ has all eigenvalues within the unit circle. Likewise, under a *detectability* assumption on the pair (A, C), it is possible to construct a constant stabilizing *output injection* (matrix) H, in that $(A + HC)$ has all eigenvalues within the unit circle. The gains F and H can be obtained from various methodologies for controller designs. Examples are the so-called linear-quadratic-Gaussian (LQG) methods, see Anderson and Moore (1989), Kwakernaak and Sivan (1972) or eigenvalues assignment approaches in Ogata (1990). The stabilizing controller K with its associated state space is then given by

$$K : \left[\begin{array}{c|c} A + BF + HC + HDF & -H \\ \hline F & 0 \end{array}\right]. \tag{4.17}$$

The plant/controller arrangement is depicted in Figure 4.1. Stable coprime factorizations for $G(z)$ and $K(z)$ are then given as follows.

$$\begin{bmatrix} M & U \\ N & V \end{bmatrix} : \left[\begin{array}{c|cc} A + BF & B & -H \\ \hline F & I & 0 \\ C + DF & D & I \end{array}\right], \tag{4.18}$$

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} : \left[\begin{array}{c|cc} A + HC & -(B + HD) & H \\ \hline F & I & 0 \\ C & -D & I \end{array}\right]. \tag{4.19}$$

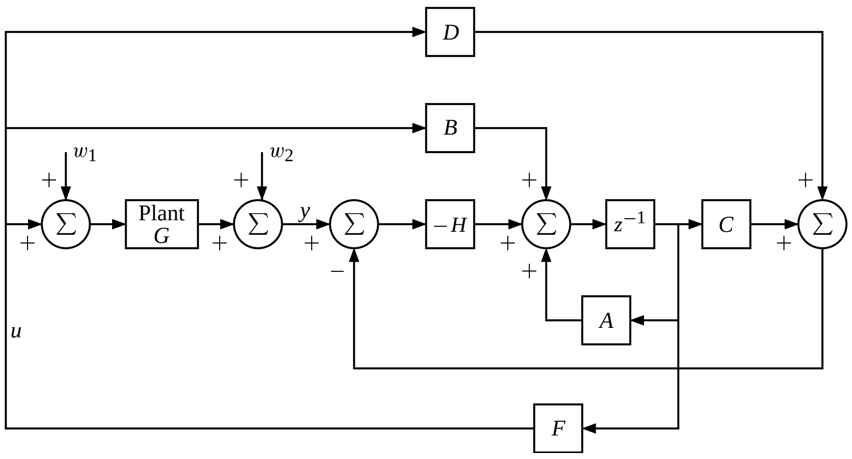


FIGURE 4.1. State estimate feedback controller

To verify that NM^{-1} is indeed a factorization for G, proceed as follows:

$$M : \left[\begin{array}{c|c} A+BF & B \\ \hline F & I \end{array}\right], \qquad N : \left[\begin{array}{c|c} A+BF & B \\ \hline C+DF & D \end{array}\right].$$

Here M^{-1} obviously exists, and has a representation

$$M^{-1} : \left[\begin{array}{c|c} A+BF-BF & -B \\ \hline F & I \end{array}\right] = \left[\begin{array}{c|c} A & -B \\ \hline F & I \end{array}\right].$$

We have then for NM^{-1}:

$$NM^{-1} : \left[\begin{array}{cc|c} A+BF & BF & B \\ 0 & A & -B \\ \hline C+DF & DF & D \end{array}\right].$$

Using the state space transformation $T = \left[\begin{smallmatrix} I & I \\ 0 & I \end{smallmatrix}\right]$ which leaves NM^{-1} unchanged, we find

$$NM^{-1} : \left[\begin{array}{cc|c} A+BF & 0 & 0 \\ 0 & A & -B \\ \hline C+DF & -C & D \end{array}\right].$$

Hence after removing stable, uncontrollable modes and changing the signs of the input and output matrices, we have:

$$NM^{-1} : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] : G.$$

In a similar way, we can show that UV^{-1} is a factorization of K.

These factorizations as presented by (4.18), (4.19) satisfy the double Bezout equations. This may be verified by a now familiar sequence of steps as follows:

$$
\left[\begin{array}{c|cc} A+BF & B & -H \\ \hline F & I & 0 \\ C+DF & D & I \end{array}\right]
\left[\begin{array}{c|cc} A+HC & -(B+HD) & H \\ \hline F & I & 0 \\ C & -D & I \end{array}\right]
$$

$$
= \left[\begin{array}{cc|cc} A+BF & BF-HC & (B+HD) & -H \\ 0 & A+HC & -(B+HD) & H \\ \hline F & F & I & 0 \\ C+DF & C+DF & 0 & I \end{array}\right]
$$

$$
= \left[\begin{array}{cc|cc} A+BF & 0 & 0 & 0 \\ 0 & A+HC & -(B+HD) & H \\ \hline F & 0 & I & 0 \\ C+DF & 0 & 0 & I \end{array}\right]
$$

$$
: \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}.
$$

Notice that the second equality is obtained via the block state transformation $T = \left[\begin{smallmatrix} I & I \\ 0 & I \end{smallmatrix}\right]$, with identity input and output transformations. The third equality is obtained by removing uncontrollable and unobservable modes.

It turns out that special selections of stabilizing F and H yield the *normalized coprime factors* satisfying (4.16). More on this in Chapter 4.

More General Stabilizing Controllers[†]

We consider the coprime factorizations with respect to an arbitrary stabilizing controller K. Let K be given by

$$
K : \left[\begin{array}{c|c} \check{A} & \check{B} \\ \hline \check{C} & \check{D} \end{array}\right], \tag{4.20}
$$

[†]This material, included for completeness, is somewhat condensed and may be merely skimmed on first reading.

with the pairs $\left(\check{A}, \check{B}\right)$ stabilizable and $\left(\check{A}, \check{C}\right)$ detectable. Because (G, K) is stabilizing:

$$\begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} : \left\{ \left[\begin{array}{cc|cc} A & 0 & -B & 0 \\ 0 & \check{A} & 0 & -\check{B} \\ \hline 0 & \check{C} & I & -\check{D} \\ C & 0 & -D & I \end{array} \right] \right\}^{-1}$$
$$: \left[\begin{array}{cc|cc} A + BY\check{D}C & BY\check{C} & BY & BY\check{D} \\ \check{B}ZC & \check{A} + \check{B}ZD\check{C} & \check{B}ZD & \check{B}Z \\ \hline Y\check{D}C & Y\check{C} & Y & Y\check{D} \\ ZC & ZD\check{C} & ZD & Z \end{array} \right] \in RH_\infty. \tag{4.21}$$

where $Y = (I - \check{D}D)^{-1}$ and $Z = (I - D\check{D})^{-1}$.

To perform the coprime factorization, a key step which only becomes obvious in hindsight is to first construct state feedback gains F and $\check{F}$ under stabilizability assumptions on the pairs (A, B) and $\left(\check{A}, \check{B}\right)$ such that $A + BF$ and $\check{A} + \check{B}\check{F}$ have all eigenvalues within the unit circle. These matrix gains are easily obtained by performing a state feedback design for the pairs (A, B) and $\left(\check{A}, \check{B}\right)$, respectively. The coprime factorizations for G and K are then given by

$$\begin{bmatrix} M & U \\ N & V \end{bmatrix} : \left[\begin{array}{cc|cc} A + BF & 0 & B & 0 \\ 0 & \check{A} + \check{B}\check{F} & 0 & \check{B} \\ \hline F & \check{C} + \check{D}\check{F} & I & \check{D} \\ C + DF & \check{F} & D & I \end{array} \right], \tag{4.22}$$

and

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} : \left[\begin{array}{cc|cc} A + BY\check{D}C & BY\check{C} & -BY & BY\check{D} \\ \check{B}ZC & \check{A} + \check{B}ZD\check{C} & -\check{B}ZD & \check{B}Z \\ \hline F - Y\check{D}C & -Y\check{C} & Y & Y\check{D} \\ ZC & -\left(\check{F} - ZD\check{C}\right) & ZD & Z \end{array} \right]. \tag{4.23}$$

Comparing with the closed-loop (matrix) transfer functions (4.21), it is clear that the factorizations here are stable only because of the construction of the state feedback gains F, $\check{F}$. The factorizations can be verified to satisfy the double Bezout equation by direct multiplication; see problems.

Main Points of Section

In this section, we have examined the representation of the (matrix) transfer function of a plant and its stabilizing controller using stable, coprime factors. A key result is that internal stability of the closed-loop system is equivalent to the existence of coprime fractional representations for the plant model and controller that satisfy the Bezout equation. This has been exploited to provide an explicit construction of coprime factorizations of the plant model and a stabilizing state feedback controller. Coprime factorizations associated with other stabilizing controllers can be derived.

2.5 All Stabilizing Feedback Controllers

In this section, the class of all stabilizing controllers for a plant is parameterized in terms of a stable, proper filter, denoted Q.

The Q-Parameterization

Let $K \in R_p$ be a nominal controller which stabilizes a plant $G \in R_p$ and with coprime factorizations given by (4.1), (4.2) satisfying the double Bezout equation (4.12). Consider also the following class of controllers $K(Q)$ parameterized in terms of $Q \in RH_\infty$ and in right factored form:

$$K(Q) := U(Q)V(Q)^{-1}, \tag{5.1}$$

$$U(Q) = U + MQ, \qquad V(Q) = V + NQ; \tag{5.2}$$

or in left factored form:

$$K(Q) := \tilde{V}(Q)^{-1}\tilde{U}(Q), \tag{5.3}$$

$$\tilde{U}(Q) = \tilde{U} + Q\tilde{M}, \qquad \tilde{V}(Q) = \tilde{V} + Q\tilde{N}. \tag{5.4}$$

Then, as simple manipulations show, these factorizations together with the factorizations for G also satisfy a double Bezout equation (4.12), so that $K(Q)$ stabilizes G for all $Q \in RH_\infty$. Equation (5.2) is referred to as a right stable linear fractional representation, while (5.4) is referred to as a left stable linear fractional representation.

We now show the more complete result that with Q spanning the entire class of RH_∞, the entire class of stabilizing controllers for the plant G is generated. Equivalently, any stabilizing controller for the plant can be generated by a particular stabilizing controller and a $Q \in RH_\infty$.

Let us consider the closed-loop system with the controller of (5.2) or (5.4). This closed-loop system can be written in terms of the closed-loop system with the nominal controller and the stable transfer function Q as follows.

$$\begin{aligned}
\begin{bmatrix} I & -K(Q) \\ -G & I \end{bmatrix}^{-1} &= \begin{bmatrix} I & -\tilde{V}(Q)^{-1}\tilde{U}(Q) \\ -\tilde{M}^{-1}\tilde{N} & I \end{bmatrix}^{-1} \\
&= \left\{ \begin{bmatrix} \tilde{V}(Q)^{-1} & 0 \\ 0 & \tilde{M}^{-1} \end{bmatrix} \begin{bmatrix} \tilde{V}(Q) & -\tilde{U}(Q) \\ -\tilde{N} & \tilde{M} \end{bmatrix} \right\}^{-1} \\
&= \begin{bmatrix} M & U(Q) \\ N & V(Q) \end{bmatrix} \begin{bmatrix} \tilde{V}(Q) & 0 \\ 0 & \tilde{M} \end{bmatrix} \\
&= \left\{ \begin{bmatrix} M & U \\ N & V \end{bmatrix} + \begin{bmatrix} 0 & MQ \\ 0 & NQ \end{bmatrix} \right\} \left\{ \begin{bmatrix} \tilde{V} & 0 \\ 0 & \tilde{M} \end{bmatrix} + \begin{bmatrix} Q\tilde{N} & 0 \\ 0 & 0 \end{bmatrix} \right\} \\
&= \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} \tilde{V} & 0 \\ 0 & \tilde{M} \end{bmatrix} + \begin{bmatrix} MQ\tilde{N} & 0 \\ NQ\tilde{N} & 0 \end{bmatrix} + \begin{bmatrix} 0 & MQ\tilde{M} \\ 0 & NQ\tilde{M} \end{bmatrix} \\
&= \begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix}^{-1} \begin{bmatrix} \tilde{V}^{-1} & 0 \\ 0 & \tilde{M}^{-1} \end{bmatrix}^{-1} + \begin{bmatrix} MQ\tilde{N} & MQ\tilde{M} \\ NQ\tilde{N} & NQ\tilde{M} \end{bmatrix}.
\end{aligned} \tag{5.5}$$

Notice that we have:

$$\begin{aligned}
\begin{bmatrix} I & -K(Q) \\ -G & I \end{bmatrix}^{-1} &= \begin{bmatrix} (I-K(Q)G)^{-1} & K\,(I-K(Q)G)^{-1} \\ G\,(I-K(Q)G)^{-1} & \left(I-GK(Q)^{-1}\right) \end{bmatrix} \\
&= \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} \tilde{V} & 0 \\ 0 & \tilde{M} \end{bmatrix} + \begin{bmatrix} M \\ N \end{bmatrix} Q \begin{bmatrix} \tilde{N} & \tilde{M} \end{bmatrix}.
\end{aligned} \tag{5.6}$$

Note that the third and last equalities are obtained by applying the double Bezout equation. We conclude that

$$\begin{bmatrix} I & -K(Q) \\ -G & I \end{bmatrix}^{-1} = \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} + \begin{bmatrix} M \\ N \end{bmatrix} Q \begin{bmatrix} \tilde{N} & \tilde{M} \end{bmatrix}. \tag{5.7}$$

From (5.7), it is again clear that any controller $K(Q)$ parameterized by $Q \in RH_\infty$ as in (5.2) stabilizes the plant G. To see the converse, we note that Q is given as follows.

$$Q = \begin{bmatrix} \tilde{V} & -\tilde{U} \end{bmatrix} \left\{ \begin{bmatrix} I & -K(Q) \\ -G & I \end{bmatrix}^{-1} - \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} \right\} \begin{bmatrix} -U \\ V \end{bmatrix}. \tag{5.8}$$

From (5.8), we note that if both K and $K(Q)$ stabilize G, that Q as given by (5.8) satisfies $Q \in RH_\infty$.

Consider now an arbitrary stabilizing controller for G, denoted K_1. Replacing $K(Q)$ in (5.8) by K_1 allows a calculation of a K_1 dependent Q, denoted Q_1, and associated $U_1 = U + MQ_1$, $V_1 = V + NQ_1$, $\tilde{U}_1 = \tilde{U} + Q_1\tilde{M}$, $\tilde{V}_1 = \tilde{V} + Q_1\tilde{N}$. Now reversing the derivations for (5.5)–(5.8) in the case $K(Q) = K_1$ leads to the first equality in (5.5) holding with $K(Q)$, $\tilde{V}(Q)$, $\tilde{U}(Q)$ replaced by K_1, $\tilde{V}_1$, $\tilde{U}_1$, or equivalently, $K_1 = \tilde{V}_1^{-1}\tilde{U}_1$ (and $G = \tilde{M}^{-1}\tilde{N}$), which allows a construction of an arbitrary stabilizing controller K_1 using our controller structure with Q replaced by Q_1. Consequently, we have established the following theorem.

Theorem 5.1. *Given a plant $G \in R_p$ with coprime factorizations* (4.1), (4.2), *and the controller class $K(Q)$ given from* (5.2), (5.4), *then* (5.7), (5.8) *hold. Moreover, $K(Q)$ stabilizes G if and only if $Q \in RH_\infty$. Furthermore, as Q varies over RH_∞ all possible proper stabilizing controllers for G are generated by $K(Q)$.*

Proof. See above. □

Realizations

We will next look at the realization of the controller $K(Q)$ parameterized in terms of $Q \in RH_\infty$. From (5.2) and the Bezout equation (4.12), we have

$$\begin{aligned} K(Q) &= (U + MQ)(V + NQ)^{-1} \\ &= (U + \tilde{V}^{-1}(I + \tilde{U}N)Q)(V + NQ)^{-1} \\ &= (U + \tilde{V}^{-1}Q + \tilde{V}^{-1}\tilde{U}NQ)(V + NQ)^{-1} \\ &= (UV^{-1}(V + NQ) + \tilde{V}^{-1}Q)(V + NQ)^{-1} \\ &= K + \tilde{V}^{-1}Q(I + V^{-1}NQ)^{-1}V^{-1}. \end{aligned} \tag{5.9}$$

The controller $K(Q)$ as given by (5.9) can be organized into a compact form depicted in Figure 5.1 with J given as

$$J = \begin{bmatrix} UV^{-1} & \tilde{V}^{-1} \\ V^{-1} & -V^{-1}N \end{bmatrix} = \begin{bmatrix} \tilde{V}^{-1}\tilde{U} & \tilde{V}^{-1} \\ V^{-1} & -V^{-1}N \end{bmatrix}, \tag{5.10}$$

and

$$\begin{bmatrix} u \\ r \end{bmatrix} = [J] \begin{bmatrix} y \\ s \end{bmatrix}, \qquad s = Qr. \tag{5.11}$$

With the internal structure of the controller $K(Q)$ shown in Figure 5.2, it is interesting to note that effectively, the control signal generated by $K(Q)$ consists of the signal generated by the nominal controller K and another second control loop involving the stable transfer function Q. When $Q \equiv 0$, then $K(Q) = K$.

For implementation purposes, the Bezout equation of (4.12) can be used to obtain a variant of the block J in Figure 5.2. The reorganized structure is shown

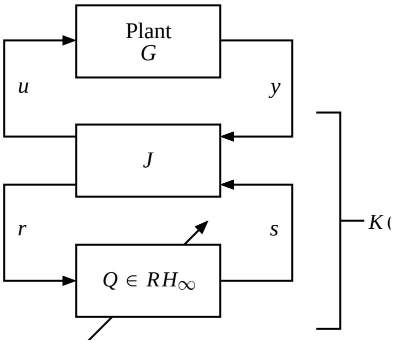


FIGURE 5.1. Class of all stabilizing controllers

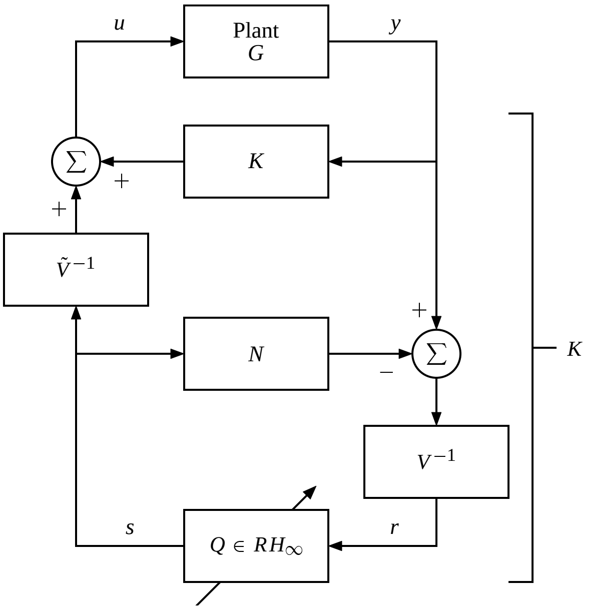


FIGURE 5.2. Class of all stabilizing controllers in terms of factors

in Figure 5.3. With this structure, and as shown below, the signal r is generated by $r = \tilde{M}y - \tilde{N}u$, where $\tilde{M}$ and $\tilde{N}$ are stable filters. Compare to the structure in Figure 5.2, where the signal r is generated using a feedback loop involving N, V^{-1} and Q, with V^{-1} possibly unstable. The structure of Figure 5.3 is more desirable.

From (5.11), we have immediately

$$s = \tilde{V}u - \tilde{U}y. \tag{5.12}$$

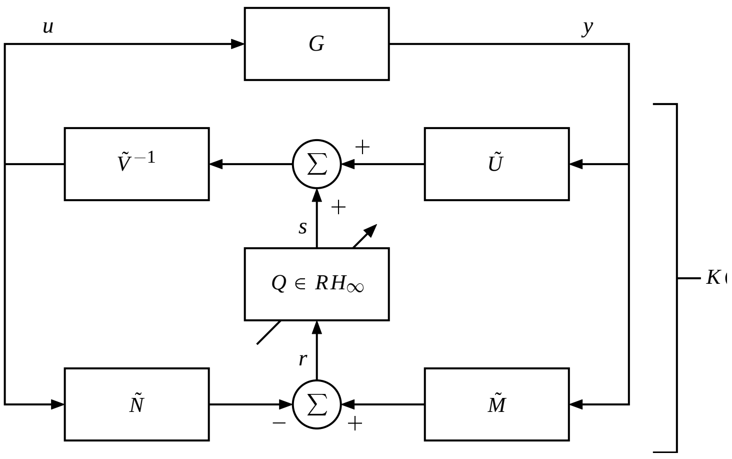


FIGURE 5.3. Reorganization of class of all stabilizing controllers

Using the second equation from (5.11), and (5.12) we have

$$\begin{aligned} r &= V^{-1}y - V^{-1}N\left(\tilde{V}u - \tilde{U}y\right) \\ &= V^{-1}\left(I + N\tilde{U}\right)y - V^{-1}N\tilde{V}u \\ &= V^{-1}\left(V\tilde{M}\right)y - V^{-1}V\tilde{N}u \\ &= \tilde{M}y - \tilde{N}u \end{aligned} \tag{5.13}$$

It may appear at this stage that to implement the controller $K(Q)$ requires augmentations to K for the generation of filtered signals r for driving the Q filter. This is not the case for the state estimate feedback controller structure, as we now show.

In the state space notation of (4.17)–(4.19), as the reader can check (see problems),

$$J : \left[\begin{array}{c|cc} A + BF + HC + HDF & -H & B + HD \\ \hline F & 0 & I \\ -(C + DF) & I & -D \end{array}\right]. \tag{5.14}$$

The implementation of the class of all stabilizing controllers in conjunction with a state estimate feedback nominal controller is shown in Figure 5.4. The input to the Q block is the estimation residual r, and the output s is summed with the state estimate feedback signal to give the final control signal. Clearly in this case, there is no additional computational requirement except for the implementation of the Q filter.

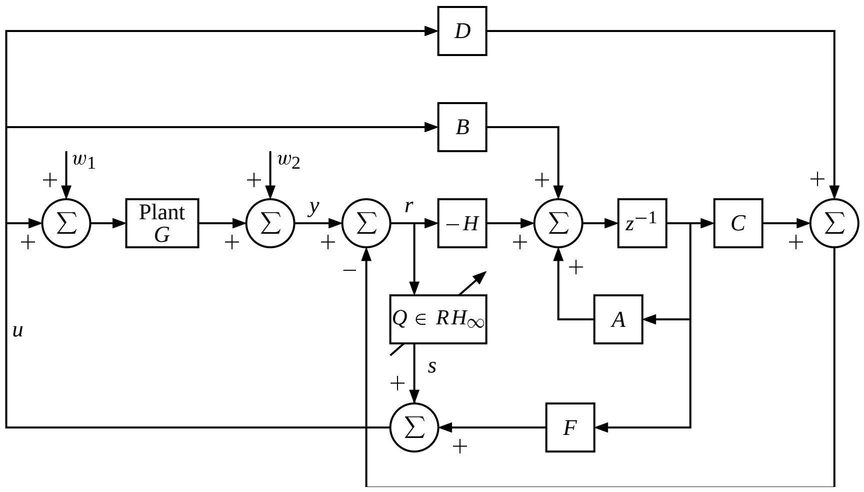


FIGURE 5.4. Class of all stabilizing controllers with state estimates feedback nominal controller

Closed Loop Properties

Let us consider again the plant model P of (2.1) depicted in Figure 2.1 with

$$P_{22} = G. \tag{5.15}$$

We will next look at the closed-loop (matrix) transfer function of the plant model with $K(Q)$ as the feedback controller, implemented as the pair (J, Q) depicted in Figure 5.5. This is a generalization of the arrangement of Figure 5.1. The (matrix) transfer function of interest is that from the disturbance w to the disturbance response e. Referring to Figure 5.5, we will first derive the (matrix) transfer function T, which consists of the plant P block (with $P_{22} = G$) and the J block. The

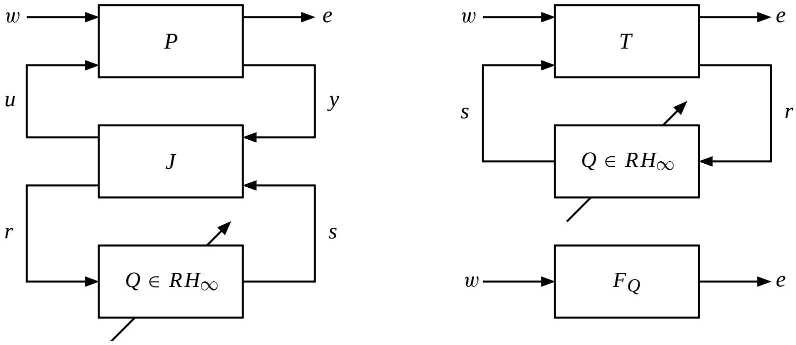


FIGURE 5.5. Closed-loop transfer functions for the class of all stabilizing controllers

various equations are listed as follows, see also (5.10), (5.15).

$$\begin{aligned}
\begin{bmatrix} e \\ y \end{bmatrix} &= \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} w \\ u \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & G \end{bmatrix} \begin{bmatrix} w \\ u \end{bmatrix}, \\
\begin{bmatrix} u \\ r \end{bmatrix} &= \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} y \\ s \end{bmatrix} = \begin{bmatrix} K & \tilde{V}^{-1} \\ V^{-1} & -V^{-1}N \end{bmatrix} \begin{bmatrix} y \\ s \end{bmatrix}, \\
s &= Qr.
\end{aligned} \tag{5.16}$$

We proceed by eliminating the variables y and u using $G = \tilde{M}^{-1}\tilde{N}$ and $K = \tilde{V}^{-1}\tilde{U}$. We have, after a number of manipulations involving the double Bezout equation (4.12), noting a key intermediate result

$$(I - J_{11}P_{22})^{-1} = (I - \tilde{V}^{-1}\tilde{U}NM^{-1})^{-1} = M\tilde{V},$$

that

$$\begin{bmatrix} e \\ r \end{bmatrix} = T \begin{bmatrix} w \\ s \end{bmatrix}; \qquad T = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} = \begin{bmatrix} P_{11} + P_{12}U\tilde{M}P_{21} & P_{12}M \\ \tilde{M}P_{21} & 0 \end{bmatrix}. \tag{5.17}$$

It is interesting to note that $T_{22} = 0$ regardless of the plant or the nominal controller used. A direct result of this property is that the closed-loop (matrix) transfer function from w to e is given by

$$e = F_Q w; \qquad F_Q = T_{11} + T_{12}QT_{21}, \tag{5.18}$$

which is *affine* in the (matrix) transfer function Q. Of course, with T, $Q \in RH_\infty$, then $F_Q \in RH_\infty$. Actually, because of the linearity in Q of the disturbance response (matrix) transfer function F_Q, it is possible to construct convenient optimization procedures to select stable Q to minimize the disturbance response according to reasonable measures. In this way, optimum disturbance rejection controllers are achieved with the search restricted to the class of all stabilizing controllers. This affine nature is also useful for numerical optimization, see Boyd and Barratt (1991). An adaptive version of this observation is presented in Chapter 6.

It is interesting and useful for work in later chapters to examine the state space description for T in the case

$$P : \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right], \tag{5.19}$$

with state estimate feedback controller (4.17). Again we work with $P_{22} = G$, so that connecting with earlier notation $G = NM^{-1} = \tilde{M}^{-1}\tilde{N}$ and $B_2 = B$,

$C_2 = C$, simple manipulations give, under (4.18), (4.19)

$$T_{12} = P_{12}M : \left[\begin{array}{c|c} A + B_2F & B_2 \\ \hline C_1 + D_{12}F & D_{12} \end{array}\right], \tag{5.20}$$

$$T_{21} = \tilde{M}P_{21} : \left[\begin{array}{c|c} A + HC_2 & B_1 + HD_{21} \\ \hline C_2 & D_{21} \end{array}\right]. \tag{5.21}$$

Further manipulations which the reader can check (see problems) give an expression for $T_{11} = P_{11} + P_{12}U\tilde{M}P_{21}$, and since $T_{22} = 0$, there follows

$$T : \left[\begin{array}{cc|cc} A + B_2F & -HC_2 & -HD_{21} & B_2 \\ 0 & A + HC_2 & B_1 + HD_{21} & 0 \\ \hline C_1 + D_{12}F & C_1 & D_{11} & D_{12} \\ 0 & C_2 & D_{21} & 0 \end{array}\right]. \tag{5.22}$$

An interesting special case is when the disturbance and disturbance response enter as in Figure 3.2, so that

$$P = \begin{bmatrix} \begin{bmatrix} G & I \end{bmatrix} & G \\ \begin{bmatrix} G & I \end{bmatrix} & G \end{bmatrix}.$$

Then simple manipulations give

$$T = \begin{bmatrix} \begin{bmatrix} V\tilde{N} & U\tilde{N} \end{bmatrix} & N \\ \begin{bmatrix} \tilde{N} & \tilde{M} \end{bmatrix} & 0 \end{bmatrix}.$$

We now consider other input/output relationships and stability properties. In the first instance, let us look directly at the possible input/output (matrix) transfer functions for the scheme of Figure 5.2, denoted (G, J, Q). Simple manipulations give internal stability conditions as

$$\begin{bmatrix} \begin{bmatrix} I & -K(Q) \\ -G & I \end{bmatrix}^{-1} & \begin{bmatrix} M \\ N \end{bmatrix} \begin{bmatrix} I & Q \end{bmatrix} \\ \begin{bmatrix} I \\ Q \end{bmatrix} \begin{bmatrix} \tilde{N} & \tilde{M} \end{bmatrix} & \begin{bmatrix} I & 0 \\ Q & I \end{bmatrix} \end{bmatrix} \in RH_\infty.$$

Equivalently, $Q \in RH_\infty$. This result tells us that $K(Q)$, when stabilizing for G, can be implemented as the (J, Q) control loop without loss of stability.

It is straightforward to generalize the above stability results, including the results of Theorems 3.1 and 5.1 to cope with the nested controller arrangements of

Figure 5.5, denoted (P, J, Q). Necessary and sufficient conditions for stability are that the pairs

$$\left(P, \begin{bmatrix} 0 & 0 \\ 0 & K(Q) \end{bmatrix}\right), \qquad \left(T, \begin{bmatrix} 0 & 0 \\ 0 & Q \end{bmatrix}\right) \quad \text{are stabilizing.} \tag{5.23}$$

All Stabilizing Feedforward/Feedback Controllers

We have earlier introduced the feedforward/feedback controller as a feedback controller for an augmented plant. We will show in this section that applying the results of the above subsection in this case allows us to generate the class of all stabilizing feedforward/feedback controllers.

Let us recall the augmented plant model of (3.9) and the corresponding nominal feedforward/feedback controller as

$$\Box = \begin{bmatrix} 0 \\ G \end{bmatrix}, \qquad \Box = \begin{bmatrix} K_f & K \end{bmatrix}. \tag{5.24}$$

Consider coprime factorizations for the plant G as in (4.1) and nominal feedback controller K as in (4.2) such that the double Bezout identity (4.12) is satisfied. Then coprime factorizations for the augmented plant $\Box$ and nominal controller $\Box$ are given by

$$\Box = \tilde{\Box}^{-1}\tilde{\Box} = \Box\Box^{-1}, \qquad \Box = \tilde{\Box}^{-1}\tilde{\Box} = \Box\Box^{-1}, \tag{5.25}$$

with rational proper stable factorizations:

$$\begin{aligned}
\Box &= \begin{bmatrix} 0 \\ N \end{bmatrix}, & \tilde{\Box} &= \begin{bmatrix} 0 \\ \tilde{N} \end{bmatrix}, \\
\Box &= M, & \tilde{\Box} &= \begin{bmatrix} I & 0 \\ 0 & \tilde{M} \end{bmatrix}, \\
\Box &= \begin{bmatrix} M\tilde{U}_f & U \end{bmatrix}, & \tilde{\Box} &= \begin{bmatrix} \tilde{U}_f & \tilde{U} \end{bmatrix}, \\
\Box &= \begin{bmatrix} I & 0 \\ N\tilde{U}_f & V \end{bmatrix}, & \tilde{\Box} &= \tilde{V}, \\
\tilde{U}_f &= \tilde{V}K_f.
\end{aligned} \tag{5.26}$$

It is readily checked that these factorizations satisfy the corresponding double Bezout equation

$$\begin{bmatrix} \tilde{\Box} & -\tilde{\Box} \\ -\tilde{\Box} & \tilde{\Box} \end{bmatrix} \begin{bmatrix} \Box & \Box \\ \Box & \Box \end{bmatrix} = \begin{bmatrix} \Box & \Box \\ \Box & \Box \end{bmatrix} \begin{bmatrix} \tilde{\Box} & -\tilde{\Box} \\ -\tilde{\Box} & \tilde{\Box} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{5.27}$$

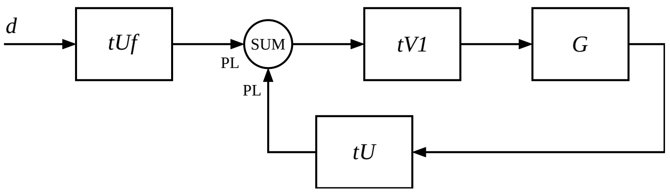


FIGURE 5.6. A stabilizing feedforward/feedback controller

Note that a coprime factorization for K_f is $K_f = \tilde{V}^{-1}\tilde{U}_f$ so that any unstable modes of K_f are necessarily that of unstable modes of K. The relevant block diagram is depicted in Figure 5.6 where the feedforward block $\tilde{U}_f$ is seen to be a stable system.

The class of all stabilizing feedforward/feedback controllers parameterized in terms of □ is then given by

$$\square(\square) = \begin{bmatrix} K_f(\square) & K(\square) \end{bmatrix} = \tilde{V}(\square)^{-1} \begin{bmatrix} \tilde{U}_f(\square) & \tilde{U}(\square) \end{bmatrix}. \tag{5.28}$$

The arrangement is depicted in Figure 5.7(a) with an augmented □ given by

$$\square = \begin{bmatrix} K_f & K & \tilde{V}^{-1} \\ -V^{-1}N\tilde{U}_f & V^{-1} & -V^{-1}N \end{bmatrix}. \tag{5.29}$$

This arrangement can be reorganized to the scheme depicted of Figure 5.7(b) where J is that of (5.10) for the feedback case. The class of all stabilizing feedforward/feedback controllers then consists of the corresponding feedback controllers and another stable block Q_f in parallel with the nominal stable feedforward filter, also termed a *precompensator*, U_f.

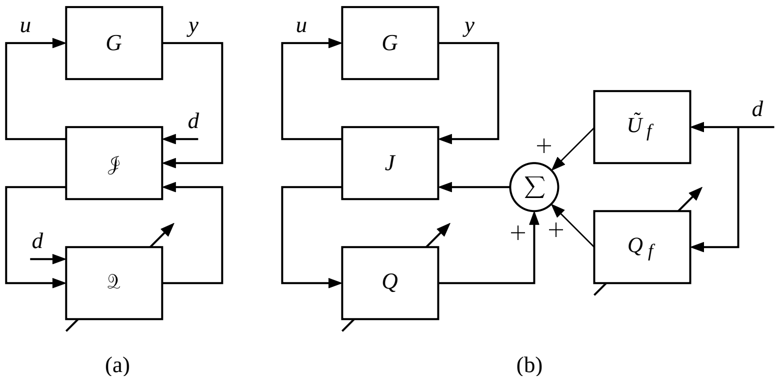


FIGURE 5.7. Class of all stabilizing feedforward/feedback controllers

2.6 All Stabilizing Regulators

A subset of the class of all stabilizing controllers is those that can regulate a disturbance response to a particular class of disturbances to zero asymptotically. Disturbance classes of interest could be, for example, the class of constant disturbances, the class of sinusoidal disturbances of known period, or the class of periodic disturbances of known period with harmonics up to a known order. Of course, the class of stabilizing regulators for certain disturbance classes and responses may be an empty set. Let us proceed assuming that this class is nonempty.

To conveniently characterize the class of all stabilizing regulators for a class of disturbances, it makes sense to exploit the knowledge we have gained so far as we illustrate now by way of example.

Consider the class of constant disturbances. In this case, for models as in Figure 2.1, the disturbance w_k is a constant of unknown value, and our desire is that the disturbance response e approach zero asymptotically for all w in this class. In order to proceed, let us modify the first step in our objective to requiring that the summed response $\bar{e}_k = \sum_{i=1}^{k} e_i$ be bounded (in ℓ_2) for bounded input disturbances w (in ℓ_2).

Now this BIBO stability for linear systems is equivalent to requiring that constant inputs w give rise to asymptotically constant responses $\bar{e}$, and in turn asymptotically zero responses e, as desired.

We conclude that the class of all regulators, regulating a disturbance response e_k to zero asymptotically for *constant input disturbances* is the class of all stabilizing controllers for the system augmented with a model of the deterministic disturbance itself, namely a summing subsystem, such that the augmented system disturbance response is $\bar{e}_k = \sum_{i=1}^{k} e_i$.

It is not difficult to check that the class of stabilizing regulators includes an *internal model* of the constant disturbances, namely a summing subsystem.

Sinusoidal disturbances and periodic disturbances can be handled in the same way, save that the augmentation and consequent internal model now consists of a discrete-time oscillator at the disturbance frequency, which is of course a model of the deterministic disturbance.

More generally, any deterministic disturbance derived from the initial condition response of linear system model, can be appended to the disturbance response e in the design approach described above to achieve regulation of such disturbances. The consequence is that in achieving a stabilizing controller design of the augmented system, there will be an internal model of the disturbances in the controller. For further details for characterizing the class of all stabilizing regulators, see Moore and Tomizuka (1989).

Of course, if regulation of only a component of the disturbance response e is required, then the augmentations of the deterministic disturbance model need be only to this component. This observation then leads us back again to the selection of the appropriate component of e so that regulation can be achieved. This component is frequently the plant output or a tracking error or filtered versions of these since, in general, it is not possible to have both the control u and the plant output y (or tracking error) regulated to zero in the presence of deterministic disturbances.

Main Points of Section

In this section, we examine the use of fractional representations of transfer functions to parameterize the class of all stabilizing controllers for a plant in terms of any stabilizing controller for the plant and a stable filter Q. The (matrix) transfer function of the closed-loop system is affine in the stable filter Q. The class of stabilizing controllers has convenient realizations in the feedforward/feedback (two-degrees-of-freedom) situation as well as in just the feedback (one-degree-of-freedom) situation. In particular, an interesting special case is where the filter Q augments a stabilizing state estimate feedback controller.

2.7 Notes and References

In this chapter, we have set the stage for the rest of the book by defining the plant representations with which we are working. We have introduced the concept of a nominal stabilizing feedback controller, and feedforward/feedback controller, for such plant models. The entire class of stabilizing controllers for the plant models is then parameterized in terms of a stable filter Q using a stable, coprime fractional representation approach. It turns out that this approach gives rise to closed-loop transfer functions that are affine in the stable filter Q. This result underpins the controller design methods to be explained in the rest of the book. In effect, our focus is on techniques that search within this class of stabilizing controllers for a controller that meets the performance objectives defined for the controller.

More information on definitions and manipulations of the plant model can be found in Ogata (1987), Åstrom and Wittenmark (1984), Franklin and Powell (1980), Boyd and Barratt (1991) and Doyle et al. (1992). A detailed exposition on linear system theory can be found in Kailath (1980) and Chen (1984).

Algebraic aspects of linear control system stability can be found in Kučera (1979), Nett (1986), Francis (1987), Vidyasagar (1985), McFarlane and Glover (1989) and Doyle et al. (1992). For the factorization approach to controller synthesis, the readers are referred to Youla, Bongiorno and Jabr (1976b), Kučera (1979), Nett, Jacobson and Balas (1984), Tay and Moore (1991), Francis (1987), and Vidyasagar (1985). For a general optimization approach exploiting this framework we refer the reader to Boyd and Barratt (1991).

Some of the third author's early works (with colleagues and students) in the topic of this chapter are briefly mentioned. Coprime factorizations based on *Luenberger observers* which could be of interest when seeking low order controllers are given in Telford and Moore (1989). Versions involving frequency shaped (dynamic) feedback "gains" F and H are studied in Moore, Glover and Telford

(1990). The difficulty of coping with decentralized controller structure constraints is studied in Moore and Xia (1989). Issues of *simultaneous stabilization* of multiple plants are studied in Obinata and Moore (1988).

Generalizations of the work of this chapter to generate the class of all stabilizing *two-degree-of-freedom controllers* are readily derived from the one-degree-of freedom controller class theory, see Vidyasagar (1985), Tay and Moore (1990) and Hara and Sugie (1988). Two-degree-of-freedom controllers incorporate both a feedforward control from an external input as well as feedback control as in this chapter. The most general form has a control block with output driving the plant and two sets of inputs, with dynamics coupling these. The controllers are parameterized in terms of a feedforward $Q_1 \in RH_\infty$ and the feedback $Q_2 \in RH_\infty$. Indeed $\square = [\, Q_1 \;\; Q_2 \,] \in RH_\infty$ can be viewed as the parameterization for the class of all stabilizing one-degree-of-freedom controllers for an augmented plant $[\, 0 \;\; G' \,]'$. An application and more details are given in Chapter 8.

Generalizations of the work of this chapter to the class of all *model matching controllers* is studied in Moore, Xia and Glover (1986). For this it is required that the closed-loop system match a model. Further details are not given here save that again there is a parameterization $Q \in RH_\infty$.

Other generalizations to the class of all *stabilizing regulators* can be achieved, see Moore and Tomizuka (1989). These regulators suppress asymptotically external deterministic disturbances such as step function inputs and sinusoidal disturbances. These can arise as discussed in the next chapter. Again the parameterization is a $Q \in RH_\infty$. Here however, the regulators/controllers must have an *internal model* of the disturbance source.

Then too there are generalizations of all these various stabilizing controller classes for *time-varying systems*, being still linear, see for example Moore and Tay (1989a). In fact, the theory of this chapter goes through line by line replacing transfer functions by linear system operators and working with BIBO stability, or equivalently now exponential asymptotic stability. Indeed, our $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ notation really refers to a linear system with matrices A, B, C, D which as it turns out can be time varying. Such results are used in Chapter 8, where also nonlinear generalizations are discussed.

Problems

1. **Derivation of plant model.**
   Consider the plant model of Figure 2.1 with $P_{22} = G$ and $P_{21} = I$. Construct the blocks P_{12} and P_{11} in terms of G so that $e = \begin{bmatrix} y \\ \lambda u \end{bmatrix}$ where λ is a constant.

   Notice that with e interpreted as a disturbance response and G restricted to a linear plant model, a control objective that minimizes the 2-norm of e will lead to the linear quadratic design problem.

2. **Transfer function to state-space transformation.**
   Consider a plant G with polynomial description given as follows.

$$A(z^{-1})y_k = B(z^{-1})u_k,$$
$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2}, \qquad B(z^{-1}) = b_1 z^{-1} + b_2 z^{-2}.$$

   By considering a related system given by $A(z^{-1})\zeta_k = u_{k-1}$, show that $y_k = B(z^{-1})u_k$. Show further that the plant G has a state-space description given by

$$x_{k+1} = \begin{bmatrix} -a_1 & -a_2 \\ 1 & 0 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_k, \qquad y_k = \begin{bmatrix} b_1 & b_2 \end{bmatrix} x_k.$$

   This particular state-space realization for the transfer function G is sometimes referred to as the controller canonical realization, although there is no universal agreement to the name. Note that this is just one of the many possible state-space realizations for the the transfer function G. The interested readers are referred to texts on linear systems such as Kailath (1980) and Chen (1984) for a detailed exposition.

3. **Elimination of uncontrollable and unobservable modes.**
   Consider the systems W_1 and W_2 of (2.9) with $A_1 = A_2$, $B_1 = B_2$ and their sum in (2.10). Show that after elimination of uncontrollable and unobservable modes, the sum of the systems can be represented as

$$(W_1 + W_2) : \left[\begin{array}{c|c} A_1 & B_1 \\ \hline C_1 + C_2 & D_1 + D_2 \end{array}\right]. \tag{7.1}$$

   With W_1, as in (2.9) with D invertible, verify that $W_1 W_1^{-1} = W_1^{-1} W_1$ has a minimal representation $\left[\begin{smallmatrix} 0 & 0 \\ 0 & I \end{smallmatrix}\right]$. Discuss internal stability.

4. **Internal stability.**
   In the series connection of two scalar systems W_1, W_2, with minimal representation given by (2.9), show that the series connection $W_1 W_2$ is minimal if and only if the transfer functions of W_1 and W_2 do not have any zeros common with poles of the transfer function of W_2 and W_1, respectively.

   Show that the single-input, single-output plant-controller pair (G, K) given by

$$G = \frac{0.4z^{-1} - 0.44z^{-2}}{1 - 2z^{-1} + 0.96z^{-2}}, \qquad K = \frac{-5 + 2.4z^{-1}}{1 - 1.1z^{-1}},$$

   is not internally stable in a feedback loop. Explain your findings in term of classical pole/zero cancellations. Does the following multi-input, multioutput plant and controller form a stabilizing pair in a feedback loop?

$$G = \begin{bmatrix} \frac{0.1z^{-1}}{1-0.6z^{-1}} & \frac{0.2z^{-1}}{1-0.6z^{-1}} \\ \frac{0.2z^{-1}}{1-0.6z^{-1}} & \frac{0.1z^{-1}}{1-1.1z^{-1}} \end{bmatrix}, \qquad K = \begin{bmatrix} \frac{-0.3z^{-1}}{1-0.1z^{-1}} & 1.65 \\ \frac{-4(1-0.08z^{-1})}{1-0.1z^{-1}} & \frac{-6(1-1.1z^{-1})}{1-0.1z^{-1}} \end{bmatrix}.$$

Are there any pole/zero cancellations between the plant and controller? How are transmission zeros of a multi-input, multioutput transfer function defined?

5. **Implementation of feedforward/feedback controllers.**
A two degrees-of-freedom controller given as

$$\Box = \begin{bmatrix} K_f & K \end{bmatrix} = \begin{bmatrix} \frac{0.5z^{-1}}{(1-1.2z^{-1})(1-0.5z^{-1})} & \frac{0.2z^{-1}(1-0.7z^{-1})}{(1-1.2z^{-1})(1-0.6z^{-1})} \end{bmatrix}$$

is implemented as in Figure 7.1. Explain why the structure used in the figure is not suitable for implementing K. Suggest a more suitable structure.

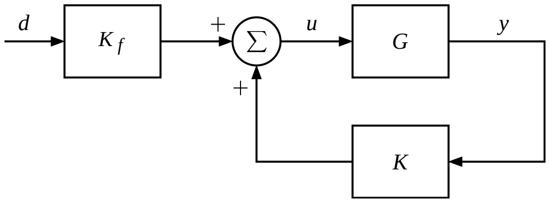


FIGURE 7.1. Signal model for Problem 5

6. **Coprime factorizations.**
Verify that the representations (4.18) and (4.19) indeed are factorizations for G and K. Establish that in the case that the plant of (4.16) has no direct feedthrough, that is $D = 0$, then we can select a gain matrix L where AL is a stabilizing output injection such that the feedback controller K is now given by

$$K : \left[\begin{array}{c|c} A + BF + ALC + BFLC & -(AL + BFL) \\ \hline F + FLC & -FL \end{array}\right]. \tag{7.2}$$

Note that this controller has a direct feedthrough term $-(FL)$, although the overall control loop transfer function GK is still strictly proper.

Also, for the plant G with the controller (7.2), stable coprime factorizations are given by

$$\begin{bmatrix} M & U \\ N & V \end{bmatrix} : \left[\begin{array}{c|cc} A + BF & B & -(A + BF)L \\ \hline F & I & -FL \\ C & 0 & I \end{array}\right],$$

and

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} : \left[\begin{array}{c|cc} A+ALC & -B & AL \\ \hline F+FLC & I & FL \\ C & 0 & I \end{array}\right]. \tag{7.3}$$

7. **Coprime factorizations.**
   For the multi-input, multioutput plant and controller given in Problem 4, suggest stable, coprime factorizations in $|z| < 1$.

8. **Coprime factorizations.**
   Verify that the coprime factorizations given in (4.22) and (4.23) satisfy the double Bezout equation.

9. **Implementation of the class of all stabilizing controllers.**
   Starting from the expression for J given in (5.10), verify that the structure of Figure 5.3 implements the class of all stabilizing controllers $K(Q)$ for the plant G, given in (5.9). Verify the state space representation of J given in (5.14) and T given in (5.22). Hints: In deriving (5.14), recall that K is given from (4.17), and note that from (4.18), (4.19)

$$V = \left[\begin{array}{c|c} A+BF & -H \\ \hline C+DF & I \end{array}\right],$$

$$N = \left[\begin{array}{c|c} A+BF & B \\ \hline C+DF & D \end{array}\right],$$

$$\tilde{V} = \left[\begin{array}{c|c} A+HC & -(B+HD) \\ \hline F & I \end{array}\right].$$

Apply the manipulations of (2.11)–(2.21) to obtain (5.14) from (5.10). In verifying (5.22), check that indeed $T_{22} = 0$, and that T_{12} and T_{21} are given by (5.20), (5.21), and that $T_{11} = P_{11} + P_{12}U\tilde{M}P_{21} = P_{11} + P_{12}UT_{21}$ is given by

$$T_{11} = \left[\begin{array}{cc|c} A+B_2F & -HC & -HD_{21} \\ 0 & A+HC & B_1+HD_{21} \\ \hline C_1+D_{12}F & C_1 & D_{11} \end{array}\right].$$

Key intermediate steps in the derivation are spelled out below.

$$
\begin{aligned}
T_{11} &= \left[\begin{array}{cc|c} A & B_2F & 0 \\ 0 & A+B_2F & -H \\ \hline C_1 & D_{12}F & 0 \end{array}\right] \left[\begin{array}{cc|c} A+HC & HC_2 & HD_{21} \\ 0 & A & B_1 \\ \hline C_2 & C_2F & D_{21} \end{array}\right] \\
&\quad + \left[\begin{array}{c|c} A & B_1 \\ \hline C_1 & D_{11} \end{array}\right] \\
&= \left[\begin{array}{ccccc|c} A & 0 & 0 & 0 & 0 & B_1 \\ 0 & A & B_2F & 0 & 0 & 0 \\ 0 & 0 & A+B_2F & -HC_2 & -HC_2 & -HD_{21} \\ 0 & 0 & 0 & A+HC_2 & HC_2 & HD_{21} \\ 0 & 0 & 0 & 0 & A & B_1 \\ \hline C_1 & C_1 & D_{12}F & 0 & 0 & D_{11} \end{array}\right].
\end{aligned}
$$

Now add block row 5 to block row 4 and subtract block column 4 from block column 5, then delete block row 5, and block column 5 (representing unobservable modes). Next subtract block column 2 from block column 1 and add block row 1 to block row 2, then delete the block first row and column (again representing unobservable modes). Next add block column 1 to block column 2 and subtract block row 2 from block row 1, then add block column 1 to block column 3 and subtract block row 3 from block row 1, and finally delete block row 1 and column 1 (representing uncontrollable modes) to give the required form for T_{11}.

CHAPTER 3

Design Environment

3.1 Introduction

In this chapter, we focus on the environment in which our controller designs are carried out. We begin with a discussion of the various types of disturbances in the design environment. We introduce models for disturbances, both predictable and nonpredictable, and the concept of norms to measure the size of disturbances. We then go on to discuss plant uncertainties in the modeling process. General frequency domain and time domain model uncertainties are discussed.

Next, we introduce a special form of frequency-shaped uncertainty motivated by the characterization of the class of all stabilizing controllers. We examine the class of all plants stabilizable by a nominal controller. This is dual to the case of the class of all stabilizing controllers for a plant, and is characterized in terms of a (matrix) transfer function, denoted S. It turns out that there is an interesting relationship between an actual plant in the class of plants stabilized by a nominal controller, the nominal plant description and the parameter S. The parameter S is a frequency-shaped deviation of the actual plant from the nominal plant with the frequency-shaping emphasizing the operating frequencies in the nominal closed loop and the actual closed-loop systems.

3.2 Signals and Disturbances

In this section, we first suggest models for some commonly encountered predictable or deterministic disturbances. This is followed by a discussion on nondeterministic or stochastic disturbances. The concept of norms to measure the size of such noise is introduced. Much of this material may be familiar to readers, but is included for completeness and reference.

Deterministic Disturbance Model

A deterministic disturbance is one where its future can be perfectly predicted based on past observations. A number of useful disturbance models can be constructed and the most appropriate one depends on the nature of the disturbance environment and control design objectives. Here we discuss a few of the more commonly encountered disturbances in applications and provide a general formulation for these. The disturbances of interest can be modeled by homogeneous (time-invariant) state space or input/output models. In turn these models can combine with the plant models to give composite models of the plants in conjunction with the disturbance environment. The inclusion of disturbance models is essential in most control applications. The ubiquity of integral action in control arises from the need to have accurate set point regulation, or equivalently, the rejection of a constant disturbance.

Deterministic disturbances characterized by a finite, discrete frequency spectrum can easily be modeled using autonomous state space models. Typical examples follow.

DC offset

A constant disturbance can be modeled as:

$$(1 - q^{-1})d_k = 0, \qquad d_0 = \text{constant}, \qquad k = 1, 2, \ldots \tag{2.1}$$

where q^{-1} is the unit delay operator (i.e. $q^{-1}d_k = d_{k-1}$).

Drift at Constant Rate

A ramp like disturbance has a discrete-time model given by

$$(1 - 2q^{-1} + q^{-2})d_k = 0, \qquad d_0 = \beta, \qquad d_{-1} = \beta - \alpha T, \qquad k = 1, 2, \ldots \tag{2.2}$$

where T is the sampling interval. Here $d_k = \alpha T k + \beta$.

Sinusoidal Disturbance

A single frequency sinusoidal disturbance can be modeled as the solution of the difference operation.

$$(1 - (2\cos\omega T)q^{-1} + q^{-2})d_k = 0, \qquad d_0 = \alpha, \qquad d_{-1} = \alpha\cos\omega T. \tag{2.3}$$

The solution of (2.3), namely $d_k = \alpha\cos(\omega T k)$, corresponds to sampling a continuous-time disturbance $d(t) = \alpha\cos(\omega t)$ with sampling period T.

More General Disturbances

Any finite combination of sinusoidal, exponential or polynomial functions of time can be represented as the solution of an appropriate difference equation. Such disturbances may be modeled as:

$$\Gamma(q^{-1})d_k = 0, \tag{2.4}$$

suitably initialized where

$$\Gamma(z^{-1}) = 1 + \gamma_1 z^{-1} + \cdots + \gamma_n z^{-n}. \tag{2.5}$$

The zeros of the polynomial Γ determine the discrete frequency spectrum contained in the disturbance. Alternatively, a state space model can be used:

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} -\gamma_1 & 1 & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \vdots & 0 & \dots & 0 & 1 \\ -\gamma_n & 0 & \dots & \dots & 0 \end{bmatrix} x_k; \qquad x_0, \\ d_k &= \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} x_k. \end{aligned} \tag{2.6}$$

The model for a periodic disturbance with known period but unknown harmonic content is in general infinite dimensional. In practice, such a model is approximated by a finite-dimensional model as above.

Stochastic Disturbance Models

In a control environment, deterministic disturbances can be dealt with adequately by appropriate control action. Many disturbances however are not predictable. A controller solely determined on the basis of models without allowing for any uncertainty could fail to achieve acceptable performance in an environment that differs from its design environment. In order to determine a practical controller, it is helpful to include unpredictable disturbances in the signal environment. Stochastic disturbances can be modeled using stochastic processes. In the context of linear system design, simple stochastic processes whose statistical characterization can be expressed in terms of first and second order means suffice. Most of the design methods in linear system theory, linear quadratic methods, ℓ_1 optimization and H_∞ can all be motivated and obtained in a purely deterministic setting as well as in a stochastic setting. This is because first and second order means can be defined via time averages rather than expected values over some underlying probability space defining the stochastic variable. This has been explored in depth in Ljung and Söderström (1983). In our setting, it suffices to consider disturbances like w_k, $k = 1, 2, \dots$ that can be characterized in the following way:

Definition. Bounded, zero mean, white noise process w_k. For some constants b, $b_1, b_2 \geq 0$ and $0 \leq \gamma < 1$ and all sufficiently large N:

1. $$|w_k| \le b,$$

   (Bounded).

2. $$\left|\sum_{k=1}^{N} w_k\right| \le b_1 N^{\gamma}, \tag{2.7}$$

   (Zero mean).

3. $$\left|\sum_{k=m+1}^{N} w_k w'_{k-m} - W\delta_m\right| \le b_2 N^{\gamma}, \qquad m = 1, 2, \dots$$

   (Uncorrelated and with variance W).

Here $\delta_k = 1$ if $k = 0$ and $\delta_k = 0$ for $k \neq 0$.

The frequency spectrum of this noise is flat, hence the term white noise.

Definition. The bounded sequences $w_k, v_k, \ k = 1, 2, \dots$ are said to be uncorrelated if for all sufficiently large N,

$$\left|\sum_{k=m+1}^{N} w_k v'_{k-m}\right| \le b_3 N^{\gamma} + b_4, \qquad m = 1, 2, \dots$$

for some positive constants $b_3, b_4 \ge 0$ and $0 \le \gamma < 1$.

If it is important to limit the frequency content of the noise, a filtered or colored noise sequence can be derived from w_k via an appropriate linear filter that accentuates the frequency spectrum of interest as follows:

$$\begin{aligned} x_{k+1} &= A_f x_k + B_f w_k, \\ y_k &= C_f x_k + D_f w_k. \end{aligned} \tag{2.8}$$

If w_k in (2.8) satisfies the white noise condition (2.7) and A_f is a stable matrix, then y_k is bounded, has zero mean, but is correlated over time. Also, y_k has a well defined *autocorrelation*. Its *frequency spectrum* is given by:

$$\Phi_{yy}(\theta) = \Big(C_f\big(Ie^{-j\theta} - A_f\big)^{-1}B_f + D_f\Big)'\Big(C_f\big(Ie^{j\theta} - A_f\big)^{-1}B_f + D_f\Big),$$

where $\theta \in [0, 2\pi)$.

Norms as Performance Measures

The performance of a controller can only be expressed via characteristics of the signals arising from the controlled loop. An important characteristic of a signal is its size. Norm functions are one way to determine the size of a signal.

Consider sequences $d := \{d_k, k = 1, 2, \dots\}$ mapping N to R^n. This constitutes a linear space of signals $\ell(\mathsf{N}, \mathsf{R}^n)$, where addition of signals and multiplication with a scalar are defined in the obvious way: $d + e = \{d_k + e_k, k = 1, 2, \dots\}$; $\alpha d = \{\alpha d_k, k = 1, 2, \dots\}$ for $d, e \in \ell(\mathsf{N}, \mathsf{R}^n)$ and $\alpha \in \mathsf{R}$. A *norm* $\|\cdot\|$ for signals has to satisfy the following properties. It maps from the signal space to the positive reals $\|d\| \geq 0$. It measures zero only for the zero signal, that is $\|d\| = 0$ if and only if $d = 0$ or $d_k = 0$ for all $k = 1, 2, \dots$. It scales appropriately $\|\alpha d\| = |\alpha| \, \|d\|$, $\alpha \in \mathsf{R}$ and satisfies the triangle inequality $\|d + e\| \leq \|d\| + \|e\|$. Typical examples are the ℓ_p norms. The ℓ_p norm is defined as follows:

$$\|d\|_p = \lim_{N \to \infty} \left(\sum_{k=1}^{N} \sum_{i=1}^{n} d_{i,k}^p \right)^{1/p}, \qquad d_k = \left(d_{1,k} \dots d_{n,k} \right)'; \qquad p > 0.$$

Commonly used ℓ_p norms are the ℓ_∞ norm, which measures the maximum value attained by the sequence; the ℓ_1 norm which measures the sum of absolute values and the ℓ_2 norm which is often referred to as an energy measure:

$$\begin{aligned}
\|d\|_\infty &= \sup_{k \in \mathsf{N}} \max_{i=1,\dots,n} \left| d_{i,k} \right|, \\
\|d\|_1 &= \lim_{N \to \infty} \sum_{k=1}^{N} \sum_{i=1}^{n} \left| d_{i,k} \right|, \\
\|d\|_2 &= \lim_{N \to \infty} \left(\sum_{k=1}^{N} \sum_{i=1}^{n} \left(d_{i,k} \right)^2 \right)^{1/2}.
\end{aligned} \tag{2.9}$$

In discrete time, signals with finite ℓ_1 or ℓ_2 norm converge to zero as time progresses; such signals are of a transient nature.

A less common norm is:

$$\|d\| = \sup_{k \in \mathsf{N}} \max_{i=1,2,\dots,n} \left(\left| d_{i,k+1} - d_{i,k} \right| \right) + \sup_{k \in \mathsf{N}} \max_{i=1,2,\dots,n} \left| d_{i,k} \right|. \tag{2.10}$$

It measures not only the magnitude of the signal but also the rate of change of the signal. A signal bounded in this norm is not only of finite magnitude but also rate limited.

Sometimes norms are not informative enough to measure the quality of a signal, for example any white noise signal with Gaussian distribution has infinite ℓ_∞ and ℓ_2 norms. A more appropriate measure for such signals would be:

$$\|d\|_{\text{rms}} = \lim_{N \to \infty} \left(\frac{1}{N} \sum_{k=1}^{N} d_k' d_k \right)^{1/2}. \tag{2.11}$$

This is not a norm however. Indeed any signal of finite duration, that is $d_k = 0$ for all $k \geq k_0 > 0$, or more generally that converges to zero has zero *root mean square* (*rms*) measure. The rms value measures the average power content of a signal. It is an excellent indicator for differentiating between persistent signals and nonpersistent ones.

For vector valued signals the different components in the vector may have different significance. This can be made explicit in measuring the size of the signal by weighting the different components in the signal differently, for example

$$\|d\| = \|Pd\|_\infty \,, \tag{2.12}$$

is a norm for any symmetric positive definite matrix P of appropriate dimension. In particular, $P = \operatorname{diag}(p_1, p_2, \ldots, p_n)$ with $p_i > 0,\ i = 1 \ldots n$ allows for individual scaling of each component of the signal d. Proper scaling of the signals is extremely important in practical application.

In concluding this section, we remark that similar norms can be defined for continuous-time signals in very much the same fashion. We refer the reader to Boyd and Barratt (1991) or Sontag (1990).

Main Points of Section

In the context of discrete-time signals, we see that the concept of a norm is a useful measure of the size of a signal. Commonly used norms are introduced.

3.3 Plant Uncertainties

Models are only approximate representations for physical plants. Approximations are due to a number of simplifications made in the modeling process, for example we use linear, time-invariant models to represent systems which are really time varying and nonlinear. As an illustration, consider a simple linear, time-invariant model for the yaw control of an airplane. Complete description of the airplane's dynamics are nonlinear in that they depend on speed, altitude, and mass distribution; factors that vary over the flight envelope of the airplane. A simple linear time-invariant model can at best capture only the essential behavior in the neighborhood of the one flight condition.

Model-plant mismatch can be represented or characterized in a number of different ways. The mismatch measure must capture the different behaviors of the model and the plant for a range of signals that can be applied to the model and the plant. This model-plant mismatch, also termed uncertainty, is therefore best measured via signals. Both time domain and frequency domain characterizations are important and play a role in control design.

Model-plant mismatch is also an important factor in determining controller properties. A controller designed for the model must at least retain stable behavior when applied to the plant, that is *robustness* to plant model uncertainty. In expressing robustness it becomes important to quantify the neighborhood of plants near the model that can be stabilized/controlled by the particular controller of interest. In this context, norms for models/transfer functions become important.

In this section we discuss concisely different model-plant mismatch characterizations and introduce some frequently used norms for models/transfer functions.

Norms for Models

When discussing differences between a plant and a model, it is important to be able to quantify this difference. A norm function is a suitable way of achieving this, at least for stable plants. Consider the class of stable rational transfer function matrices, with a realization:

$$G : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \in RH_\infty. \tag{3.1}$$

Induced ℓ_p Norms

One way of defining a norm for a plant model G is via the signals it links. Suppose that Gu is the output produced by applying the signal u to the model G, setting all initial conditions to zero as follows:

$$x_{k+1} = Ax_k + Bu_k; \qquad x_0 = 0, \tag{3.2}$$

$$(Gu)_k = Cx_k + Du_k. \tag{3.3}$$

The ℓ_p gain of G is defined by

$$\|G\|_{p-gn} := \sup_{\|u\|_p=1} \|Gu\|_p \,. \tag{3.4}$$

It measures the gain between the input to the plant G and the output produced by the plant acting on its input.

The ℓ_∞ gain of the system G is equal to the ℓ_1 norm of the response of G to an impulse input $u_k = \delta_k$ where $\delta_k = 1$ for $k = 1$ and $\delta_k \neq 0$ otherwise. This so-called *impulse response* is denoted g or $g_1, g_2, \ldots$ with $g_0 = 0,\ g_1 = CB + D$, $g_2 = CAB,\ g_i = CA^{i-1}B$ for $i = 2, 3, \ldots$. Indeed, we have:

$$\|G\|_{\infty-gn} = \sum_{i=1}^{\infty} \|g_i\|_\infty := \|g\|_1 \,.$$

Here $\|g_i\|_\infty$ is the induced ∞-norm for the matrix g_i. (See Appendix A).

The ℓ_2 gain of the system G is also referred to as the H_∞ norm or as the rms gain.

$$\|G\|_{2-gn} = \|G\|_{\text{rms}} = \sup_{\|u\|_{\text{rms}}=1} \|Gu\|_{\text{rms}} \,.$$

Observe that unlike for signals, $\|G\|_{\text{rms}}$ is a true norm for the system G. It can be computed as:

$$\|G\|_{2-gn} = \sup_{0\le\theta\le2\pi} \sigma_{\max}\left(G\left(e^{j\theta}\right)\right),$$

where $\sigma_{\max}(A)$ stands for the maximum singular value of the matrix A. In reluctant compliance with the literature, we at times denote $\|G\|_{2-gn} = \|G\|_\infty$, and refer to it as the H_∞ *norm* of the system G, see also Appendix B.

The frequency domain interpretation of the ℓ_2 gain (or H_∞ norm) for a system is very natural in many applications. Unfortunately it puts the whole frequency spectrum on an equal footing. Often the gains in specific frequency ranges are more important than outside these ranges. A frequency weighted ℓ_2 gain can then be used:

$$\begin{aligned}\|G\|_W &= \|W_o G W_i\|_{2-gn}\\ &= \sup_{0\le\theta\le 2\pi} \sigma_{\max}\left(W_o G W_i\left(e^{j\theta}\right)\right).\end{aligned}$$

Here W_o and W_i are frequency weighting systems;

$$W_o : \left[\begin{array}{c|c} A_o & B_o \\ \hline B_o' & D_o \end{array}\right], \qquad W_i : \left[\begin{array}{c|c} A_i & B_i \\ \hline B_i' & D_i \end{array}\right],$$

respectively, at the output and input of the plant G. Without loss of generality, the realizations are symmetric: $A_o = A_o'$, $D_o = D_o'$, $A_i = A_i'$ and $D_i = D_i'$. Also $W_o\left(e^{j\theta}\right) \ge 0$, $W_i\left(e^{j\theta}\right) \ge 0$ for all $0 \le \theta \le 2\pi$. The main motivation for having W_0, W_1 with symmetric realizations is to interpret the frequency weighted norm as a proper norm, see Boyd and Barratt (1991).

Computing the ℓ_2 Gain

The ℓ_2 gain can be approximated using state space methods. For the system G, we have that for stable G with a realization as given in (3.1) there exists an upper bound γ such that

$$\|G\|_{2-gn} = \sup_{\substack{0\le\theta<2\pi\\ r>1}} \sigma_{\max}\left(G\left(re^{j\theta}\right)\right) \le \gamma,$$

if and only if there exists a symmetric positive definite P and matrices L, W satisfying

$$\begin{bmatrix} A' & C' & L' \\ B' & D' & W' \end{bmatrix} \begin{bmatrix} P & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ C & D \\ L & W \end{bmatrix} = \begin{bmatrix} P & 0 \\ 0 & \gamma^2 I \end{bmatrix}.$$

This can be used as the basis for a numerical procedure to compute the ℓ_2 gain (H_∞ norm) of a stable rational transfer function matrix. Starting with a suitably high value of γ for which a positive semi-definite solution P of the above linear equation exists, then γ is progressively reduced until semi-definiteness of the solution P fails. This procedure yields a least upper bound γ, or equivalently, the ℓ_2 gain, see Green and Limebeer (1994).

H_2 norm of a system

The ℓ_2 norm of the response to an impulse input is also often used as a measure of a system; this is the so-called H_2 norm.

$$\|G\|_2 = \|G\delta\|_2 = \frac{1}{2\pi}\int_0^{2\pi} G\left(e^{-j\theta}\right)' G\left(e^{j\theta}\right) d\theta. \tag{3.5}$$

Here as before, $\delta_k = 1$ for $k = 0$ and $\delta_k = 0$ for $k \neq 0$.

Parseval's Theorem which equates energy in the time and frequency domains, is used here to establish the second equality in (3.5). It allows us to interpret the H_2 norm as the energy content of the output of the system subject to white noise—recall that white noise has a unit frequency spectrum. This simple equivalence allows one to interpret many design optimization schemes in an either purely time-domain deterministic, or frequency-domain stochastic context.

As a final interpretation of the H_2 norm, it is possible to show that

$$\|G\|_2 = \sup_{\|w\|_2=1} \|Gw\|_\infty .$$

This establishes a link between the ℓ_∞ norm of the time response to an ℓ_2 signal and a frequency defined 2-norm.

Frequency Domain Uncertainty

Expressing uncertainty in the frequency domain is appealing as it provides physical insight. The ℓ_2 gain, especially the weighted ℓ_2 gain, is the preferred tool for this.

We illustrate the ideas using stable single-input, single-output (SISO) stable plants. Consider:

$$G : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \in RH_\infty.$$

The ℓ_2 gain (H_∞ norm) of G is then simply the maximal amplitude in the Bode plot of G's transfer function:

$$\|G\|_{2-gn} = \max_{0\le\theta\le 2\pi} \left| C\left(e^{j\theta}I - A\right)^{-1} B + D \right| .$$

When G is a model for a real plant $\bar{G}$, the uncertainty associated with G can be expressed in terms of a weighting system W as:

$$\Delta_W = \left\{ \bar{G} : \left\| \left(G - \bar{G}\right) W \right\|_{2-gn} \le 1 \right\} .$$

In frequency terms the above inequality corresponds to:

$$\left| G\left(e^{j\theta}\right) - \bar{G}\left(e^{j\theta}\right) \right| \cdot \left| W\left(e^{j\theta}\right) \right| < 1.$$

It simply states that the plant's transfer function response at any one frequency can be found in a ball centered at the model's response and with radius $\left|W\left(e^{j\theta}\right)\right|^{-1}$. A large weight indicates small uncertainty, while a small weight indicates large uncertainty. This uncertainty description allows for both amplitude and phase errors in the transfer functions.

Time Domain Uncertainty

Besides having a nonexact model for the system's transfer function from control inputs to controlled variables, the response may be perturbed by other signals from other systems affecting the output variables. Such uncertainties are best captured by a combination of signal norms and system norms. Our standard two port model for the plant allows for such uncertainties, see Figures 2.2.1 and 2.2.2*. With reference to Figure 2.2.2, w_1, w_2 can be interpreted as disturbances, G as an approximation for the true system linking control input u and controlled output y. Interpreting the uncertainty associated with G in ℓ_2 gain terms, it then makes sense to interpret the use of w_1 and w_2 in terms of rms measures. The uncertainty environment associated with the control loop could then be expressed as:

$$\left\|\left(G-\bar{G}\right)W\right\|_{\text{rms}} \leq 1, \qquad \|w_2\|_{\text{rms}} \leq \bar{W}_2, \qquad \|w_1\|_{\text{rms}} \leq \bar{W}_1.$$

The effect of additive disturbances in a linear system is to deteriorate the control performance. Such signals can not effect stability. In the context of H_∞ optimization, discussed in Chapter 4, we introduce the concept of a worst case disturbance, which results in a worst case performance.

Main Points of Section

Norms for linear systems such as the induced ℓ_p norm and H_2 norm, and other measures such as rms measures, are useful for representing uncertainty in plant models. Frequency domain or time domain representations of this uncertainty are useful to achieve robust controller design.

3.4 Plants Stabilized by a Controller

In this section we apply the characterization of the class of all stabilizing controllers for a nominal plant to the dual situation of characterizing the class of all plants stabilizable by a given controller. In particular, we begin with the internally stable closed-loop system of Figure 2.3.2 formed by the nominal plant G and a stabilizing controller K. We then classify the entire class of plants $G(S)$ parameterized by a (matrix) transfer function, referred to as $S \in RH_\infty$, that can be stabilized by the controller K.

*In referencing figures from another chapter, the first of the three numbers indicates the chapter.

Class of All Plants Stabilizable by a Controller

Let us consider the plant $G \in R_p$ of (2.2.4)[†] and its corresponding stabilizing controller K. Let the stable, coprime factorizations for both G and K that satisfy the double Bezout identity of (2.4.12) be given by (2.4.1) and (2.4.2) of the previous chapter, and repeated here for convenience as

$$\begin{aligned} G &= NM^{-1} = \tilde{M}^{-1}\tilde{N}; \qquad N, M, \tilde{N}, \tilde{M} \in RH_\infty, \\ K &= UV^{-1} = \tilde{V}^{-1}\tilde{U}; \qquad U, V, \tilde{U}, \tilde{V} \in RH_\infty, \end{aligned} \tag{4.1}$$

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix}\begin{bmatrix} M & U \\ N & V \end{bmatrix} = \begin{bmatrix} M & U \\ N & V \end{bmatrix}\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{4.2}$$

The class of all proper plants stabilized by the controller K can then be characterized as in the following theorem.

Theorem 4.1. *For the factorizations of* (4.1) *the class of all proper linear plants stabilized by the controller K can be parameterized by an arbitrary $S \in RH_\infty$ as*

$$\{G(S) \mid S \in RH_\infty\} \tag{4.3}$$

where

$$\begin{aligned} G(S) &= N(S)M(S)^{-1}; \quad N(S) = N + VS, \quad M(S) = M + US, \quad \textit{or} \\ G(S) &= \tilde{M}(S)^{-1}\tilde{N}(S); \quad \tilde{N}(S) = \tilde{N} + S\tilde{V}, \quad \tilde{M}(S) = \tilde{M} + S\tilde{U}. \end{aligned}$$

Proof. The proof of the theorem follows closely the development of the class of all stabilizing controllers for a proper plant in the previous chapter. Interchanging the role of G and K, (2.5.7) of the previous chapter is written as

$$\begin{bmatrix} I & -G(S) \\ -K & I \end{bmatrix}^{-1} = \begin{bmatrix} I & -G \\ -K & I \end{bmatrix}^{-1} + \begin{bmatrix} V \\ U \end{bmatrix} S \begin{bmatrix} \tilde{U} & \tilde{V} \end{bmatrix}. \tag{4.4}$$

From (4.4), it is clear that any plant parameterized by $S \in RH_\infty$ is stabilized by the controller K. Conversely, from (4.4) and the double Bezout identity (4.2), the (matrix) transfer function S is given as

$$S = \begin{bmatrix} \tilde{M} & -\tilde{N} \end{bmatrix}\left\{\begin{bmatrix} I & -G(S) \\ -K & I \end{bmatrix}^{-1} - \begin{bmatrix} I & -G \\ -K & I \end{bmatrix}^{-1}\right\}\begin{bmatrix} -N \\ M \end{bmatrix}. \tag{4.5}$$

If both G and $G(S)$ are stabilized by the controller K, then the closed-loop (matrix) transfer functions of $(G(S), K)$ are stable. In this case, S as given in (4.5) satisfies $S \in RH_\infty$ □

[†]In referencing equations from another chapter, the first of the three numbers indicates the chapter.

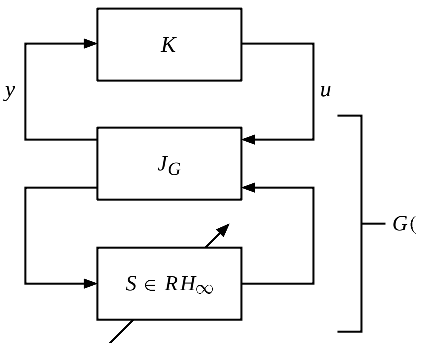


FIGURE 4.1. Class of all proper plants stabilized by K

As in (2.5.9), $G(S)$ can be re-expressed via the double Bezout identity of (4.2) as

$$G(S) = G + \tilde{M}^{-1}S(I + M^{-1}US)^{-1}M^{-1}. \tag{4.6}$$

which can then be reorganized as in Figure 4.1 with

$$J_G = \begin{bmatrix} G & \tilde{M}^{-1} \\ M^{-1} & -M^{-1}U \end{bmatrix}. \tag{4.7}$$

From (4.6), note that $G(S)$ consists of the sum of G, the nominal plant and a term involving the parameter $S \in RH_\infty$. In the next subsection we actually interpret the parameter S as a frequency-shaped mismatch between the actual plant and the nominal plant model.

Interpretation of S

From (4.6), S can be written as

$$\begin{aligned} S &= \tilde{M}(G(S) - G)M(I + M^{-1}US) \\ &= \tilde{M}(G(S) - G)(M + US) \\ &= \tilde{M}(G(S) - G)M(S), \end{aligned} \tag{4.8}$$

or alternatively, as

$$S = \tilde{M}(S)(G(S) - G)M. \tag{4.9}$$

We see that any $S \in RH_\infty$ will generate a unique $G(S)$ that forms a stabilizing pair $(G(S), K)$, and conversely, for each stabilizing pair (G_1, K) there exists a unique $S \in RH_\infty$ which generates $G_1 = G(S)$. It is immediate from (4.8) and (4.9) that S can be interpreted as the difference in the (matrix) transfer functions between the actual plant $G(S)$ and the plant G, frequency-shaped by either

$\tilde{M}(S)$, M or $\tilde{M}$, $M(S)$. It is important to realize that since the factorizations are not unique, the frequency-shaping is not unique. Let us explore the nature of the frequency shaping.

From the Bezout identity, we have

$$\begin{aligned} \tilde{V}M(S) - \tilde{U}N(S) &= I, \\ \tilde{V}(I - KG(S))M(S) &= I, \\ M(S) &= (I - KG(S))^{-1}\tilde{V}^{-1}. \end{aligned} \tag{4.10}$$

Similarly

$$\tilde{M}(S) = V^{-1}(I - G(S)K)^{-1}. \tag{4.11}$$

It is immediate that $M(S)$ or $\tilde{M}(S)$ provides frequency shaping of $(G(S) - G)$ to emphasize the actual operating frequencies in the closed loop. Of course, taking $S = 0$ in (4.10), (4.11), we see that M or $\tilde{M}$ provides frequency shaping for $(G(S) - G)$ to emphasize the operating frequencies in the nominal closed loop. Note that $\tilde{M}$ and $M(S)$, M and $\tilde{M}(S)$ are determined by the plant $G(S)$, the model G and the nominal controller K.

Example. Let us now demonstrate some of the results above for the case where the underlying actual process has a scalar *auto-regressive, moving average, exogenous input model* (ARMAX model) description, in operator (transform) notation

$$y = \bar{G}u + \bar{G}_w w, \tag{4.12}$$

where

$$\begin{aligned} \bar{G} &= \frac{\bar{\square}(z^{-1})}{\bar{\square}(z^{-1})}, \qquad \bar{G}_w = \frac{\bar{\square}(z^{-1})}{\bar{\square}(z^{-1})}, \\ \bar{\square}(z^{-1}) &= 1 + \bar{a}_1 z^{-1} + \cdots + \bar{a}_{\bar{m}} z^{-\bar{m}}, \\ \bar{\square}(z^{-1}) &= \bar{b}_0 + \bar{b}_1 z^{-1} + \cdots + \bar{b}_{\bar{n}} z^{-\bar{n}}, \\ \bar{\square}(z^{-1}) &= 1 + \bar{c}_1 z^{-1} + \cdots + \bar{c}_{\bar{p}} z^{-\bar{p}}. \end{aligned} \tag{4.13}$$

Assume that the nominal plant G is available and is given by

$$\begin{aligned} G &= \frac{\square(z^{-1})}{\square(z^{-1})}, \\ \square(z^{-1}) &= 1 + a_1 z^{-1} + \cdots + a_m z^{-m}, \\ \square(z^{-1}) &= b_0 + b_1 z^{-1} + \cdots + b_n z^{-n}. \end{aligned} \tag{4.14}$$

Let the factorizations for $\bar{G}$ and G be given by

$$\begin{aligned} M = \tilde{M} &= \frac{\square(z^{-1})}{\square(z^{-1})}, \\ N = \tilde{N} &= \frac{\square(z^{-1})}{\square(z^{-1})}, \\ M(S) = \tilde{M}(S) &= \frac{\bar{\square}(z^{-1})}{\bar{\square}(z^{-1})}, \\ N(S) = \tilde{N}(S) &= \frac{\bar{\square}(z^{-1})}{\bar{\square}(z^{-1})}, \end{aligned} \tag{4.15}$$

where $\bar{D}(z^{-1})$ and $D(z^{-1})$ are stable polynomials derived from the factorizations of (2.4.18), (2.4.19), (2.4.22) and (2.4.23). These reflect the nominal closed-loop poles and can be appropriately designed through the design of some nominal controller K.

Now let us assume that S is parameterized by polynomials $\square_S(z^{-1})$, $\square_S(z^{-1})$ as

$$S = frac\square_S(z^{-1})\square_S(z^{-1}). \tag{4.16}$$

Then from (4.8), we have

$$\begin{aligned} \square_S(z^{-1}) &= \bar{\square}(z^{-1})\square(z{-}1), \\ \square_S(z^{-1}) &= \bar{\square}(z^{-1})\square(z{-}1) - \bar{\square}(z^{-1})\square(z{-}1), \\ \tilde{M}(S)\bar{G}_w &= \frac{\bar{\square}(z^{-1})}{\bar{\square}(z^{-1})}, \\ \square_S(z^{-1}) &= \bar{\square}(z^{-1})\square(z^{-1}). \end{aligned} \tag{4.17}$$

When designing controllers for an actual plant where an *a priori* model G is known, there is advantage in specifying dynamic uncertainty in terms of S and thus working with a plant description $G(S)$. Note that for an actual plant $\bar{G} = G(S)$, the order of S may be higher than the order of the plant $\bar{G}$ or its actual model G. However through proper frequency-shaping, or equivalently through a good initial robust design for a stabilizing controller on the nominal plant, we may well have an S that can be fairly accurately described by a low order approximation, or perhaps a low gain S where its complexity is not of any real consequence. These ideas are best illustrated by an example.

Example. In this example we consider an eighth-order nominal plant G. The magnitude and phase plots are shown in Figure 4.2. This nominal plant has the characteristics of a band pass filter, more precisely an *elliptic filter*. The actual $G(S)$ is a perturbed version of G and the magnitude/phase plots are shown along side the nominal plant G.

An LQG nominal controller K (see details in Chapter 4) is designed for the nominal plant. This nominal controller stabilizes both the nominal plant and the

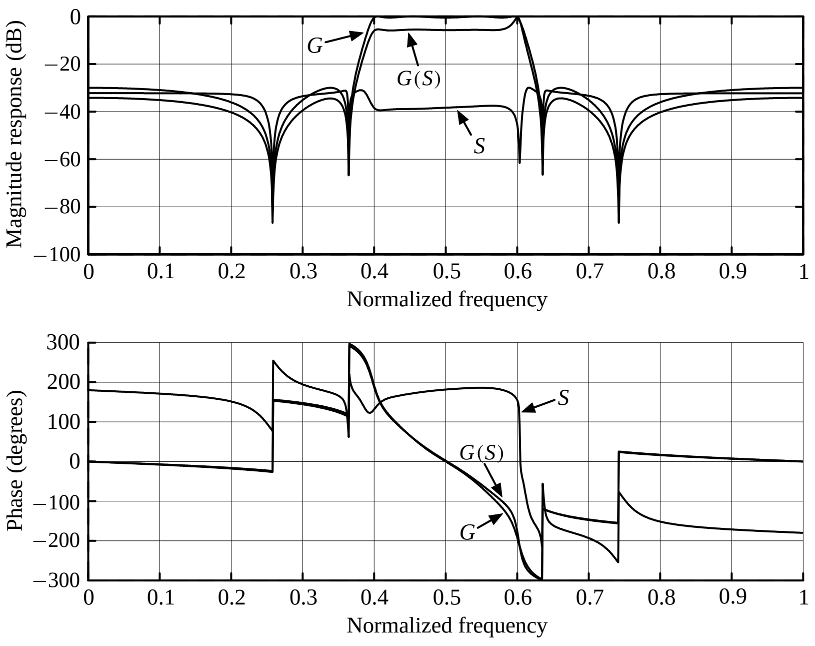


FIGURE 4.2. Magnitude/phase plots for G, S, and $G(S)$

actual plant $G(S)$. Using the techniques described in this section, we compute the corresponding S. The magnitude/phase plot for S is also shown alongside the plot for G and $G(S)$ in Figure 4.2.

At first glance, it may appear that S is too complex to work with, being perhaps more complex than either G or $G(S)$. From our derivation in this section, the complexity of S given any arbitrary G or $G(S)$ would be twice that of either G or $G(S)$. Intuitively, consider the adverse situation where the nominal plant G is a poor approximation to the actual plant. In this case, the S which together with G give the actual plant will have to first “cancel” the dynamics of G before “constructing” the dynamics of the actual plant. In such a case, we would expect the complexity of S to be the complexity of the nominal plant plus that of the actual plant, and with no possibility to even approximate S by a lower complexity object.

However, in the event that the nominal plant is a good approximation of the actual plant in a certain frequency band, the situation could be rather less daunting since some of the complexity in S could be of negligible significance since its magnitude would be relatively small. Take the present case. On close examination of the plot, one quickly realizes that the overall magnitude response of S is very small compared to either the nominal plant or actual plant. In this case, we see a dip of about 25 dB. In fact, as long as the nominal plant is a reasonably good estimate of the actual plant, the resultant S is going to be small in gain.

In fact, it turns out that a good gauge of S, as to its significant complexity, is

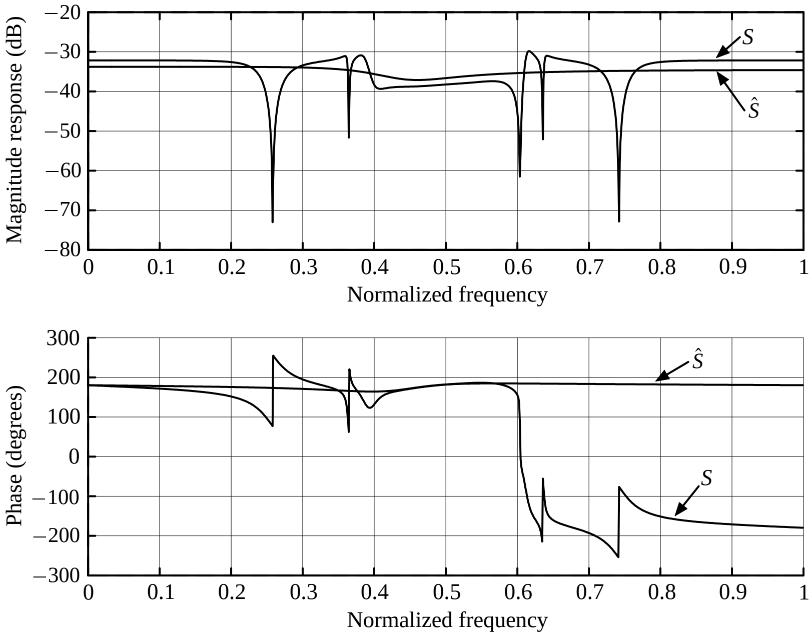


FIGURE 4.3. Magnitude/phase plots for S and a second order approximation for $\hat{S}$

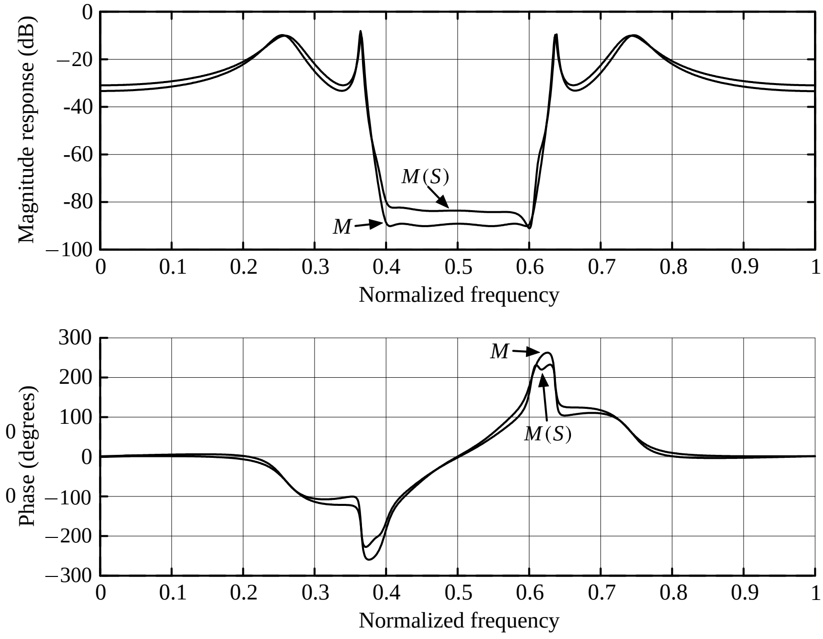


FIGURE 4.4. Magnitude/phase plots for M and $M(S)$

whether the nominal controller can robustly stabilize both G and $G(S)$. In the event that K can stabilize G and $G(S)$ simultaneously, then S is likely to be a small gain object which can then be approximated without significant loss by a low complexity object, denoted $\hat{S}$. Figure 4.3 shows S, and its approximation $\hat{S}$ by a second order transfer function. The magnitude/phase response of M and $M(S)$ of (4.9) is shown in Figure 4.4. Recall that M and $\tilde{M}(S)$ are frequency-shaping transfer functions for $(G(S) - G)$, which gives us S.

To give a better feel of what may arise in practice, let us consider another perturbation of the nominal plant G, giving rise to a new plant $G(S)$, parameterized by S. The magnitude/phase plot for the new $G(S)$ is given in Figure 4.5 alongside G and the corresponding S. Notice that for this case, there is a poor approximation of $G(S)$ by G, so that the nominal controller K no longer stabilizes $G(S)$. Note that the new S is no longer a small gain object. It is no longer possible to ignore the dips in the frequency response and use a low order transfer function to approximate S.

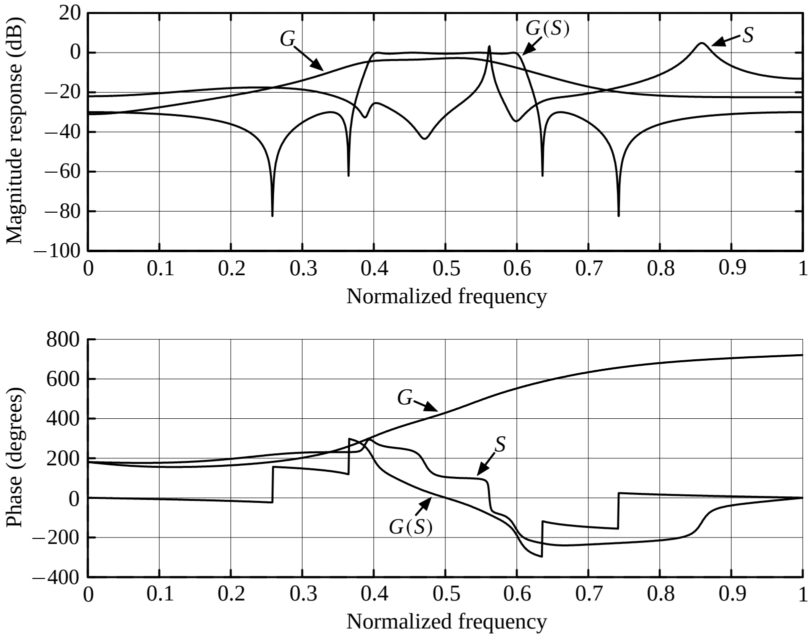


FIGURE 4.5. Magnitude/phase plots for the new $G(S)$, S and G

Robust Stabilization

In this subsection, we consider the class of plants $G(S)$, $S \in RH_\infty$ stabilizable by a controller K. Let us also consider the class of controllers $K(Q)$, $Q \in RH_\infty$, parameterized as in (2.5.2) of the previous chapter, such that $(K(Q), G)$ is a stabilizing pair. We investigate the stability of the closed-loop system formed by $G(S)$ and $K(Q)$. We have the following result:

Theorem 4.2. *Let (G, K) be a stabilizing plant controller pair. Let (G, K) have coprime factor representations as in* (4.1) *and* (4.2)*. Consider*

$$G(S) = (N + VS)(M + US)^{-1} = (\tilde{M} + S\tilde{U})^{-1}(\tilde{N} + S\tilde{V}),$$

and

$$K(Q) = (U + MQ)(V + NQ)^{-1} = (\tilde{V} + Q\tilde{N})^{-1}(\tilde{U} + Q\tilde{M}).$$

with $Q, S \in R_p$. The pair $(G(S), K(Q))$ is stabilizing if and only if the pair (Q, S) is stabilizing as depicted in Figure 4.6. In particular:

$$\begin{aligned}&\begin{bmatrix} I & -K(Q) \\ -G(S) & I \end{bmatrix}^{-1} \\ &= \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} + \begin{bmatrix} M & U \\ N & V \end{bmatrix} \left\{ \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} - I \right\} \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix}. \end{aligned} \tag{4.18}$$

Proof. Equation (4.18) is derived as follows.

$$\begin{aligned}\begin{bmatrix} I & -K(Q) \\ -G(S) & I \end{bmatrix}^{-1} &= \begin{bmatrix} I & -\tilde{V}(Q)^{-1}\tilde{U}(Q) \\ -\tilde{M}(S)^{-1}\tilde{N}(S) & I \end{bmatrix}^{-1} \\ &= \left\{ \begin{bmatrix} \tilde{V}(Q)^{-1} & 0 \\ 0 & \tilde{M}(S)^{-1} \end{bmatrix} \begin{bmatrix} \tilde{V}(Q) & -\tilde{U}(Q) \\ -\tilde{N}(S) & \tilde{M}(S) \end{bmatrix} \right\}^{-1} \\ &= \left\{ \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix} \begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \right\}^{-1} \begin{bmatrix} \tilde{V}(Q) & 0 \\ 0 & \tilde{M}(S) \end{bmatrix} \\ &= \begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix}^{-1} \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} \\ &\quad \times \left\{ \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix} \begin{bmatrix} \tilde{V} & 0 \\ 0 & \tilde{M} \end{bmatrix} + \begin{bmatrix} Q\tilde{N} & Q\tilde{M} \\ S\tilde{V} & S\tilde{U} \end{bmatrix} \right\}. \end{aligned}$$

Finally using the double Bezout identity:

$$\begin{aligned}&\begin{bmatrix} I & -K(Q) \\ -G(S) & I \end{bmatrix}^{-1} \\ &= \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} \tilde{V} & 0 \\ 0 & \tilde{M} \end{bmatrix} + \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} \begin{bmatrix} 0 & Q \\ S & 0 \end{bmatrix} \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix}, \end{aligned} \tag{4.19}$$

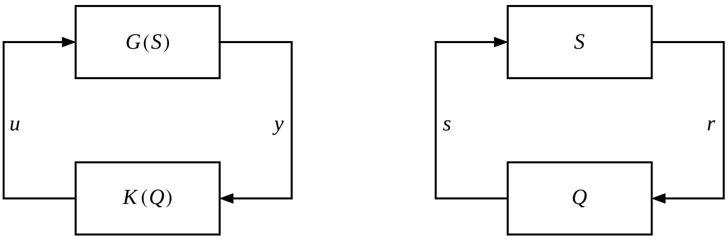


FIGURE 4.6. Robust stability property

which yields the desired result (4.18) since

$$\begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} \tilde{V} & 0 \\ 0 & \tilde{M} \end{bmatrix} = \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1}, \tag{4.20}$$

and trivially

$$\begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} \begin{bmatrix} 0 & Q \\ S & 0 \end{bmatrix} = \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} - I. \tag{4.21}$$

From this expression it is readily concluded using arguments as in deriving Theorem 2.5.1, that under the assumption that (G, K) is a stabilizing pair,

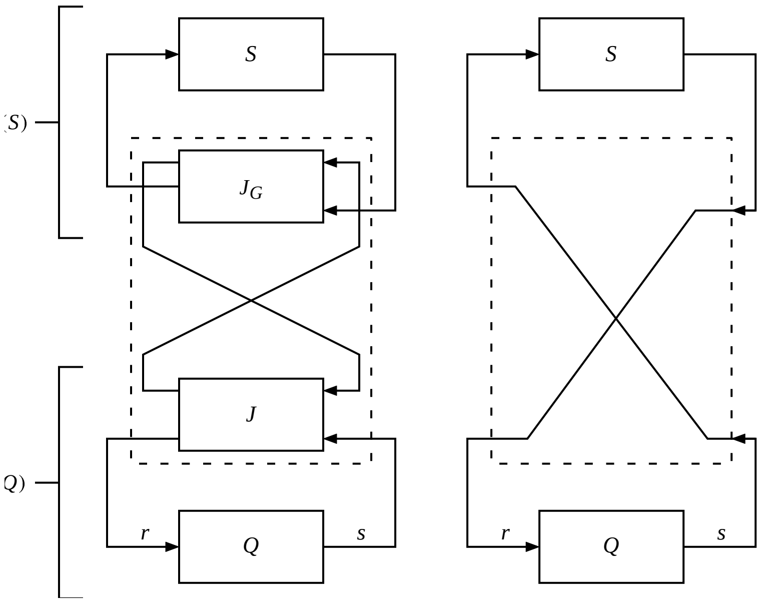


FIGURE 4.7. Cancellations in the J, J_G connections

$(G(S), K(Q))$ is stabilizing if and only if (Q, S) is stabilizing. This establishes the result. □

Actually, the situation of the above Theorem 4.2 can be viewed by combining Figures 4.1 and 2.5.1. A little effort shows that the block consisting of J and J_G has a transform $\left[\begin{smallmatrix} 0 & I \\ I & 0 \end{smallmatrix}\right]$.

This Theorem 4.2 is at the heart of the iterated designs to be discussed in later chapters. It leads to the following idea. An initial controller $K = K(0)$ stabilizing both model $G(0)$ and plant $G(S)$ can be refined by identifying a new model $G(S_1)$ and selecting an appropriate Q_1 stabilizing S_1 to yield $K(Q_1)$. In order to exploit the idea it is important to be able to identify S.

Closed-loop S Interpretation

We consider the closed-loop system $G(S)$, $K(Q)$ and show how S can be interpreted in terms of signals that can be obtained from the closed loop. This is an essential step in presenting an identification scheme for S, which is postponed until Chapter 5, where iterated designs are discussed.

The following result is crucial. Refer to Figure 4.8.

Lemma 4.3. *With reference to Figure 4.8, let* (G, K) *be a stabilizing pair. Let* $G(S)$ *represent any plant stabilizable by* K*; (see* (4.1), (4.2) *and* (4.3) *for a parameterization). The transfer function block* J *is given by* (2.5.10) *repeated as*

$$J = \begin{bmatrix} K & \tilde{V}^{-1} \\ V^{-1} & -V^{-1}N \end{bmatrix}.$$

The system W *with inputs* (w_1, w_2, s) *and outputs* (e_1, e_2, r) *has a stable transfer*

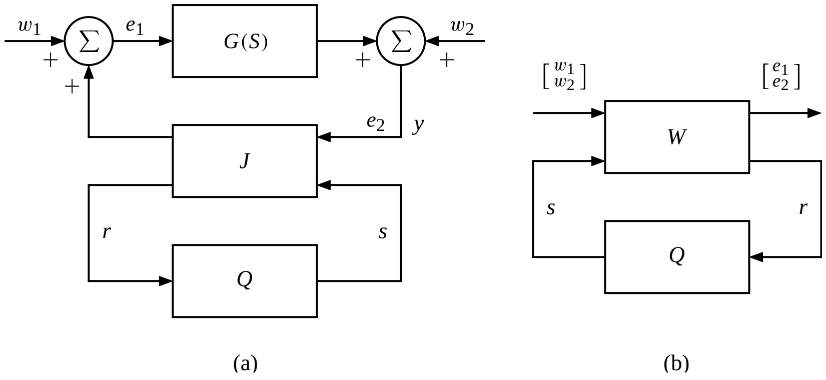

w_1
Σ
e_1
$G(S)$
Σ
w_2
$+$
$+$
$+$
$+$
e_2
y
J
r
s
Q
(a)
$\left[\begin{smallmatrix} w_1 \\ w_2 \end{smallmatrix}\right]$
$\left[\begin{smallmatrix} e_1 \\ e_2 \end{smallmatrix}\right]$
W
s
r
Q
(b)

FIGURE 4.8. Closed-loop transfer function

function:

$$W = \begin{bmatrix} \begin{bmatrix} I & -K \\ -G(S) & I \end{bmatrix}^{-1} & \begin{bmatrix} M(S) \\ N(S) \end{bmatrix} \\ \begin{bmatrix} \tilde{N}(S) & \tilde{M}(S) \end{bmatrix} & S \end{bmatrix} \in RH_\infty. \tag{4.22}$$

In particular, the transfer function from signal s to signal r is S. Moreover, the systems $(G(S), J, Q)$ and (W, Q) of Figure 4.8 are internally stable.

Proof. From Figure 4.8, simple manipulations show that W is given by

$$W = \begin{bmatrix} \begin{bmatrix} \Delta_1 & \Delta_1 K \\ \Delta_2 G(S) & \Delta_2 \end{bmatrix} & \begin{bmatrix} \Delta_1 \tilde{V}^{-1} \\ \Delta_2 G(S) \tilde{V}^{-1} \end{bmatrix} \\ \begin{bmatrix} V^{-1}\Delta_2 G(S) & V^{-1}\Delta_2 \end{bmatrix} & V^{-1}\Delta_2 G(S)\tilde{V}^{-1} - V^{-1}N \end{bmatrix}.$$

$\Delta_1 = (I - KG(S))^{-1}$ and $\Delta_2 = (I - G(S)K)^{-1}$. Now utilizing the double Bezout identity (4.2) and (4.1) we have

$$\begin{aligned} V^{-1}(I - G(S)K)^{-1}G(S)\tilde{V}^{-1} - V^{-1}N &= \tilde{N}(S)\tilde{V}^{-1} - V^{-1}N \\ &= (\tilde{N} + S\tilde{V})\tilde{V}^{-1} - V^{-1}N \\ &= S, \end{aligned} \tag{4.23}$$

and the other identifications to yield the desired result (4.22). The stability results follow from the definition of closed-loop stability, and closed-loop stability results of Chapter 2. (The reader can check the details). □

An immediate implication of the lemma is that the (matrix) transfer function from the input s to the output r is S. In fact information about S can be deduced by observing the signals r and s. This leads naturally to an identification problem where the uncertainty S can be directly identified through observations of r and s. This is developed in Chapter 5, but suffice it to say here, for the earlier ARMAX example (4.12)–(4.17), that we have

$$\square_S(z^{-1})r = \square_S(z^{-1})s + \square_S(z^{-1})w. \tag{4.24}$$

Generalizations of the above results for the case of the situation depicted in Figure 4.9 and denoted $(P(S), J, Q)$ are straightforward. They are also a natural generalization of the results developed for the scheme (P, J, Q) of Figure 2.5.5. Indeed, as expected,

$$\begin{aligned} T(S) &= \begin{bmatrix} T_{11}(S) & T_{12}(S) \\ T_{21}(S) & T_{22}(S) \end{bmatrix} \\ &= \begin{bmatrix} P_{11}(S) + P_{12}(S)U\tilde{M}(S)P_{21}(S) & P_{12}(S)M(S) \\ \tilde{M}(S)P_{21}(S) & S \end{bmatrix}. \end{aligned} \tag{4.25}$$

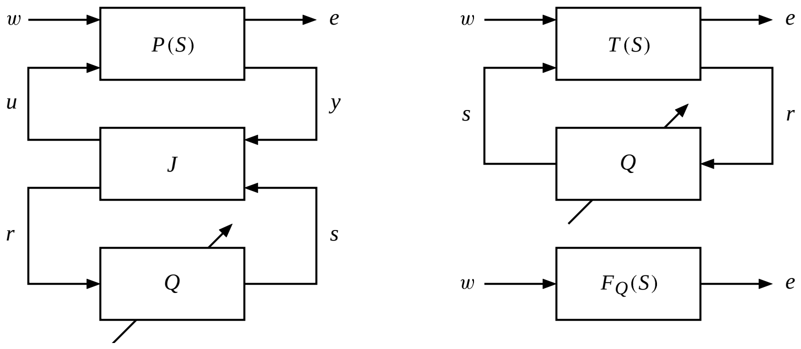


FIGURE 4.9. Plant/noise model

We see that the arrangements of Figure 4.9 can be viewed as $T(S)$ in feedback with a controller $\left[\begin{smallmatrix} 0 & 0 \\ 0 & Q \end{smallmatrix}\right]$, since there is zero feedback from e to w. Thus closed-loop internal stability of the systems of Figure 4.9 requires the stability of the pair $\left(\left[\begin{smallmatrix} 0 & 0 \\ 0 & Q \end{smallmatrix}\right], T(S)\right)$. This is a stronger condition than merely requiring that (Q, S) is stabilizing.

That stability of the pair $\left(\left[\begin{smallmatrix} 0 & 0 \\ 0 & Q \end{smallmatrix}\right], T(S)\right)$ is also a sufficient condition for internal stability of the systems of Figure 4.9 and follows by exploiting the necessary condition that (Q, S) is stabilizing and thus $(K(Q), G(S))$ is stabilizing. First observe that the responses r, s, e to bounded input w are bounded under the assumption, which means that $y = Vr + VNs$ is also bounded. Now with $(K(Q), G(S))$ stabilizing, bounded disturbances at y give rise to bounded response to u, so that the disturbance w gives rise to bounded responses r, s, y, u, e. It is not difficult to see that bounded disturbances in r, s, y, u, w give bounded responses in e, and stability of each of the systems of Figure 4.8 imply the stability of the other.

Plants Regulated by a Controller

As we have seen already in the previous chapter, regulators are essentially stabilizing controllers for plants augmented by the deterministic disturbance class model, see (2.1) - (2.6). The augmentations can be absorbed in P, and indeed $P(S)$. This allows us to apply the theory for the class of all plants stabilized by a controller, and thereby deduce for the unaugmented P, or $P(S)$, the appropriate results for the class of plants regulated by a controller. We do not proceed further to spell out details here. However, we should stress that if the class of disturbances is itself uncertain, and therefore S dependent, then the augmentations will be also S dependent. This in turn will raise concerns about the existence of any finite dimensional controller to achieve the regulation. For certain nongeneric S, say belonging to a discrete set, it may be possible for the pair $\left(\left[\begin{smallmatrix} 0 & 0 \\ 0 & Q \end{smallmatrix}\right], T(S)\right)$ to be stabilizing for some Q. Here $T(S)$ in (4.25) is now the plant augmented with the disturbance model.

In dealing with the question of *robust regulators* it is usual to assume that the class of disturbances is precisely known and thus not S dependent. Adaptive schemes, as developed in Chapter 6, overcome this problem to some extent.

Main Points of Section

In this section a special class of frequency-shaped uncertainty is introduced. The uncertainty is characterized by a (matrix) transfer function S which, when restricted to RH_∞, also parameterizes the class of all plants stabilizable by a controller. The (matrix) transfer function S turns out to be the deviation of the actual plant from a nominal plant in the class, frequency-shaped to emphasize the operating frequency band of interest. Robust stabilization results arising from the parameterization in term of S are also presented. It is demonstrated that the (matrix) transfer function S can be accessed via signals measurable in the closed-loop system. Finally, we point out the relevance of the results to characterize the class of plants with deterministic disturbances regulated by a controller. Stabilization theory is applied to plants augmented by the disturbance models.

3.5 State Space Representation [‡]

In this section, we consider again all plants stabilized by a given controller K, denoted $G(S)$, where S is an arbitrary stable proper rational transfer function. The results presented are at a more advanced level than most of the book. They are of interest in their own right but the casual reader need only see that deeper results do exist in these waters. In particular, we are interested in providing convenient state space representations for any plant $\bar{G}$ expressed as $G(S)$ where $G = G(0)$ is a nominal model and S is a parameterization. The question of minimizing the order of realizations is addressed. A constructive approach for any plant $\bar{G}$ and a nominal model G to achieve a minimal degree parameterization S is presented.

The material in this section is used in the analysis of the adaptive controller based on the Q-parameterization in Chapter 6.

Consider a controller $K = UV^{-1} = \tilde{V}^{-1}\tilde{U}$ with $U, V, \tilde{U}, \tilde{V} \in RH_\infty$ and a nominal plant $G = NM^{-1} = \tilde{M}^{-1}\tilde{N}$ with $M, N, \tilde{M}, \tilde{N} \in RH_\infty$. Let the Bezout identity (4.2) be satisfied. We consider first how a state space realization can be devised for $\bar{G}$ in the form $G(S)$ in terms of the realizations:

$$Z := \begin{bmatrix} M & U \\ N & V \end{bmatrix} : \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right], \tag{5.1}$$

[‡]This section may be omitted on first reading, or at least the proofs which are relatively technical.

and

$$S : \left[\begin{array}{c|c} A_S & B_S \\ \hline C_S & D_S \end{array}\right]. \tag{5.2}$$

Here Z is referred to as a *stable linear fractional representation* of G. Assume that $\Delta := (D_{11} + D_{12}D_S)^{-1}$ exists. We can then represent $\bar{G} = G(S)$ as $G(S) = (N + VS)(M + US)^{-1}$, where:

$$\begin{bmatrix} M + US \\ N + VS \end{bmatrix} = \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} I \\ S \end{bmatrix} : \left[\begin{array}{cc|c} A & B_2C_S & B_1 + B_2D_S \\ 0 & A_S & B_S \\ \hline C_1 & D_{12}C_S & D_{11} + D_{12}D_S \\ C_2 & D_{22}C_S & D_{21} + D_{22}D_S \end{array}\right].$$

Manipulations using the techniques of Chapter 2 tell us that $\bar{G}$, here $G(S)$, has a realization

$$\bar{G} : \left[\begin{array}{cc|c} A - \Omega_1\Delta C_1 & B_2C_S - \Omega_1\Delta D_{12}C_S & \Omega_1\Delta \\ -B_S\Delta C_1 & A_S - B_S\Delta D_{12}C_S & B_S\Delta \\ \hline -\Omega_2\Delta C_1 + C_2 & -\Omega_2\Delta D_{12}C_S + D_{22}C_S & \Omega_2\Delta \end{array}\right], \tag{5.3}$$

where $\Omega_1 = B_1 + B_2D_S$ and $\Omega_2 = D_{21} + D_{22}D_S$. Given this realization of $\bar{G} = G(S)$, we have the following lemma.

Lemma 5.1. *Consider the state space realization for Z, S and $\bar{G} = G(S)$ in* (5.1)–(5.3). *Let $\left[\begin{smallmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{smallmatrix}\right]$ and $D_{11} + D_{12}D_S$ be invertible. Let S of* (5.2) *be a minimal realization. Then any uncontrollable modes of $\bar{G}$ as realized in* (5.3) *are also poles of Z, and any unobservable modes of $\bar{G}$ of* (5.3) *are also poles of Z^{-1}.*

Proof. Let λ_0 be an uncontrollable mode of $\bar{G}$. Then noting (5.3), there exists a nonzero vector $x' = [\, x_1' \; x_2' \,]$ such that

$$\begin{bmatrix} x_1' & x_2' \end{bmatrix} \begin{bmatrix} \lambda_0 I - A + \Omega_1\Delta C_1 & -B_2C_S + \Omega_1\Delta D_{12}C_S & \Omega_1\Delta \\ B_S\Delta C_1 & \lambda_0 I - A_S + B_S\Delta D_{12}C_S & B_S\Delta \end{bmatrix} = 0.$$

Post multiplying with the invertible matrix:

$$\begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -C_1 & -D_{12}C_S & \Delta^{-1} \end{bmatrix},$$

yields the equivalent expression:

$$\begin{bmatrix} x_1' & x_2' \end{bmatrix} \begin{bmatrix} \lambda_0 I - A & -B_2C_S & B_1 + B_2D_S \\ 0 & \lambda_0 I - A_S & B_S \end{bmatrix} = 0. \tag{5.4}$$

This implies that $x_1'(\lambda_0 I - A) = 0$. Since $x \neq 0$, we can claim that $x_1 \neq 0$ because, otherwise, (5.4) would result in $x_2' \left[\, \lambda_0 I - A_S \;\; B_S \,\right] = 0$, which is contradictory to the assumption that (A_S, B_S) is controllable. In this way, it follows that λ_0 is an eigenvalue of A. Also, if λ_0 is an uncontrollable mode of multiplicity k, then there exist k linearly independent vectors $x_i' = [\, x_{i1}' \;\; x_{i2}' \,]$, $i = 1, \ldots, k$, satisfying (5.4), which in turn tells us that $x_{i1}, \ldots, x_{k1}$ are linearly independent eigenvectors of A associated with λ_0. Hence, uncontrollable modes of $\bar{G}$ are poles of Z as claimed.

Now observe that the inverse of $T := D_{22} - (D_{21} + D_{22}D_S)\Delta D_{12}$ exists as the (2, 2)-block element of the block matrix

$$\begin{bmatrix} D_{11} + D_{12}D_S & D_{12} \\ D_{21} + D_{22}D_S & D_{22} \end{bmatrix}^{-1}.$$

Thus, denoting $\tilde{A}_S = A_S - B_S\Delta D_{12}C_S$, and again working with (5.3), the matrix

$$\begin{bmatrix} \lambda I - A + (B_1 + B_2D_S)\Delta C_1 & (B_1 + B_2D_S)\Delta D_{12}C_S - B_2C_S \\ B_S\Delta C_1 & \lambda I - \tilde{A}_S \\ C_2 - (D_{21} + D_{22}D_S)\Delta C_1 & D_{22}C_S - (D_{21} + D_{22}D_S)\Delta D_{12}C_S \end{bmatrix} \tag{5.5}$$

is equivalent to

$$\begin{bmatrix} \lambda I - A + \Omega_1\Delta C_1 + [B_2 - \Omega_1\Delta D_{12}]T^{-1}[C_2 - \Omega_2\Delta C_1] & 0 \\ B_S\Delta C_1 & \lambda I - \tilde{A}_S \\ T^{-1}[C_2 - \Omega_2\Delta C_1] & C_S \end{bmatrix},$$

which in turn can be reorganized as

$$\begin{bmatrix} \lambda I - A + \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}^{-1} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} & 0 \\ B_S\Delta C_1 & \lambda I - \tilde{A}_S \\ T^{-1}\,[C_2 - \Omega_2\Delta C_1] & C_S \end{bmatrix}. \tag{5.6}$$

In view of the equivalence between (5.5) and (5.6), and the fact that the poles of Z^{-1} consist of eigenvalues of the matrix

$$A - \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}^{-1} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix},$$

one can show that the unobservable poles of $\bar{G}$ are poles of Z^{-1}, using the same argument concerning uncontrollable modes of $\bar{G}$ being poles of Z. □

We can now prove the following lemma which is of interest in relating the orders of a given plant $G(s)$ to the order of S and Z.

Lemma 5.2. *With the same notation and hypotheses as in Lemma 5.1, and let C_s be full row rank:*

1. *The state-space realization* (5.3) *of* $\bar{G} = G(S)$ *is stabilizable and detectable (with no unobservable or uncontrollable unstable modes).*

2. *Let* $\delta(\cdot)$ *denote the* McMillan degree *(the degree of a minimal state realization), and m the number of rows of* C. *Assume that* (A, C_1) *is observable and that*

$$\operatorname{rank}\begin{bmatrix} \lambda I - A & B_1 & B_2 & 0 \\ C_1 & D_{11} & 0 & D_{12} \end{bmatrix} = \delta(Z) + m, \quad \text{for all } \lambda \in \mathsf{C}. \tag{5.7}$$

Then there holds

$$\delta(\bar{G}) = \delta(Z) + \delta(S), \tag{5.8}$$

Proof. Item 1 is a direct consequence of Lemma 5.1 since both Z and Z^{-1} are stable.

To prove Item 2, observe from Lemma 5.1 that (5.8) holds if the realization (5.3) does not contain unobservable or uncontrollable modes. Write the state matrix of the realization (5.3) in the form

$$\begin{aligned}
&\begin{bmatrix} A - (B_1 + B_2 D_S)\Delta C_1 & B_2 C_S - (B_1 + B_2 D_S)\Delta D_{12} C_S \\ -B_S \Delta C_1 & A_S - B_S \Delta D_{12} C_S \end{bmatrix} \\
&\qquad = \begin{bmatrix} A - B_1 \Delta C_1 & -B_1 \Delta D_{12} C_S \\ 0 & 0 \end{bmatrix} \\
&\qquad\quad + \begin{bmatrix} -B_2 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} D_S & C_S \\ B_S & A_S \end{bmatrix} \begin{bmatrix} \Delta C_1 & -\Delta D_{12} C_S \\ 0 & I \end{bmatrix}.
\end{aligned} \tag{5.9}$$

It is easy to see that the observability of (A, C_1) implies that of

$$\left(\begin{bmatrix} A - B_1 \Delta C_1 & -B_1 \Delta D_{12} C_S \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} \Delta C_1 & -\Delta D_{12} C_S \\ 0 & I \end{bmatrix} \right).$$

It will be shown that the pair

$$\left(\begin{bmatrix} A - B_1 \Delta C_1 & -B_1 \Delta D_{12} C_S \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} -B_2 & 0 \\ 0 & I \end{bmatrix} \right) \tag{5.10}$$

is controllable for generic (C_S, D_S). In fact, the pair (5.10) is controllable if and only if

$$\operatorname{rank}\begin{bmatrix} \lambda I - A + B_1 \Delta C_1 & B_1 \Delta D_{12} C_S & B_2 \end{bmatrix} = \delta(G), \quad \text{for all } \lambda \in \mathsf{C}, \tag{5.11}$$

which is equivalent to

$$\operatorname{rank}\begin{bmatrix} \lambda I - A & -B_2 & 0 & -B_1 \\ C_1 & 0 & D_{12}C_S & D_{11} + D_{12}D_S \end{bmatrix} = \delta(G) + m, \qquad \text{for all } \lambda \in \mathsf{C}, \tag{5.12}$$

under the constraint that $D_{11} + D_{12}D_S$ is invertible. Because $\delta(G) = \delta(Z)$ and the fact that C_S has full rank, (5.12) follows (5.7). Thus from Davison and Wang (1989), the left hand side in (5.9), regarded as a closed-loop state matrix under a static output feedback, has in the generic case no uncontrollable or unobservable modes. □

Item 2 of Lemma 5.2 tells us that given a Z satisfying certain properties, the order of a plant $\bar{G} = G(S)$ generated via this Z generically equals the sum of the orders of S and Z. For a high order plant, here denoted $\bar{G} = G(S)$, it is often convenient to work first with its model, here denoted $G(0)$ and then with the frequency-shaped modeling error S. This method can divide a complex design problem into two or more simpler problems. Needless to say, the choice of model is important for success of the method in terms of efficiency and computational reduction. The acceptable choice should be one resulting in S which has a lower order or can be approximated by some $\hat{S}$ of low order. Obviously, the most ideal case is where the complexity of S is the plant's complexity minus the model's complexity. This motivates a question as to whether there exists a model for a given plant such that the order of the plant is the sum of the orders of the model G and of S and how such G can be constructed if it exists. To address this issue, we need to consider a *minimal stable linear fractional representation* for a plant, whose definition is given as follows.

Definition. In the notation above, a plant $\bar{G}$ is said to have a minimal stable linear fractional representation if there exists a stable linear fractional representation Z belonging to the set of all such representations for a nominal plant G, and an associated system $S \in R_p$ such that $\bar{G} = G(S)$ and

$$\delta(Z), \delta(S) > 0 \quad \text{and} \quad \delta(\bar{G}) = \delta(Z) + \delta(S). \tag{5.13}$$

It turns out that the problem of existence of such minimal representations for a given plant is closely related to the problem of the existence of a minimal factorization of a transfer function. The latter problem has been addressed and solved, see Bart, Gohberg, Kaashoek and Dooren (1980), Dooren and Dewilde (1981). Recall that the factorization $R = R_1 R_2$ is said to be minimal if $\delta(R) = \delta(R_1) + \delta(R_2)$ where R, R_1, R_2 are square rational matrices. We proceed through this deeper theory as follows.

From Wonham (1985) we recall the following definition:

Definition. Consider $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$. Denote the state space by X. Also, $\Phi \subset X$ is called a stable invariant subspace for the pair (A, B) provided $(A+BF)\Phi \subset \Phi$ for some F such that $(A + BF)$ is stable.

The following two lemmas are needed in the proof of the main result in this section.

Lemma 5.3. *Given a nominal plant $G \in R_p$. Then $Z = \left[\begin{smallmatrix} M & U \\ N & V \end{smallmatrix}\right]$ with $Z, Z^{-1} \in RH_\infty$ satisfies*

$$\begin{aligned} G &= NM^{-1} \\ \delta(G) &= \delta(Z) \end{aligned} \tag{5.14}$$

if and only if there exists a minimal realization $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ of G together with a stabilizing state feedback gain F and a stabilizing output injection H such that

$$Z : \left[\begin{array}{c|cc} A+BF & B & -H \\ \hline F & I & 0 \\ C+DF & D & I \end{array}\right]. \tag{5.15}$$

Proof. The if part was established in Section 2.4. In particular compare with (2.4.18). Now it is assumed that the block matrix Z satisfies the conditions (5.14) and has the following minimal realization

$$Z : \left[\begin{array}{c|cc} \bar{A} & \bar{B}_1 & \bar{B}_2 \\ \hline \bar{C}_1 & I & 0 \\ \bar{C}_2 & \bar{D} & I \end{array}\right]. \tag{5.16}$$

By definition of the coprime factorization, Z and $Z^{-1} \in RH_\infty$, then $\bar{A}$ and $\bar{A} - \bar{B}_1\bar{C}_1 - \bar{B}_2\bar{C}_2 + \bar{B}_2\bar{D}\bar{C}_1$ are stable. Setting

$$\begin{gathered} A = \bar{A} - \bar{B}_1\bar{C}_1, \qquad B = \bar{B}_1, \qquad C = \bar{C}_2 - D\bar{C}_1, \\ F = \bar{C}_1, \qquad H = -\bar{B}_2, \end{gathered} \tag{5.17}$$

one can check that $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ is a minimal realization of G, that F is a stabilizing state feedback gain and H a stabilizing output injection, and that (5.15) holds. $\square$

Note that for Z given in (5.15), the conditions for (5.8) to hold generically amounts to the minimality of the system (F, A, H).

The following lemma is proved in Bart et al. (1980) or Dooren and Dewilde (1981) and the interested reader is referred to the papers for the proof.

Lemma 5.4. *Consider an $n \times n$ invertible matrix transfer function $\bar{G}$ with minimal realization:*

$$\bar{G} : \left[\begin{array}{c|c} \bar{A} & \bar{B} \\ \hline \bar{C} & \bar{D} \end{array}\right]. \tag{5.18}$$

Then there exists a minimal factorization for $\bar{G}$ if there exist independent subspaces $\bar{X}_1$ and $\bar{X}_2$ such that

$$\begin{aligned} \bar{A}\bar{X}_1 &\subset \bar{X}_1, \\ (\bar{A} - \bar{B}\bar{D}^{-1}\bar{C})\bar{X}_2 &\subset \bar{X}_2, \\ \bar{X}_1 \oplus \bar{X}_2 &= \bar{X}, \end{aligned} \tag{5.19}$$

where $\bar{X}$ denotes the state space, and $\oplus$ denotes the direct sum.

Theorem 5.5. *A given plant $\bar{G} \in R_p$ with a minimal realization* (5.18) *has a minimal stable linear fractional representation Z if and only if there exist two stable invariant subspaces X_1 and X_2 associated with $(\bar{A}, \bar{B})$ and $(\bar{A}', \bar{C}')$, respectively, such that*

$$X_1 \oplus X_2^{\perp} = X. \tag{5.20}$$

where X denotes the state space of the realization (5.18) *and $Y^{\perp}$ denotes the orthogonal space of Y.*

Proof.

1. **Necessity.**
   Assume that there exists a nontrivial $Z_0 = \left[\begin{smallmatrix} M_0 & U_0 \\ N_0 & V_0 \end{smallmatrix}\right] \in RH_\infty$ with $Z_0^{-1} \in RH_\infty$ and a system $S \in R_p$ such that

   $$\bar{G} = (N_0 + V_0 S)(M_0 + U_0 S)^{-1} \quad \text{and} \quad \delta(\bar{G}) = \delta(Z_0) + \delta(S). \tag{5.21}$$

   Since one can associate S with a unit[§] $Z_S = \left[\begin{smallmatrix} M_S & U_S \\ N_S & V_S \end{smallmatrix}\right]$ where $N_S M_S^{-1}$ being an right coprime factorization of S and $\delta(Z_S) = \delta(S)$, it is seen that $\bar{G}$ has the following right coprime factorization

   $$\bar{G} = (N_0 M_S + V_0 N_S)(M_0 M_S + U_0 N_S)^{-1}. \tag{5.22}$$

   Define

   $$Z = \begin{bmatrix} M & U \\ N & V \end{bmatrix} := \begin{bmatrix} M_0 & U_0 \\ N_0 & V_0 \end{bmatrix} \begin{bmatrix} M_S & U_S \\ N_S & V_S \end{bmatrix}. \tag{5.23}$$

   Since $\delta(Z) \le \delta(Z_0) + \delta(Z_S) = \delta(\bar{G})$ and $\delta(Z) \ge \delta(\bar{G})$, it follows that Z satisfies (5.14) and

   $$\delta(Z) = \delta(Z_0) + \delta(Z_S). \tag{5.24}$$

[§]We say that $Z \in RH_\infty$ is a unit if also $Z^{-1} \in RH_\infty$.

Therefore, $Z = Z_0 Z_S$ is a minimal factorization of Z. By Lemma 5.3, there exists a minimal realization $\left[\begin{array}{c|c}\bar{\square} & \bar{\square} \\ \hline \bar{\square} & \bar{\square}\end{array}\right]$ of $\bar{G}$, a stabilizing state feedback gain F and a stabilizing output injection H such that

$$\left[\begin{array}{c|cc} \bar{\square} + \bar{\square}F & \bar{\square} & -H \\ \hline F & I & 0 \\ \bar{\square} + \bar{\square}F & \bar{\square} & I \end{array}\right] \tag{5.25}$$

is a minimal realization of Z. By Lemma 5.4, there exist subspaces Y_1 and Y_2 such that

$$\begin{array}{ll} \text{(a)} & (\bar{\square} + \bar{\square}F)Y_1 \subset Y_1 \\ \text{(b)} & (\bar{\square} + H\bar{\square})Y_2 \subset Y_2 \\ \text{(c)} & Y_1 \oplus Y_2 = X \end{array} \tag{5.26}$$

Evidently, (a) above implies that Y_1 is a stable invariant subspace of $(\bar{\square}, \bar{\square})$ while (b) implies that $Y_2^{\perp}$ is that of $(\bar{\square}', \bar{\square}')$. Since $(\bar{\square}, \bar{\square}, \bar{\square}, \bar{D})$ and $(\bar{A}, \bar{B}, \bar{C}, \bar{D})$ are minimal realizations of G, there exists a similarity transformation T such that

$$\bar{\square} = T^{-1}\bar{A}T, \qquad \bar{\square} = T^{-1}\bar{B}, \qquad \bar{\square} = \bar{C}T.$$

Let $X_1 = TY_1$ and $X_2 = (T^{-1})'Y_2^{\perp}$. Then it is not hard to see that X_1 and X_2 are two stable invariant subspaces X_1 and X_2 associated with $(\bar{A}, \bar{B})$ and $(\bar{A}', \bar{C}')$, respectively, and satisfy (5.20).

2. **Sufficiency.**
Choose F and H such that $\bar{A} + \bar{B}F$ and $\bar{A} + H\bar{C}$ are stable, and that

$$(\bar{A} + \bar{B}F)X_1 \subset X_1, \qquad (\bar{A}' + \bar{C}'H')X_2 \subset X_2, \tag{5.27}$$

where $X_1 \oplus X_2^{\perp} = X$. By Lemma 5.4 this implies that the Z with the minimal realization (5.15) has a minimal factorization, say $Z = Z_1 Z_2$ where $Z_i \in R_p$. Since Z is a unit in RH_∞ and there is no pole/zero cancellation between its two factors Z_1 and Z_2, Z_1 and Z_2 are units in RH_∞. Furthermore, there is no loss of generality in assuming that $Z_i,\ i = 1, 2$ are representations of $\bar{G}$. Letting $S := N_2 M_2^{-1}$ leads to $\bar{G} = G(S)$ derived using a model G_1 with factorization matrix Z_1, which implies

$$\delta(\bar{G}) \le \delta(Z_1) + \delta(S) \le \delta(Z_1) + \delta(Z_2).$$

From the minimality of the factorization $Z = Z_1 Z_2$ and the above inequalities, it follows that $\delta(\bar{G}) = \delta(Z_1) + \delta(S)$. In this way, the proof of the theorem is completed.

□

Corollary 5.6. *With the same assumption and notation as in Theorem 5.5. If $(\bar{A}, \bar{B})$ and $(\bar{A}', \bar{C}')$ have a common stable invariant subspace, then $\bar{G}$ has a minimal stable linear fractional representation Z.*

Construction of minimal Z

The proof of Theorem 5.5 apparently provides a two-step procedure to construct a minimal Z. The first step is to find a pair of stable invariant subspaces X_1 and X_2 of $(\bar{A}, \bar{B})$ and $(\bar{A}', \bar{C}')$, respectively, which satisfy (5.20). In the single-input, single-output case, this reduces to finding a nontrivial stable invariant subspace of $(\bar{A}, \bar{B})$ since $(\bar{A}, \bar{B})$ and $(\bar{A}', \bar{C}')$ have the same set of stable invariant subspaces. An iterative algorithm to compute the supremal $(\bar{A}, \bar{B})$-stable invariant subspace contained in a subspace is given in Wonham (1985). The second step involves constructing a minimal factorization of a representation Z. For algorithms to perform a minimal factorization of a rational matrix, see Dooren and Dewilde (1981).

Main Points of Section

In this section, we begin with a state-space representation of a stable linear fractional representation Z and show where pole/zero cancellations may occur for the state-space realization of the matrix transfer function $G(S)$ given Z. We then move on to derive a generic McMillan degree relation between the matrix transfer function of the plant $G(S)$ and the stable linear fractional representation Z derived based on the nominal representation G and S.

Finally, we consider the problem of minimal representation of any plant as a stable linear fractional representation of another plant. In particular, given an actual plant $\bar{G}$, we derive a nominal model G and an S such that the order of $\bar{G}$ equals the sum of the order of $Z = \begin{bmatrix} M & U \\ N & V \end{bmatrix}$, derived from G, and that of S. We also show that the necessary and sufficient conditions for solution to this problem can be derived in terms of (A, B)-stable invariant subspaces.

3.6 Notes and References

The material on signals, disturbances, disturbance responses, and their norms as performance measures is now quite standard in the optimal control literature. Likewise, the use of norms in both the time domain and the frequency domain to express plant performance and plant uncertainty is now standard in books focusing on robust control issues. For a parallel treatment see for example Boyd and Barratt (1991) and Green and Limebeer (1994).

The characterization of the class of plants $G(S)$ stabilized by a controller, parameterized in terms of an arbitrary $S \in RH_\infty$ is merely the dual of the stabilizing controller class $K(Q)$ parameterized in terms of arbitrary $Q \in RH_\infty$, see Chapter 2. The robust stabilization theory for the pair $(K(Q), G(S))$ was first developed in Tay, Moore and Horowitz (1989). Its generalization to plants $P(S)$ incorporating a disturbance response, that is to pair $(K(Q), P(S))$, is straightforward, as is the application of the results to robust regulation.

The main lemmas and theorems concerning state space representations for plants, and their nominal representations and uncertainties S are taken from Yan and Moore (1992).

Problems

1. Computing norms is not a trivial exercise. For the scalar signal $d_k = \sin(\omega k)$, compute the following norms: $\| \; \|_1$, $\| \; \|_\infty$, $\| \; \|_p$, rms value. How do the different norms compare?

2. Show that $\|u\|_{\text{rms}} \leq \sqrt{n}\, \|u\|_\infty$ for a vector valued signal
$$u = \left\{ u_k ;\, k = 1, 2, \ldots, u_k \in \mathsf{R}^n \right\}. \tag{6.1}$$

3. For scalar sequences $\{u_k, k = 1, 2, \ldots\ u_k \in \mathsf{R}\}$ consider:
$$\|u\|_{aa} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} |u_k| . \tag{6.2}$$
Compare $\|u\|_{aa}$ with $\|u\|_{\text{rms}}$. Is $\|u\|_{aa}$ a norm?

4. Show that the rms value of a signal u, such that $\lim_{k\to\infty} u_k = 0$, is zero.

5. Given
$$G : \left[\begin{array}{c|c} 1 & 1 \\ \hline 1 & 0 \end{array}\right] \quad \text{and} \quad \bar{G} = \left[\begin{array}{cc|c} 1-\alpha & 1 & 0 \\ 0 & 1 & 1 \\ \hline 1 & 0 & 0 \end{array}\right],$$
find a stabilizing controller K for the model G that also stabilizes $\bar{G}$. Express K as a function of α. Express $\bar{G} = G(S)$ with respect to this controller. Discuss (Q, S) and $(G(S), K(Q))$ in terms of α.

6. From Lemma 4.3 verify that the relationship between r, s and w_1, w_2 of Figure 4.8 is given from
$$r = Ss + \tilde{M}(S)w_2 + \tilde{N}(S)w_1. \tag{6.3}$$

CHAPTER **4**

Off-line Controller Design

4.1 Introduction

In Chapter 2, the parameterization of a stabilizing controller $K(Q)$ for a linear plant in terms of a stable (matrix) transfer function Q is discussed. It is shown that if Q spans the class of all stable proper (matrix) transfer functions, the entire class of stabilizing controllers for the plant is spanned. It can make sense in a controller design to optimize engineering objectives over this class, rather than over the class of all possible controllers, which of course includes the undesirable subclass of destabilizing controllers. In this chapter, we present various off-line optimal controller designs that, in addition to stabilization of the nominal plant will also allow the controller to track some reference signals and/or reject various classes of disturbances in some optimal fashion.

Before going into design methodologies, it is necessary to discuss performance criteria for optimization. Different applications have different control requirements. High performance control demanded for some feedback loops may not be important in other loops. As an example, in an oil refinery, typically less than 20% of the loops are critical and require well-tuned controllers. Many of the control loops are applied to buffer tanks where there is a less demanding control objective; simply to ensure that the tanks do not overflow or become empty. Of course, there is possibly scope to reduce the size of the tanks and thus refinery costs by improving control strategies.

For control loops where performance is relatively important, the control objective is usually to keep some or all the states close to desired references in the presence of disturbances. In regulation problems the references are constants, whereas in tracking problems the references may vary with time. The desire to keep the states close to the references is obviously justified. However what is not yet precise is the meaning of ‘close’. We caution that such is frequently debatable and is problem dependent.

There are many ways to specify the objective to be achieved by a controller. For less critical loops, this may be a visual inspection of how closely the states of the process follow the references. For more critical loops, specification becomes more precise. Performance requirements can be prescribed in the time domain, the frequency domain or both.

In this chapter we will first discuss some useful performance indices and the sort of situations where each of these indices is commonly used. We will then present some off-line controller design techniques that allow us to select a controller which minimizes these performance indices.

4.2 Selection of Performance Index

One of the first tasks in the design of a controller is to specify the performance to be achieved by the control system. Specifications can be in the frequency domain or the time domain. For the frequency domain approach, there are specifications on the gain and phase over the pass- and stop-band of the desired closed-loop system. For the time domain approach, there are specifications on the output behavior of the closed-loop system for given input sequences. The input sequences can have many forms. They can be deterministic signals such as constants, steps, ramps, sinusoids of known frequency or other known signals. Alternatively, they can be stochastic signals where only certain signal statistics are known. Examples of stochastic signal statistics are mean energy level, signal level variance and spectrum.

In classical controller design, the reference input is usually a known deterministic signal, such as the unit step function. In this case, the performance of the controller is specified in terms of the closed-loop process output response to the unit step, commonly just known as the step response.

In modern controller design where a linear dynamical model based approach is adopted, any known deterministic input signal is usually modeled as the output of an autonomous dynamical system with appropriate initial conditions. The resultant model of the plant in this case is then an augmented model which includes dynamics to generate the deterministic input signal from appropriate initial conditions. The composite model is constructed so that the tracking error is an output of this model. A significant area for modern controller design is for tracking or regulation performance when the inputs are stochastic in nature and therefore cannot be modeled in this way. One can use a stochastic model such as a linear dynamical system driven by noise, as for example white noise. In this case, a controller is designed to minimize the norm of the error signal between some desired trajectory and the process output. Let us first recall for the sake of completeness a typical step response specification in classical controller design before moving on to examine error signal specification.

Step Response Specification

Reference signals that remain constant for a long period of time and change to a new value in a relatively short period of time are usually represented for control design purposes as a unit step.

The steady-state closed-loop unit step response of the process output is usually specified by an asymptotic tracking requirement written as follows

$$\lim_{k\to\infty} y_k = 1. \tag{2.1}$$

This condition requires that in steady state there be no offset of the process output y_k from the reference signal. That is, the difference between y_k and the constant reference must eventually reach zero.

The *transient response* on the other hand is commonly specified by some or all of six attributes; namely *delay time, rise time, peak time, maximum overshoot, maximum undershoot* and *settling time*. These six attributes are defined as follows.

1. Delay time t_d: This is the time required for the response to first reach 50% of the final value.

2. Rise time t_r: This is the time required for the response to rise from (say) 10% to 90%, or 5% to 95%, or 0% to 100% of the final value. For under-damped second order systems, the time from 0% to 100% is used. For over-damped systems, the time taken from 10% to 90% is used.

3. Peak time t_p: This is the time taken for the response to reach the first peak of the overshoot.

4. Maximum overshoot y^{over}: This is the maximum value of the response curve measured from unity defined as follows:
$$y^{over} = \max_{k>0}(y_k - 1). \tag{2.2}$$

5. Maximum undershoot y^{under}: This is the maximum negative value of the response curve defined as follows:
$$y^{under} = \max_{k>0}(-y_k). \tag{2.3}$$

6. Settling time t_s: This is the time required for the response curve to reach and stay within a range of certain percentage (usually 5% or 2%) of the final value.

The various attributes are illustrated in Figure 2.1. It is noted here that some of the above specifications may conflict with one another.

Another requirement that is specified alongside the above step response specification are constraints on the actuator. In any practical control system, the size and frequency of the actuator signal or the output signal of the controller is effectively limited by saturation or mechanical limitations of the actuator.

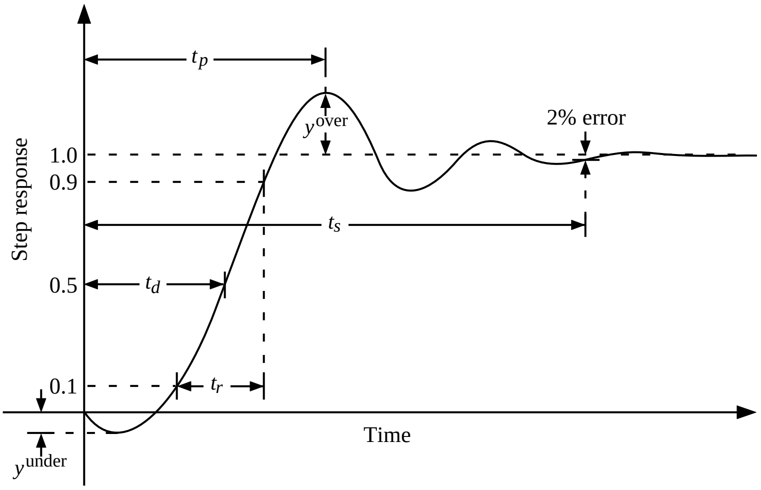


FIGURE 2.1. Transient specifications of the step response

Error Signal Specifications

Let us consider the system of Figure 2.3.1, with (2.3.2) reproduced here for convenience:

$$\begin{aligned} e &= P_{11}w + P_{12}u, \\ y &= P_{21}w + P_{22}u, \\ u &= Ky. \end{aligned} \tag{2.4}$$

Let w_k and e_k denote the value of the vectors w and e at the kth sample time. Here the input vector sequence w includes both disturbances into the system of Figure 2.3.1, and reference signals for the system outputs to track. We will exclude here known deterministic reference signals which can be modeled and included into the plant model. The vector sequence e is the system response to control signals u, and to disturbances and reference signals w. The vector e is constructed by selecting appropriate P_{11} and P_{12} to form components consisting of appropriately scaled functions or filtered values of the states of P_{22} and the input w. The two vectors can be written into their component form as

$$w_k = \begin{bmatrix} w_{1,k} & w_{2,k} & \dots & w_{p,k} \end{bmatrix}', \qquad e_k = \begin{bmatrix} e_{1,k} & e_{2,k} & \dots & e_{m,k} \end{bmatrix}'. \tag{2.5}$$

We do not explore in much more detail here the process of the P_{11}, P_{12} selection. Sometimes this is obvious given the design objectives, but at other times, the selection may be the most important part of the control design and require a number of trials. There is no ‘optimal’ method for the P_{11}, P_{12} selection. This is where it can be an advantage to think in classical frequency domain terms as we now briefly discuss.

We know that (first or second order) minimum phase systems are the easiest to control and allow robust, good performance designs. The frequency shaping in P_{11}, P_{12} is often selected so that e looks like the output of such a system.

Also in classical control design, we are aware that in closing a control loop and increasing the loop gain, the closed-loop poles move to the open-loop zeros where these exist and to infinite asymptotes otherwise. There is difficulty in classical design to achieve a robust high loop gain design when the open-loop zeros are ‘unstable’ or nearly so, since these attract the closed-loop poles as the loop gain increases. The classical *proportional, plus integral plus differential* (PID) controller design technique effectively introduces open-loop zeros in appropriate locations so that in closed loop, as the loop gain increases, the poles approach these assigned locations and a desired system bandwidth determined by these is achieved. Here the selection of P_{12} can be such as to introduce zeros in desired closed-loop pole locations. That is, the insights of PID design can be brought to bear in designing P_{12}. Closing the loop with the appropriate (frequency shaped) gains is then the province of the optimal disturbance rejection method. The optimization is such as to achieve stability, desired (approximate) closed-loop pole locations, and thereby desired bandwidth, as well as disturbance rejection.

With the sequence e appropriately constructed, as just discussed, the next step is to formulate a measure of e so that we can design a controller to minimize e according to this measure. The commonly used measures of e are norms such as the one-, two- and infinity- norm. Which norm to use will depend on the problem to be solved.

The question to be asked is: If the error vector e deviates from its ideal value, what is the consequence to the overall objective of the control problem? To answer this question, we will have to examine the relationship between the controlled variables defined within e and the physics of the process concerned. For example, in chemical processes, the variables to be controlled in an operation should be at some particular levels prescribed by physical, chemical and thermodynamical laws. A deviation of any controlled variable from the stipulated level will upset the chemical reaction which may lead to undesirable effects.

Specification using 2-norms

This specification is popularized by the so-called linear quadratic (LQ) design method. Here we define the performance index J as

$$J = \|e\|_2^2 \,. \tag{2.6}$$

With the definitions expressed in (2.5), J can be written as (see Appendix B for 2-norm definitions)

$$J = \sum_{k=1}^{\infty} \left(e_{1,k}^2 + e_{2,k}^2 + \cdots + e_{m,k}^2 \right). \tag{2.7}$$

In an LQ regulation design, the error components e_i represent the plant states and controls or a linear combination of these. In the LQ tracking case, the errors are the difference between some reference signals and certain linear combinations of the states. Of course, without any penalty on the control energy, high performance can be achieved for the model, but practical designs require some trade between tracking (regulation) performance and control energy to achieve this performance.

If we assume that the weights are built into the components of e, then it is obvious that this formulation is a weighted sum approach which combines the various components of the e vector into a single performance index. The weights need not necessarily be constants, but can be time varying.

There is also the concept of *frequency shaped* weights to give more penalty to certain components of e in some frequency bands in relation to other bands. Frequency shaping filters can be actually built into the dynamics of P_{11}, P_{12} just as can constant weights on the components of e_i. A commonly used frequency shaped filter is the integrator given as

$$F(z^{-1}) = \frac{1}{1 - z^{-1}}. \tag{2.8}$$

Now $F(z^{-1})$ has an infinite magnitude at zero frequency $(z = 1)$. This gives rise to an infinite penalty for an error of zero frequency. This infinite penalty at zero frequency ensures that the optimal controller achieves zero steady-state error. Other frequency shaping filters may penalize control energy at high frequencies to ensure that high frequency lightly damped unmodeled dynamics are not excited. The concept of frequency shaping is important because it allows us to incorporate frequency domain specifications into an otherwise time domain performance index. It should be emphasized that in an engineering design, an engineer may well spend a large portion of the project time on selecting and adjusting frequency shaped filters.

With the performance index J, as given in (2.6), the design task is then to select a stabilizing controller K, or equivalently using the Q parameterization approach, a stable matrix transfer function Q that will minimize the index J, as

$$\min_{Q \in RH_\infty} \|e\|_2^2 = \min_{Q \in RH_\infty} \sum_{k=1}^{\infty} \left(e_{1,k}^2 + e_{2,k}^2 + \cdots + e_{m,k}^2 \right). \tag{2.9}$$

To perform this minimization, we will have to write the errors e_i in term of the matrix transfer function Q and the input disturbances and references w. With F_Q denoting the transfer function matrix from w to e, in terms of Q, the key relevant equation is (2.5.18), repeated here as

$$e = F_Q w; \qquad F_Q = (P_{11} + P_{12} U \tilde{M} P_{21}) + P_{12} M Q \tilde{M} P_{21} \in RH_\infty. \tag{2.10}$$

Recall that F_Q is affine in Q. For the frequently considered special case where w is white noise, the minimization of (2.9) is then equivalent to

$$\min_{Q \in RH_\infty} \left\| F_Q \right\|_2^2. \tag{2.11}$$

Worst Case Design using 2-norms

Let us consider a 2-norm cousin of the index (2.6)

$$J = \max_{w \in \ell_2, \|w\|_2 \leq 1} \|e\|_2^2 , \tag{2.12}$$

where the ℓ_2 space is defined in Appendix B. Here, the performance index is the worst case 2-norm of e over all 2-norm bounded input disturbances w. Thus w has bounded energy. The controller design task is then given as

$$\min_{Q \in RH_\infty} \max_{w \in \ell_2, \|w\|_2 \leq 1} \sum_{k=1}^{\infty} \left(e_{1,k}^2 + e_{2,k}^2 + \cdots + e_{m,k}^2 \right) . \tag{2.13}$$

If the matrix transfer function from w to e is F_Q of (2.10), then it turns out that (see Vidyasagar (1985), and Chapter 3) this minimization can be rewritten in terms of an ∞-norm as

$$\min_{Q \in RH_\infty} \left\| F_Q \right\|_\infty^2 , \tag{2.14}$$

which is the so-called H_∞ minimization problem. Here $\left\| F_Q \right\|_\infty$ is defined as

$$\begin{aligned} \left\| F_Q \right\|_\infty &= \sup_{\|w\|_2 = 1} \left\| F_Q w \right\|_2 \\ &= \sup_{\theta \in (-\pi, \pi]} \sigma_{\max} \left(F_Q(e^{j\theta}) \right) , \end{aligned} \tag{2.15}$$

where $\sigma_{\max}$ denotes the maximal singular value. Hence, the optimization problem (2.13) can be restated in frequency domain terms as:

$$\min_{Q \in RH_\infty} \sup_{\theta \in (-\pi, \pi]} \sigma_{\max} \left(F_Q(e^{j\theta}) \right) .$$

The H_∞ optimal controller attempts to minimize the worst possible adverse impact of a whole class of disturbance as opposed to just working with disturbances and responses in an average sense as in the linear quadratic design case. In fact the resulting controller rejects disturbances uniformly in all frequency bands. Without frequency shaping, this is not the desired outcome in most practical cases. The controller, though robust to the various input disturbances, generally lacks performance. Thus in practice, the success of an H_∞ optimal design depends very much on the frequency shaping filters incorporated into the performance index. The frequency shaping filters seek to emphasize certain frequency bands. Those frequency bands that contain more uncertainties are given more emphasis in relation to those that have less uncertainties or are outside the plant's operating bandwidth. The result of incorporating these frequency shaping filters, in effect, is to reduce the size of the class of disturbances rejected uniformly by the H_∞ optimal controller.

Specification in ∞-norm

We have already noted a connection between 2-norms in the time domain and ∞-norms in the frequency domain. The ∞-norm specification in the time domain is appropriate when we are looking at minimizing the peak of the system output, as it is precisely the infinity norm of the output sequence. In the case when the error is a vector of sequences such as e of (2.5), then its infinity norm is

$$\|e\|_\infty = \max_i \|e_i\|_\infty \tag{2.16}$$

where, with Z^+ denoting $\{0, 1, 2, \dots\}$,

$$\|e_i\|_\infty = \max_{k \in \mathsf{Z}^+} \left\{ \left| e_{i,k} \right| \right\}. \tag{2.17}$$

So that we can specify the nature of the optimization problem, it is necessary to be precise about which class of disturbance signals w we want to consider. If the input signal w is the class of ℓ_2 bounded sequences (finite energy signals) such that $\|w\|_2 \leq 1$, then (see Vidyasagar (1985) and Section 3.3) we have

$$J = \max_{w \in \ell_2, \|w\|_2 \leq 1} \|e\|_\infty = \left\| F_Q \right\|_2, \tag{2.18}$$

where F_Q is the matrix transfer function from w to e. Moreover, it can be shown that the controller that minimizes this performance index is the same as the controller that minimizes (2.9) when the input w is a zero mean white noise signal of unit variance.

Let us next examine input sequences that are not necessarily 2-norm bounded. In particular let us consider the case where the magnitudes of the inputs are bounded. Without loss of generality, due to linearity we write $\|w\|_\infty \leq 1$. We can then write the following performance index:

$$J = \max_{\|w\|_\infty \leq 1} \|e\|_\infty = \max_{\|w\|_\infty \leq 1, i} \|e_i\|_\infty. \tag{2.19}$$

Again let F_Q be the matrix transfer function from w to e and denote by $f_Q = \left\{ f_{Q,k}, k \in \mathsf{Z}^+ \right\}$ its impulse response sequence. Then

$$F_Q(z^{-1}) = \sum_{k=0}^{\infty} f_{Q,k} z^{-k}. \tag{2.20}$$

The induced norm on F_Q is then given as

$$\begin{aligned}
\left\|F_Q\right\|_1 &= \max_{(\|w\|_\infty=1,i)} \sum_{j=1}^{p} \sum_{\ell=0}^{k} \left\| f_{Q,k-\ell}^{ij} w_{j,\ell} \right\|_\infty \\
&= \max_i \sum_{j=1}^{p} \sum_{\ell=0}^{\infty} \left| f_{Q,\ell}^{ij} \right| \\
&= \max_i \sum_{j=1}^{p} \left\| f_Q^{ij} \right\|_1 \\
&= \left\| f_Q \right\|_1 ,
\end{aligned} \tag{2.21}$$

with f_Q^{ij} the ijth element of f_Q ($f_Q^{ij} = \left\{ f_{Q,k}^{ij}, k \in \mathsf{N} \right\}$ is the impulse response sequence). Thus minimizing the performance index of (2.19) turns out to be equivalent to minimizing the 1 norm of f_Q, or the 1-norm of F_Q, the matrix transfer function from w to e.

$$J = \left\| f_Q \right\|_1 = \left\| F_Q \right\|_1 . \tag{2.22}$$

We shall next examine other variations of specifying an ∞ norm type performance index. The above specification uses a weighted maximum method to combine the various components of the vector e. There are other ways to combine the various components of the vector e. One possibility is to use the weighted sum method, reminiscent of the LQ approach, as follows.

$$J = \sum_{i=1}^{m} \lambda_i \left\| e_i \right\|_\infty , \tag{2.23}$$

where $\lambda_i > 0$ are constant weights. However it turns out that such an approach does not achieve a unique controller for each weight selection. Moreover, each optimal controller corresponds to a range of weighting selections, so that the objective of relative penalty of the various components of e is not necessarily realized.

Another possibility is to write the performance index as follows:

$$J = \sum_{i=1}^{m} \lambda_i \left| e_i \right| , \tag{2.24}$$

where again $\lambda_i > 0$ are constant weights and e_i are the components of e. This differs from the previous index in that the focus is on the weighted sum of the magnitude of the various components at each instant. This is in contrast to the weighted sum of the infinity norm of each component of e. Such a performance index is applicable in cases where the instantaneous value of the sum of the various $|e_i|$ is important. An example is the case where the maximum current drawn from a power supply at every instant of time be kept below the absolute maximum.

Main Points of Section

In this section, we have discussed the formulation of performance measures. There are two main approaches to specify performance of a control system. The first, used commonly in classical controller design, is specified in terms of the closed-loop, steady-state and transient response to a step input. The second, used commonly in optimal control design, incorporates error signals into a performance index. The strategy to be developed here is to optimize performance over the class of all the stabilizing controllers for the process.

4.3 An LQG/LTR Design

In this section, we present the design of the very successful linear quadratic Gaussian (LQG) controller with *loop transfer recovery* (LTR). For further details, see Anderson and Moore (1989) or Kwakernaak and Sivan (1972). First, linear quadratic (LQ) controller design is based on the minimization of a quadratic performance index under the assumption that the plant for which it is designed is linear. The resultant control law is a linear feedback of the states of the plant. This controller is optimal with respect to the defined performance index and is appealing because irrespective of the weight selections, the closed loop is robust to certain plant variations. (In classical terms, continuous-time LQ designs guarantee 60° phase margins and $[-6, \infty)$ dB gain margins. In discrete-time no such margins are guaranteed!)

In the event that the states of the plant are not accessible, then the next step is to replace the states by estimated states using an optimal state estimator. This gives rise to the LQG design. The strategy is optimal if the actual plant to be controlled is the nominal model used in the design of the controller. Otherwise it may be a poor strategy. An LQG design may have closed-loop properties indicating poor stability margins at the plant input and/or output, see Doyle (1974).

A final step in the LQG based design strategy is to modify the LQG controller design in such a way as to achieve full or partial loop transfer recovery of the original state feedback design, see Doyle and Stein (1979) and Moore and Tay (1989c). There is usually a scalar design parameter that can be adjusted to achieve a tradeoff between performance for the nominal design and robustness via recovery of the state feedback design loop gain properties.

There are a number of concerns regarding the LQG/LTR design approach. For minimum phase plants, loop recovery is obtained by increasing the loop gains. The resulting high gain systems may not be attractive for implementation due to their high sensitivity to certain external disturbances, plant parameter changes and unmodeled dynamics. For nonminimum phase plants, full loop recovery of LQG designs can only take place when there is an unstable pole/zero cancellation in the control loop matrix transfer function and consequent instability. This suggests that only partial loop recovery be attempted, or that the state estimate feedback control laws should be constrained as discussed by Zhang and Freudenberg (1987).

In this section, we present an approach to loop recovery, termed sensitivity recovery, which is in essence a frequency-shaped loop recovery emphasizing the frequency region of unity gain (the cross-over frequency). Sensitivity recovery is achieved by augmenting the original LQG controller with an additional filter with matrix transfer function Q, and optimizing the Q for sensitivity recovery using the Q-parameterization technique introduced in Chapter 2.

LQG Design

Let us consider the following state space description of a discrete, linear, time-invariant plant as follows.

$$\begin{aligned} x_{k+1} &= Ax_k + B\left(u_k + w_{1,k}\right); \qquad x_0 \quad \text{given,} \\ y_k &= Cx_k + Du_k + w_{2,k}. \end{aligned} \tag{3.1}$$

Let

$$G : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right], \tag{3.2}$$

where $x_k \in \mathsf{R}^n$ is the state vector, $u_k \in \mathsf{R}^p$ the input vector, $y_k \in \mathsf{R}^m$ the output vector and $w_{1,k} \in \mathsf{R}^p,\ w_{2,k} \in \mathsf{R}^m$ are independent white noise disturbances with covariance matrices given as

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} w_{1,k}w_{1,k}' =: Q_e, \qquad \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} w_{2,k}w_{2,k}' =: R_e. \tag{3.3}$$

We assume that $Q_e = Q_e' \geq 0$ and $R_e = R_e' > 0$.

Let us also assume that (A, B) is stabilizable and (A, C) is detectable. Consider also a quadratic performance index given as follows.

$$J = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \left(x_k' Q_c x_k + u_k' R_c u_k\right), \tag{3.4}$$

where Q_c and R_c are weighting matrices $Q_c = Q_c' \geq 0$ and $R_c = R_c' > 0$. In terms of the error signal introduced in (2.5), we have

$$e_k = \begin{bmatrix} Q_c^{1/2} x_k \\ R_c^{1/2} u_k \end{bmatrix}. \tag{3.5}$$

and the performance index of (3.4) is the squared *root mean square* (*rms*) value of this error signal. This performance index is only concerned with the steady state performance of the controlled loop. Indeed, any decaying transient is averaged out of the index (3.4). The optimal controller for the plant of (3.1) minimizing the performance index of (3.4) is given as follows, see for example Anderson and Moore (1989):

$$u_k = F\hat{x}_k, \tag{3.6}$$
$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + H(C\hat{x}_k + Du_k - y_k); \qquad \hat{x}_0 \quad \text{given}, \tag{3.7}$$

where

$$\begin{aligned} S_{c,k} &= A'S_{c,k+1}A - A'S_{c,k+1}B(R_c + B'S_{c,k+1}B)^{-1}B'S_{c,k+1}A + Q_c, \\ \bar{S}_c &= \lim_{k\to-\infty} S_{c,k}, \qquad S_{c,N} = Q_c, \\ F &= -(R_c + B'\bar{S}_cB)^{-1}B'\bar{S}_cA, \end{aligned} \tag{3.8}$$

and

$$\begin{aligned} S_{e,k+1} &= AS_{e,k}A' - AS_{e,k}C'(R_e + CS_{e,k}C')^{-1}CS_{e,k}A' + BQ_eB', \\ \bar{S}_e &= \lim_{k\to\infty} S_{e,k}, \qquad S_{e,0} = 0, \\ H &= -A\bar{S}_eC'(R_e + C\bar{S}_eC')^{-1}. \end{aligned} \tag{3.9}$$

Existence of $\bar{S}_c$, $\bar{S}_e$ are guaranteed by stabilizability of the pair (A, B) and detectability of the pair (A, C), respectively. Closed-loop asymptotic stability follows also from both these condition.

The output feedback LQG controller constructed from the regulator state feedback gain F and the estimator gain (output injection) H is realized as (see also (2.4.17))

$$K : \left[\begin{array}{c|c} A + BF + HC + HDF & -H \\ \hline F & 0 \end{array}\right]. \tag{3.10}$$

Actually in much of the theory to follow, F and H can represent any stabilizing gains for the derived plants, not necessarily obtained from an LQG design.

Let us also define stable coprime factorizations for the plant (3.2) and LQG controller (3.10). In the notation of (2.4.1), (2.4.2) and (2.4.18), (2.4.19), this is given as

$$\begin{bmatrix} M & U \\ N & V \end{bmatrix} : \left[\begin{array}{c|cc} A + BF & B & -H \\ \hline F & I & 0 \\ C + DF & D & I \end{array}\right], \tag{3.11}$$

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} : \left[\begin{array}{c|cc} A + HC & -(B + HD) & H \\ \hline F & I & 0 \\ C & -D & I \end{array}\right]. \tag{3.12}$$

See Section 3.5 for a discussion of the minimality of these realizations.

The class of all stabilizing controllers for the plant (3.1) is then given by (2.5.2), repeated here as

$$\begin{aligned} K(Q) &= U(Q)V(Q)^{-1} = \tilde{V}(Q)^{-1}\tilde{U}(Q), \\ U(Q) &= U + MQ, \qquad V(Q) = V + NQ, \\ \tilde{U}(Q) &= \tilde{U} + Q\tilde{M}, \qquad \tilde{V}(Q) = \tilde{V} + Q\tilde{N}, \end{aligned} \tag{3.13}$$

and depicted in Figure 2.5.4 for the case where the nominal controller is a state estimate feedback controller, as for example, derived via the LQ control problem discussed above.

We observe that a state space realization for *normalized* coprime factorizations for the plant G and controller K are obtained in the form (3.11), (3.12), but with F and H calculated from an LQG design by setting

$$e = \begin{bmatrix} Cx + Du \\ u \end{bmatrix}. \tag{3.14}$$

A *normalized* right coprime factor follows from the following control Riccati equation leading to a state feedback gain F:

$$\begin{aligned} S_{c,k} &= \left(A - BT_c R_c^{-1} C\right)' \left(S_{c,k+1} - S_{c,k+1} B \left(R_c + B' S_{c,k+1} B\right)^{-1} B' S_{c,k+1}\right), \\ &\quad \times \left(A - BT_c R_c^{-1} C\right) + \left(Q_c - T_c' R_c^{-1} T_c\right), \\ \bar{S}_c &= \lim_{k\to-\infty} S_{c,k}, \qquad S_{c,N} = Q_c, \\ Q_c &= C'C, \qquad R_c = D'D + I, \qquad T_c = 2C'D, \\ F &= -\left(R_c + B'\bar{S}_c B\right)^{-1} B'\bar{S}_c A, \end{aligned}$$

while the left coprime factor's state space realization follows from the filter Riccati equation leading to an output injection H:

$$\begin{aligned} S_{c,k+1} &= \left(A - BT_e R_e^{-1} C\right) \left(S_{e,k} - S_{e,k} C' \left(R_e + C S_{e,k} C'\right)^{-1} C S_{e,k}\right) \\ &\quad \times \left(A - BT_e R_e^{-1} C\right)' + B\left(Q_e - T_e R_e^{-1} T_e'\right) B', \\ \bar{S}_e &= \lim_{k\to\infty} S_{e,k}, \qquad S_{e,0} = 0, \qquad T_e = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} w_{1,k} w_{2,k}', \\ H &= -A\bar{S}_e C' \left(R_e + C\bar{S}_e C'\right)^{-1}. \end{aligned}$$

Formulation of Sensitivity Recovery

In this subsection, we consider sensitivity functions of the full state feedback design and the full state estimator feedback design. We formulate an error function through which we achieve recovery of loop robustness of the full state feedback system. As a point of notation, recall the following realizations definitions:

$$G_F : \left[\begin{array}{c|c} A & B \\ \hline I & 0 \end{array}\right], \qquad G_H : \left[\begin{array}{c|c} A & I \\ \hline C & 0 \end{array}\right]. \tag{3.15}$$

Input Sensitivity Recovery

Consider the full state feedback control system design of Figure 3.1. The closed-loop matrix transfer function from w to z is the input sensitivity function matrix given by

$$S^i_{SF} = (I - FG_F)^{-1} = M : \left[\begin{array}{c|c} A + BF & B \\ \hline F & I \end{array}\right]. \tag{3.16}$$

For the state estimate feedback design, the input sensitivity function matrix is given by

$$S^i_{SEF} = (I - KG)^{-1} = M\tilde{V}. \tag{3.17}$$

where K is given by (3.10). Let us now consider the the class of all stabilizing controllers $K(Q)$ parameterized by $Q \in RH_\infty$ as given in (3.11) - (3.13). For a stabilizing controller $K(Q),\ Q \in RH_\infty$, the associated input sensitivity function matrix is

$$S^i_Q = (I - K(Q)G)^{-1} = M(\tilde{V} + Q\tilde{N}). \tag{3.18}$$

(Refer to (2.5.6).)

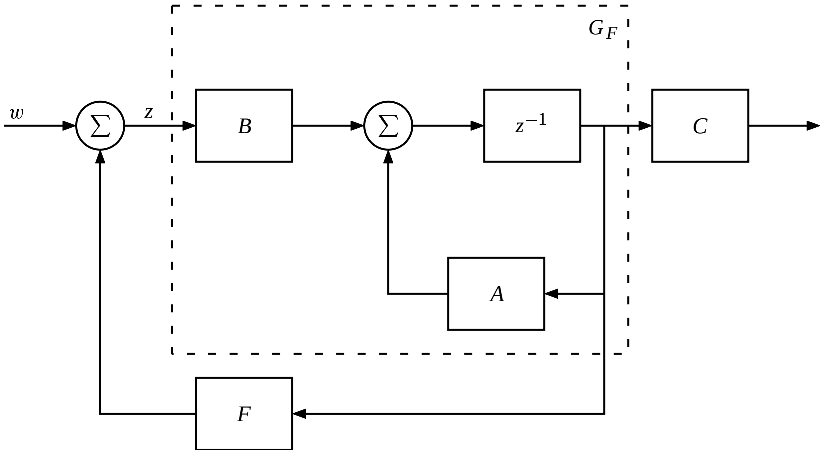


FIGURE 3.1. Target state feedback design

Let us now define an error matrix transfer function

$$\epsilon^i_Q = S^i_{SF} - S^i_Q. \tag{3.19}$$

When plant disturbances or uncertainties occur at the plant inputs, the minimizing of ϵ^i_Q gives robustness to a controller design.

Using (3.16), (3.17) and (3.18) we observe that this error matrix transfer function is affine in Q

$$\epsilon^i_Q = M - M(\tilde{V} + Q\tilde{N}) = M(I - \tilde{V}) - MQ\tilde{N}. \tag{3.20}$$

It can equally be rewritten as

$$\epsilon^i_Q = (I - FG_F)^{-1}(FG_F - K(Q)G)(I - K(Q)G)^{-1}. \tag{3.21}$$

This allows us to interpret ϵ^i_Q as a frequency-shaped version of the loop-gain transfer function error $(FG_F - K(Q)G)$. The frequency shaping is provided by $M = (I - FG_F)^{-1}$, the target sensitivity function and $(I - K(Q)G)^{-1}$, which is the actual closed-loop sensitivity function (parameterized by $Q \in RH_\infty$). These weightings together serve to emphasize the unity loop gain frequencies, that is, the crossover frequencies.

Output Sensitivity Recovery

Consider the full state estimator feedback loop of Figure 3.2. The closed-loop matrix transfer function from $\tilde{w}$ to $\tilde{z}$ is the output sensitivity function matrix given in terms of the output injection H as

$$S^o_{OI} = (I - G_H H)^{-1} = \tilde{M} : \left[\begin{array}{c|c} A + HC & H \\ \hline C & I \end{array}\right]. \tag{3.22}$$

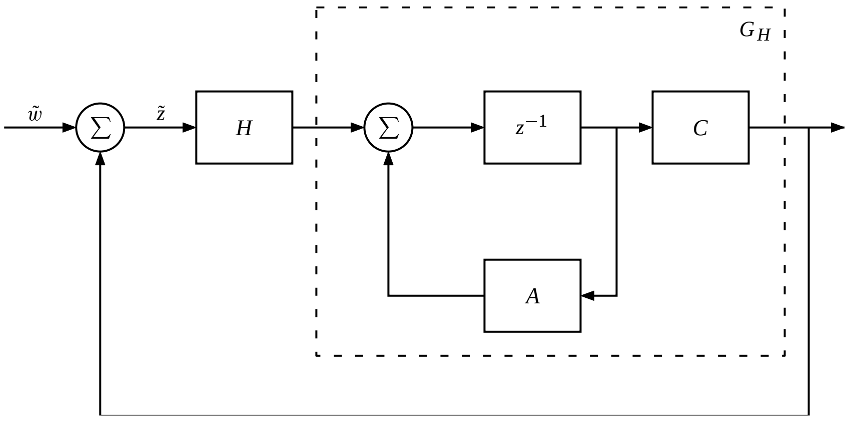


FIGURE 3.2. Target estimator feedback loop design

For the state estimate feedback design, the output sensitivity function matrix is given by

$$S^o_{SEF} = (I - GK)^{-1} = V\tilde{M}, \tag{3.23}$$

where K is given by (3.10). Let us now consider a stabilizing controller $K(Q)$ parameterized by $Q \in RH_\infty$ as given in (2.5.2). For a stabilizing controller $K(Q)$, $Q \in RH_\infty$, the associated output sensitivity function matrix is

$$S^o_Q = (I - GK(Q))^{-1} = (V + NQ)\tilde{M}, \tag{3.24}$$

(where we used the identity (2.5.6)).

Introduce an error matrix transfer function as:

$$\epsilon^o_Q = S^o_{OI} - S^o_Q. \tag{3.25}$$

By duality with the input sensitivity case, when disturbances or uncertainties occur at the plant outputs, the minimization of ϵ^0_Q gives robustness to a controller design.

From (3.23) and (3.24) we observe that ϵ^o_Q is affine in Q as follows:

$$\epsilon^o_Q = (I - V)\tilde{M} - NQ\tilde{M}. \tag{3.26}$$

It may equally be interpreted as a frequency weighted loop gain error by rewriting ϵ^o_Q as follows:

$$\epsilon^o_Q = (I - G_H H)^{-1} (G_H H - GK(Q)) (I - GK(Q))^{-1}. \tag{3.27}$$

Loop Recovery via Sensitivity Recovery

Asymptotic loop recovery of a suitably parameterized LQG design is said to occur when, with design parameter adjustments, the loop matrix transfer function of the LQG design, namely KG (or GK), approaches the loop transfer matrix transfer function of the target LQ (or estimator) design, namely FG_F (or $G_H H$), for all $z = e^{j\omega T}, 0 \le \omega T < \pi$. Now since F and H are stabilizing controllers for G_F and G_H, respectively, and K is a stabilizing controller for G, then $S^i_{OI} = (I - FG_F)^{-1}$, $S^o_{OI}(I - G_H H)^{-1}$, $S^i_{SEF} = (I - KG)^{-1}$ and $S^o_{SEF}(I - GK)^{-1}$ are well defined for all $z = e^{j\omega T}, 0 \le \omega T < \pi$. In view of these properties, we see that loop recovery occurs, that is, $KG \to FG_F$ $(GK \to G_H H)$, if and only if $(I - KG)^{-1} \to (I - FG_F)$ $((I - GK)^{-1} \to (I - G_H H)^{-1})$, or equivalently, when the loop sensitivity function matrix of the LQG design, namely S^i_{SEF} (or S^o_{SEF}), when suitably parameterized, approaches the loop sensitivity function matrix of the LQ (or estimator) design S^i_{SF} (or S^o_{SF}) for all $z = e^{j\omega T}, 0 \le \omega T < \pi$. Of course, these equivalent loop recovery definitions also apply to the LQG design augmented with arbitrary $Q \in RH_\infty$, with S^i_{SEF} and S^o_{SEF} replaced by S^i_Q and S^o_Q. More specifically, since $K(Q)$ stabilizes G so that $(I - K(Q)G)^{-1}$

and $(I - GK(Q))^{-1}$ exists for all $z = e^{j\omega T}, 0 \leq \omega T < \pi$, then from (3.21) and (3.27) we see that loop recovery occurs, equivalently, when ϵ_Q^i tends to zero (or ϵ_Q^o tends to zero), that is we have asymptotic sensitivity recovery. We conclude the following equivalent asymptotic loop and sensitivity recovery conditions.

Lemma 3.1. *Asymptotic loop recovery at the plant input (or output) occurs if and only if, for suitable parameterizations, there is asymptotic sensitivity recovery given by*

$$\epsilon_Q^i \to 0 \quad (\text{or} \quad \epsilon_Q^o \to 0) \quad \text{for } z = e^{j\omega T}, \qquad 0 \leq \omega T < \pi. \tag{3.28}$$

A partial sensitivity recovery is said to occur when ϵ_Q^i (or ϵ_Q^o) are made small in some sense. This also corresponds to a partial loop recovery, albeit frequency shaped by virtue of (3.21) and (3.27). Reasonable measures are the 2-norm or ∞-norm with tasks defined as

$$\begin{aligned} &\min_{Q\in RH_\infty} \left\|\epsilon_Q^i\right\|_2, \qquad \min_{Q\in RH_\infty} \left\|\epsilon_Q^i\right\|_\infty, \\ \Big(\text{or} \quad &\min_{Q\in RH_\infty} \left\|\epsilon_Q^o\right\|_2, \qquad \min_{Q\in RH_\infty} \left\|\epsilon_Q^o\right\|_\infty \Big). \end{aligned} \tag{3.29}$$

These are standard H_2, H_∞ optimization tasks (see also the next section) since $\epsilon_Q^i, \epsilon_Q^o \in RH_\infty$ are affine in $Q \in RH_\infty$. When

$$\begin{aligned} &\left\|\epsilon_Q^i\right\|_2 = 0, \qquad \left\|\epsilon_Q^i\right\|_\infty = 0, \\ \Big(\text{or} \quad &\left\|\epsilon_Q^o\right\|_2 = 0, \qquad \left\|\epsilon_Q^o\right\|_\infty = 0 \Big), \end{aligned} \tag{3.30}$$

there is full sensitivity recovery, and by virtue of (3.21) and (3.27) there is full loop recovery. We have the following lemma.

Lemma 3.2. *The state estimate and residue feedback controllers $K(Q)$, with F and H fixed and $Q \in RH_\infty$ variable, achieve full loop recovery, equivalently input (or output) sensitivity recovery, if and only if Q is selected as $Q_i \in RH_\infty$ (or $Q_o \in RH_\infty$) with*

$$(I - \tilde{V}) = Q_i \tilde{N}, \qquad (I - V) = N Q_o. \tag{3.31}$$

We will next examine the existence of such Q_i and Q_o.

Full Loop Recovery Cases

In this subsection we will examine cases where full loop recovery can be achieved.

Minimum Phase Plants

We present full loop recovery results for the case of plants whose finite zeros are stable, that is plants with full rank properties as follows:

$$\operatorname{rank} \begin{bmatrix} zI - A & B \\ C & D \end{bmatrix} \quad \text{is constant for } |z| \geq 1, \tag{3.32}$$

or equivalently, G is minimum phase. Moreover, we require the plants to satisfy the following invertibility condition:

$$G^{-L} \quad \text{(left inverse)} \quad \text{or} \quad (zG)^{-L} \in R_p \quad \text{exists}, \tag{3.33}$$

or

$$G^{-R} \quad \text{(right inverse)} \quad \text{or} \quad (zG)^{-R} \in R_p \quad \text{exists}. \tag{3.34}$$

We have the following result:

Theorem 3.3. *Consider a plant* (3.2) *and a state estimate feedback controller* (3.10) *constructed from a state feedback gain F and a state estimate gain H and with $Q \in RH_\infty$. Consider also associated factorizations* (3.11), (3.12). *Let the plant G be minimum phase, in that condition* (3.32) *holds. Full input sensitivity (output sensitivity) recovery is possible provided the plant has a left (right) inverse, that is condition* (3.33) *(or condition* (3.34)*) is satisfied. The recovery $\epsilon_Q^i = 0$, see* (3.19), *(or $\epsilon_Q^o = 0$, see* (3.25)*), is achieved when Q is selected as Q_i (Q_o) given by:*

$$Q_i = (I - \tilde{V})\tilde{N}^{-L} \in RH_\infty \quad (\text{or} \quad Q_o = N^{-R}(I - V) \in RH_\infty \quad), \tag{3.35}$$

or equivalently,

$$Q_i = z(I - \tilde{V})(z\tilde{N})^{-L} \in RH_\infty \quad (\text{or} \quad Q_o = (zN)^{-R} z(I - V) \in RH_\infty \quad). \tag{3.36}$$

Proof. Condition (3.32), together with $G = \tilde{M}^{-1}\tilde{N}$ implies that $\tilde{N}$ is minimum phase. Moreover, if $G^{-L} \in R_p$ exists, we have that $\tilde{N}^{-L}$ exists, and is stable. It follows that condition (3.31) can be realized by selecting $Q = (I - \tilde{V})\tilde{N}^{-L}$. This selection satisfies $Q \in RH_\infty$ by virtue of $\tilde{N}$ being minimum phase. The other conditions can be explored in a similar manner. □

We remark that if the plant has the same number of inputs as outputs, the modes of Q_i (or Q_o) are identical to the set or subset of the zeros of the plant (values of z for which the plant G loses rank in (3.32)), and the McMillan degree is upper bounded by n.

Nonminimum Phase Plants:

As discussed earlier for nonminimum phase plants, it is in general not feasible to achieve full loop (sensitivity) recovery. However, for certain partial state feedback designs, full or partial (sensitivity) recovery can in fact be achieved.

Qualitatively, this is done as follows. First write the plant G as a product of two factors where one is *all-pass* (possibly unstable) and the other a square *minimum phase* stable factor. Recall that all-pass systems have a flat spectrum: Poles z_p and zeros z_z occur in pairs satisfying $z_p = z_z^{-1}$. Let us assume then that for the factored plant there exists a state estimator gain and dynamic state feedback gain such that only the states associated with the minimum phase factor are fed back.

For the partial state feedback, we can write down the the input sensitivity function, and the output sensitivity function, and the corresponding sensitivity difference functions as in (3.20) and (3.26).

Once the sensitivity difference functions are defined, being affine in Q_i (or Q_o), full or partial loop sensitivity recovery is achieved by appropriate selection of Q_i and Q_o, see Moore and Tay (1989c).

Example. Consider the continuous-time, unstable, minimum phase plant with transfer functions

$$G_c = \frac{s+1}{s^2 - 3s + 3}.$$

Note that this plant has two unstable poles. Discretizing G_c with a time sampling period of $t_s = 0.7$, (see Chapter 9) we have for the equivalent Z-domain representation:

$$G : \left[\begin{array}{cc|c} 4.6969 & -8.1662 & 1 \\ 1 & 0 & 0 \\ \hline 2.3706 & -0.8809 & 0 \end{array}\right],$$

with a minimum phase zero at 0.3716 and unstable poles at $2.3484 \pm j1.6282$.

We design LQ and LQG controllers assuming $Q_e = I$, $R_e = 1$ and $e_k' = [\, y_k \;\; u_k \,]$. The state feedback gain and state estimator gains are given by

$$F = \begin{bmatrix} -4.1375 & 8.0536 \end{bmatrix}, \qquad H' = \begin{bmatrix} -2.0949 & -0.4136 \end{bmatrix}.$$

The Z-domain Nyquist plots for the LQ and LQG controllers are shown in Figure 3.3. The open loop in the LQ case has two unstable (plant) poles and so encircles the Nyquist point $(-1, 0)$ twice, whereas, for the LQG case, the open loop has four unstable poles (two plant and two controller) an so encircles the Nyquist point four times. Obviously, the robustness of the LQG controller is inferior compared to the LQ controller, at least in terms of gain margins and phase margins. Using the loop transfer recovery described in this section, we obtain Q_i as

$$Q_i = \begin{bmatrix} 0.3716 & 1 \\ 0.0536 & -1.7453 \end{bmatrix} \in RH_\infty.$$

This gives rise to $K(Q_i)$ which achieves full loop transfer recovery. For partial loop recovery, we use $K(\alpha Q_i)$, $0 \le \alpha \le 1$. Figure 3.4 shows the extent of recovery for $\alpha = 0.5$, $\alpha = 0.95$. Clearly when $\alpha = 0.5$ there is a significant recovery and when $\alpha = 0.95$ there is near recovery. (Full recovery occurs when $\alpha = 1$.)

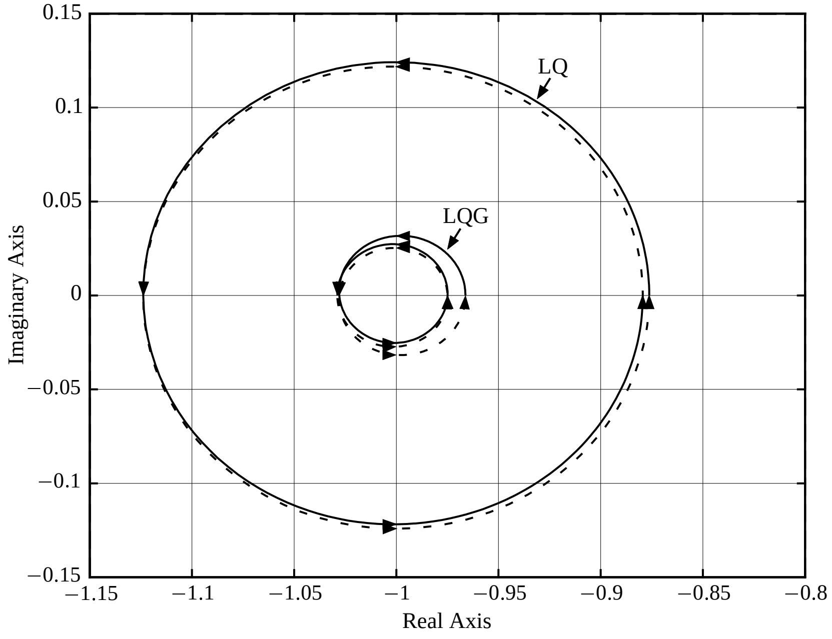


FIGURE 3.3. Nyquist plots—LQ, LQG

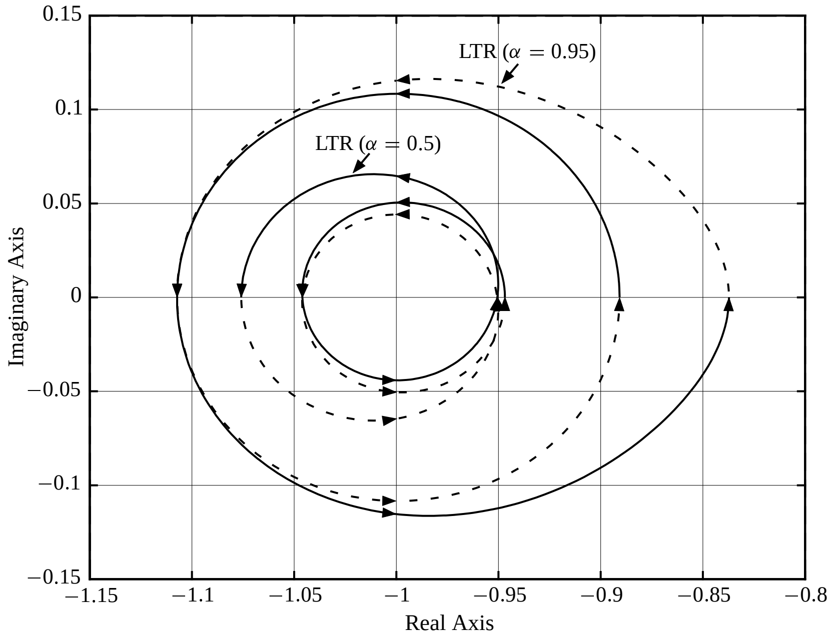


FIGURE 3.4. Nyquist plots—LQG/LTR: $\alpha = 0.5, 0.95$

Main Points of Section

In this section the design of an LQG controller with loop transfer recovery or more precisely sensitivity recovery has been discussed. Sensitivity recovery is achieved by augmenting the original LQG controller with an additional matrix transfer function Q, feeding back the estimation residuals. We show that for minimum phase plants and some nonminimum phase plants where a particular partial state estimate feedback controller is used, full loop recovery may be achieved. Otherwise only partial recovery is achieved and this can be done in an optimal manner through the sensitivity recovery approach using the Q parameterization.

4.4 H_∞ Optimal Design

In this section, we present the formulas for solving the H_∞ optimization task of (2.14). Most of the formulas are quoted from Green and Limebeer (1994).

Problem Formulation

Let us consider the plant model of (2.2.1) with realizations written as follows.

$$\begin{bmatrix} e \\ y \end{bmatrix} = P \begin{bmatrix} w \\ u \end{bmatrix}, \qquad P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} : \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0 \end{array}\right]. \tag{4.1}$$

In this realization[*], (A, B_2) is stabilizable and (A, C_2) is detectable. We assume that w is any bounded sequence with $\|w\|_2 \leq 1$ and P_{11}, P_{21} contain stable frequency shaping filters that determine the influence of w in the various frequency bands on e and y. The selection of the frequency shaping filters will depend on *a priori* knowledge of the disturbance to the plant in the various frequency bands. Similarly we assume that P_{11}, P_{12} contain stable frequency shaping filters that penalize the elements of e appropriately in the various frequency bands.

Let us consider a stabilizing controller K for (4.1) ($u = Ky$) and corresponding stable coprime factorizations of (3.13) for K and (4.1). Realizations in terms of the parameters in (4.1) are given in (3.10), (3.11) and (3.12) with $B = B_2$, $C = C_2$ and $D = D_{22}$. We can now write down the class of all stabilizing controllers for (4.1) in term of a stable matrix transfer function $Q \in RH_\infty$. The closed-loop

[*]Without (great) loss of generality, one can assume that there is no direct feedthrough term from input u to the control output y. It is always possible to replace an output with feedthrough term by an equivalent output without it, by simple subtraction. This simplifies some of the algebraic expressions.

matrix transfer function is then given by (2.5.18), repeated here as

$$e = F_Q w, \tag{4.2}$$

$$\begin{aligned} F_Q &= (P_{11} + P_{12} U \tilde{M} P_{21}) + P_{12} M Q \tilde{M} P_{21} \in RH_\infty \\ &= T_{11} + T_{12} Q T_{21}. \end{aligned} \tag{4.3}$$

The realization for T is repeated from (2.5.22) as follows.

$$T : \left[\begin{array}{cc|cc} A + B_2 F & -HC_2 & -HD_{21} & B_2 \\ 0 & A + HC_2 & B_1 + HD_{21} & 0 \\ \hline C_1 + D_{12} F & C_1 & D_{11} & D_{12} \\ 0 & C_2 & D_{21} & 0 \end{array} \right]. \tag{4.4}$$

Now consider the performance index

$$J = \max_{\|w\|_2 \le 2} \|e\|_2 = \left\| F_Q \right\|_\infty = \|T_{11} + T_{12} Q T_{21}\|_\infty . \tag{4.5}$$

The optimization task is

$$\min_{Q \in RH_\infty} \|T_{11} + T_{12} Q T_{21}\|_\infty . \tag{4.6}$$

In order to be able to solve this so-called H_∞ control problem, the following sufficiency assumptions are made:

1. (A, B_2) is stabilizable and (A, C_2) is detectable,
2. $D_{12}' D_{12} > 0$ and $D_{21}' D_{21} > 0$,
3. rank $\begin{bmatrix} A - e^{j\theta} I & B_2 \\ C_1 & D_{12} \end{bmatrix} = n + m$, for all $\theta \in [0, 2\pi]$,
4. rank $\begin{bmatrix} A - e^{j\theta} I & B_2 \\ C_1 & D_{21} \end{bmatrix} = n + m$, for all $\theta \in [0, 2\pi]$.

Assumption 1 is obviously necessary from a control point of view; without Assumption 1 no stabilizing output feedback controller can be constructed. Assumption 2 provides sufficient conditions under which the control strategy can be implemented, often ≥ 0 will suffice. Assumptions 3 and 4 are conditions that amount to T_{12} and T_{21} having no zeros on the unit circle. These conditions are crucial as we need to invert T_{12} and T_{21} in some sense to find the optimal controls.

Before presenting a solution to the optimal H_∞ control problem (4.6), we present a state space formulated solution for the characterization of all controllers that achieve the less stringent performance objective

$$\left\| F_Q \right\|_\infty < \gamma, \tag{4.7}$$

or equivalently,

$$\|e\|_2 < \gamma \, \|w\|_2 \, . \tag{4.8}$$

The solution to this problem requires one to solve a set of coupled algebraic Riccati equations. If for a certain γ we fail to find a solution then this indicates that it is impossible to decrease the gain between disturbance and performance signals any further. This observation can be used to construct a crude method for iteratively finding a controller that approaches the H_∞ optimal controller that minimizes the criterion (4.6).

Consider the algebraic Riccati equation:

$$X = C'JC + A'XA - \square'\square^{-1}\square, \tag{4.9}$$

where

$$C = \begin{bmatrix} C_1 \\ 0 \end{bmatrix}, \qquad J = \begin{bmatrix} I_1 & 0 \\ 0 & -\gamma^2 I_2 \end{bmatrix},$$

$$D = \begin{bmatrix} D_{11} & D_{12} \\ I_2 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} B_1 & B_2 \end{bmatrix},$$

are defined from the plant realization matrices (see (4.1)) and

$$\square = \begin{bmatrix} \square_1 \\ \square_2 \end{bmatrix} = D'JC + B'XA,$$

$$\square = \begin{bmatrix} \square_1 & \square_2 \\ \square_2 & \square_3 \end{bmatrix} = D'JD + B'XB.$$

Here I_1 is an identity matrix with dimension of the performance variable e, I_2 is an identity matrix whose dimension corresponds with that of the disturbance variable w. The matrix $\square_1$ has row dimension equal to that of w, and $\square_2$ has row dimension equal to that of the input u. Similarly for $\square$.

Assume that the control Riccati equation (4.9) has a solution such that

$$\begin{aligned} &X \geq 0 \quad \text{and} \quad A - B\square^{-1}\square \text{ is stable,} \\ &\square_3 > 0, \qquad \square_1 - \square_2'\square_3^{-1}\square_2 < 0. \end{aligned} \tag{4.10}$$

Introduce the square root factors for $\square_3$ and $-\square_1 + \square_2'\square_3^{-1}\square_2$

$$R'R = \square_3, \qquad -\gamma^2 T'T = \square_1 - \square_2'\square_3^{-1}\square_2, \tag{4.11}$$

and define

$$W = \begin{bmatrix} R\square_3^{-1}\square_2' & R \\ T & 0 \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & 0 \end{bmatrix}, \tag{4.12}$$

$$L = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} = J^{-1}\left(W'\right)^{-1}\square. \tag{4.13}$$

Also introduce the “bar” variables as

$$\begin{bmatrix} \bar{A} & \bar{B}_1 & \bar{B}_2 \\ \bar{C}_1 & \bar{D}_{11} & \bar{D}_{12} \\ \bar{C}_2 & \bar{D}_{21} & 0 \end{bmatrix} = \begin{bmatrix} A - B_1 W_{21}^{-1} L_2 & B_1 W_{21}^{-1} & B_2 \\ L_1 - W_{11} W_{21}^{-1} L_2 & W_{11} W_{21}^{-1} & W_{12} \\ C_2 - D_{21} W_{21}^{-1} L_2 & D_{21} W_{21}^{-1} & 0 \end{bmatrix}. \tag{4.14}$$

Finally introduce

$$\bar{D} = \begin{bmatrix} \bar{D}_{11} & I \\ \bar{D}_{21} & 0 \end{bmatrix}, \qquad \bar{C} = \begin{bmatrix} \bar{C}_1 \\ \bar{C}_2 \end{bmatrix}, \qquad \bar{B} = \begin{bmatrix} \bar{B}_1 & 0 \end{bmatrix}.$$

Consider now the Riccati equation

$$Z = \bar{B} J \bar{B}' + AZA' - \bar{\square}\bar{\square}^{-1}\bar{\square}', \tag{4.15}$$

where

$$\bar{\square} = \bar{A} Z \bar{C}' + \bar{B} J \bar{D}', \qquad \bar{\square} = \begin{bmatrix} \bar{\square}_1 & \bar{\square}_2 \\ \bar{\square}_2' & \bar{\square}_3 \end{bmatrix} = \bar{D} J \bar{D}' + \bar{C} Z \bar{C}'. \tag{4.16}$$

Assume furthermore that the filter Riccati equation (4.15) has a solution such that

$$\begin{aligned} Z \geq 0 \quad &\text{and} \quad \bar{A} - \bar{\square}\bar{\square}^{-1}\bar{C} \text{ is stable}, \\ \bar{S}_3 > 0, \qquad &\bar{\square}_1 - \bar{\square}_2 \bar{\square}_3' \bar{\square}_2^{-1} < 0. \end{aligned} \tag{4.17}$$

Under the conditions in (4.10) and (4.17) there exists an output feedback controller that achieves the performance measure (4.7) or (4.8). All such controllers are generated from

$$\begin{bmatrix} \hat{x}_{k+1} \\ u_k \\ r_k \end{bmatrix} = \begin{bmatrix} A_C & B_{C1} & B_{C2} \\ C_{C1} & D_{C11} & D_{C12} \\ C_{C2} & D_{C21} & 0 \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ y_k \\ s_k \end{bmatrix}, \tag{4.18}$$

where (r_k, s_k) are any signals that satisfy a relationship

$$s = Qr,$$

where $Q(z)$ is a stable rational transfer function such that $\|Q\|_\infty < \gamma$. The matrices in (4.18) are defined via the solution of the coupled Riccati equations (4.9) and (4.15) as follows:

$$\begin{aligned} A_C &= \bar{A} - \bar{B}_2 W_{12}^{-1} \bar{C}_1 + \left[\bar{B}_2 W_{12}^{-1} \bar{W}_{11} - \bar{L}\right] \bar{W}_{21}^{-1} \bar{C}_2, \\ \begin{bmatrix} B_{C1} & B_{C2} \end{bmatrix} &= \left[\left(\bar{B}_2 W_{12}^{-1} \bar{W}_{11} - \bar{L}_1\right) \bar{W}_{21}^{-1} \quad \left(\bar{L}_2 - \bar{B}_2 W_{12}^{-1} \bar{W}_{12}\right)\right], \\ \begin{bmatrix} C_{C1} \\ C_{C2} \end{bmatrix} &= \begin{bmatrix} W_{12}^{-1}\left(\bar{C}_1 - \bar{W}_{11} \bar{W}_{21}^{-1} \bar{C}_2\right) \\ \bar{W}_{21}^{-1} \bar{C}_2 \end{bmatrix}, \\ \begin{bmatrix} D_{C11} & D_{C12} \\ D_{C21} & 0 \end{bmatrix} &= \begin{bmatrix} -W_{12}^{-1} \bar{W}_{11} \bar{W}_{21}^{-1} & W_{12}^{-1} \bar{W}_{12} \\ \bar{W}_{21}^{-1} & 0 \end{bmatrix}, \end{aligned}$$

where

$$\bar{W} = \begin{bmatrix} \bar{W}_{11} & \bar{W}_{12} \\ \bar{W}_{21} & 0 \end{bmatrix},$$

is such that $\bar{W} J \bar{W}' = \bar{\Box}$ and

$$\bar{L} = \begin{bmatrix} \bar{L} & \bar{L}_2 \end{bmatrix} = \left(\bar{B} J \bar{D}' + \bar{A} Z \bar{C}'\right)\left(J \bar{W}'\right)^{-1}.$$

The class of controllers (4.18) can be interpreted in terms of an observer/linear feedback structure just as in the LQG design problem. The main difference with the LQG design is that here no separation principle applies. The observer and controller designs are linked. In the above expressions this is seen from the link between the controller Riccati equation (4.9) and the filter Riccati equation (4.15) via the equations (4.11) – (4.14).

4.5 An ℓ_1 Design Approach

Often in applications the control objective is to keep the tracking error within a certain tolerance. An example of this is the regulation of the read/write head of a hard disk drive onto a particular track. Here the control objective is to keep the magnitude of the tracking error to within the width of the track. This type of control objective leads naturally to an ℓ_∞ type performance index

$$J = \|e\|_\infty ,$$

being the infinity norm of the error vector. When the input disturbances w are known to be infinity-norm bounded, then J can be reformulated as an ℓ_1 index

$$J = \left\|F_Q\right\|_1 ,$$

where F_Q is the closed-loop transfer function between the input w to the output e.

Often in the design of controllers, a compromise has to be taken to balance objectives for the plant output and the controller output. As discussed in Chapter 3 and popularized in LQ type controller design, a weighted index is appropriate. In the context of ℓ_1 design, this becomes either a weighted sum or a weighted maximum index given, respectively, as follows

$$J_1 = \|(|y| + \lambda\,|u|)\|_\infty ,$$
$$J_2 = \|y\|_\infty + \lambda\,\|u\|_\infty .$$

However, this double penalty approach in an ℓ_1 design context does not actually achieve the type of effect one would expect it to do, based on experience in the

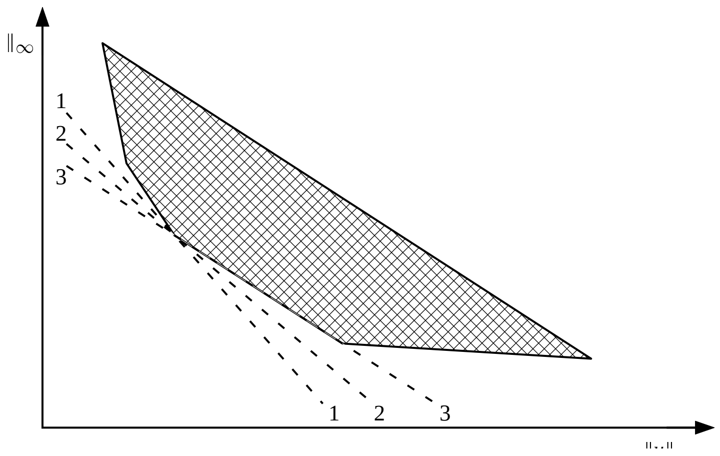


FIGURE 5.1. Limits of performance curve for an infinity norm index for a general system

LQ design context. To see this, let us consider the solution space of the ℓ_1 optimization problem. The set of all feasible solutions for the control effort amplitude and the output amplitude for some particular system forms a convex polygon as illustrated in Figure 5.1.

The boundary of the polygon, termed the *limit-of-performance curve*, is a plot of the best achievable performance for the particular control configuration and is constructed using all possible positive weights λ in the weighted index function. It turns out that this curve consists of only a finite number of linear equations and its gradient is monotonically nondecreasing.

Using a weighted function index results in the solutions of the optimization problem remaining unchanged for some range of weights, that is for some values of λ there are an infinite number of solutions of the optimization problem.

Consider, for example, if the weight is chosen to be any value between the gradients of Line 1 and 2 in Figure 5.1, the solution of the ℓ_1 optimization problem remains unchanged since the optimal solutions are at the same vertex. In other cases such as when the weight is chosen to be the gradient of Line 3, then the resultant ℓ_1 optimization problem has an infinite number of solutions along the edge of the line.

In any weighted index approach, the weights are usually chosen, or at least fine tuned, by trial and error. Without knowledge of the shape of the solution set, a trial and error approach will almost certainly lead to a selection that will give a unique solution at a vertex of the solution set. In the event that the weight chosen leads to an infinite number of optimal solutions, there is no mechanism to select any one of these infinite controllers to practically fulfill the objective of compromising between the magnitude of the output signal and magnitude of the controller effort.

Mathematical Preliminaries

Fact 5.1. *Given a linear function $f : \mathsf{R}^n \to \mathsf{R}$*

1. *If f has the same value at two distinct points, $Y \in \mathsf{R}^n$ and $Z \in \mathsf{R}^n$, then f remains constant along the line YZ.*

2. *If f has different values at Y and Z, then at each point on the open line segment YZ, f has a value strictly between its values at Y and Z.*

Fact 5.2. *The maximum and minimum values of a linear function $f : \mathsf{R}^n \to \mathsf{R}$, restricted to a bounded convex polytope $A \in \mathsf{R}^n$, exist and are to be found on the boundary of A.*

Fact 5.3. *The intersection of any number of convex regions in R^n is convex.*

Problem Formulation

Let us consider a single-input, single-output, discrete-time, linear, time-invariant, proper system expressed as follows

$$\square(q^{-1})y_k = \square(q^{-1})u_k + \square(q^{-1})w_k, \tag{5.1}$$

where y_k, u_k and w_k are the system output, the system input, and the input disturbance to the system at the kth sample, respectively. The input disturbance w_k is assumed to belong to ℓ_∞ with a maximum bound normalized to unity. Here $\square(q^{-1})$, $\square(q^{-1})$, and $\square(q^{-1})$ are polynominals in q^{-1} given as follows

$$\square(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_{n_p} q^{-n_p}, \tag{5.2}$$

$$\square(q^{-1}) = b_0 + b_1 q^{-1} + \cdots + b_{n_p} q^{-n_p}, \tag{5.3}$$

$$\square(q^{-1}) = c_0 + c_1 q^{-1} + \cdots + c_{n_c} q^{-n_c}, \tag{5.4}$$

with $\square(q^{-1})$ and $\square(q^{-1})$ assumed to be coprime. Let us consider a stabilizing control law for system (5.1) as

$$\square(q^{-1})u_k = -\square(q^{-1})y_k, \tag{5.5}$$

where

$$\square(q^{-1}) = 1 + r_1 q^{-1} + \cdots + r_{n_r} q^{-n_r}, \tag{5.6}$$

$$\square(q^{-1}) = s_0 + s_1 q^{-1} + \cdots + s_{n_r} q^{-n_r}, \tag{5.7}$$

with $n_r \geq n_p - 1$. Note that when the plant has direct feed through, s_0 is constrained to be zero to avoid an algebraic loop. The system with its controller is

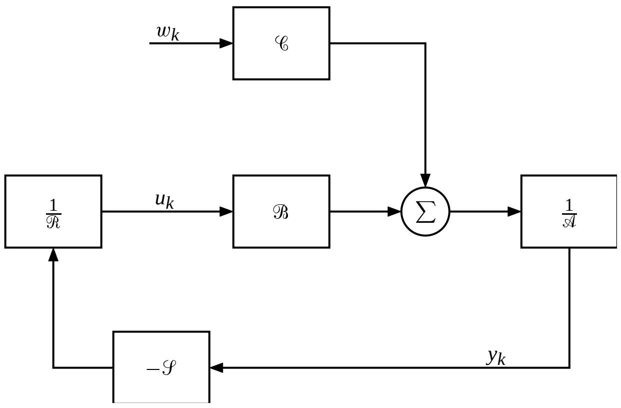


FIGURE 5.2. Plant with controller configuration

shown in Figure 5.2. The closed-loop transfer operators from w_k to y_k and w_k to u_k are then given as

$$y_k = \frac{\square(q^{-1})\square(q^{-1})}{\square(q^{-1})\square(q^{-1}) + \square(q^{-1})\square(q^{-1})} w_k =: G_y(q^{-1}) w_k, \tag{5.8}$$

$$u_k = \frac{-\square(q^{-1})\square(q^{-1})}{\square(q^{-1})\square(q^{-1}) + \square(q^{-1})\square(q^{-1})} w_k =: G_u(q^{-1}) w_k. \tag{5.9}$$

Let

$$U_0 = \{u_i\}; \qquad u_i \in \mathsf{R}, \qquad |u_i| \le 1, \qquad i \in \mathsf{N}, \qquad \lim_{i\to\infty} u_i = 0. \tag{5.10}$$

We can then write

$$\left\| G_y(q^{-1}) \right\|_1 := \sup_{\substack{w\in U_0 \\ \|w_k\|_\infty = 1}} \left| G_y(q^{-1}) w_k \right| = \sup_{\substack{w\in U_0 \\ \|w_k\|_\infty = 1}} |y_k| =: \|y_k\|_\infty , \tag{5.11}$$

$$\left\| G_u(q^{-1}) \right\|_1 := \sup_{\substack{w\in U_0 \\ \|w_k\|_\infty = 1}} \left| G_u(q^{-1}) w_k \right| = \sup_{\substack{w\in U_0 \\ \|w_k\|_\infty = 1}} |u_k| =: \|u_k\|_\infty , \tag{5.12}$$

and the following minimization tasks can be defined:

$$\min_{\square,\square} \|y_k\|_\infty \equiv \min_{R,S} \left\| \square(q^{-1})\square(q^{-1}) \right\|_1 , \tag{5.13}$$

$$\min_{\square,\square} \|u_k\|_\infty \equiv \min_{R,S} \left\| \square(q^{-1})\square(q^{-1}) \right\|_1 , \tag{5.14}$$

subject to the constraint

$$\square(q^{-1})\square(q^{-1}) + \square(q^{-1})\square(q^{-1}) = 1. \tag{5.15}$$

A composite minimization task can be defined from the above two tasks.

We can, without loss of generality, assign the closed-loop poles to the origin with the consequence that the optimal numerator is an infinite impulse response. This in turn can be interpreted as a series expansion of the closed-loop operator about the origin.

Alternatively, we can also assign the closed-loop poles to be the stable zeros of the polynomial $\square(q^{-1})$ so that the denominator of the closed loop remains as unity.

Equation (5.15) can be written into a matrix equation as

$$M\theta = Y, \tag{5.16}$$

where

$$\begin{aligned} \theta &= \left(r_1, \ldots, r_{n_r}, s_0, \ldots, s_{n_r}\right)' \in \mathsf{R}^{2n_r+1}, \\ Y &= \left(-a_1, \ldots, -a_{n_p}, 0, \ldots, 0\right)' \in \mathsf{R}^{n_r+n_p}, \end{aligned}$$

and the matrix M is given as

$$M = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & b_1 & b_0 & 0 & \ldots & 0 \\ a_1 & 1 & 0 & \ldots & 0 & b_2 & b_1 & b_0 & \ddots & \vdots \\ a_2 & \ddots & \ddots & \ddots & \vdots & b_3 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & a_1 & 1 & 0 & \vdots & \ddots & b_2 & b_1 & b_0 \\ a_{n_p} & \ldots & a_2 & a_1 & 1 & b_{n_p} & \ldots & b_3 & b_2 & b_1 \end{pmatrix}$$
$$\in \mathsf{R}^{(n_r+n_p)\times(2n_r+1)}. \tag{5.17}$$

Note that for $n_r = n_p - 1$ the polynomials $\square(q^{-1})$ and $\square(q^{-1})$ are unique, Goodwin and Sin (1984). If $n_r > n_p - 1$, however, the polynomials $\square(q^{-1})$ and $\square(q^{-1})$ are no longer unique. In this case write (5.15) in partitioned form as

$$\begin{bmatrix} \bar{M} & \underline{M} \end{bmatrix} \begin{bmatrix} \bar{\theta} \\ \underline{\theta} \end{bmatrix} = Y, \tag{5.18}$$

where $\bar{M}$ is an invertible square matrix of dimension $(n_r + n_p)$, $\underline{M} \in \mathsf{R}^{(n_r+n_p)\times(n_r+1-n_p)}$, $\bar{\theta} \in \mathsf{R}^{n_r+n_p}$ and $\underline{\theta} \in \mathsf{R}^{n_r+1-n_p}$. Note that this is always possible under the controllability assumption on (5.1). We can then write $\bar{\theta}$ as

$$\bar{\theta} = \bar{M}^{-1}(Y - \underline{M}\underline{\theta}). \tag{5.19}$$

The objective is now to find $\underline{\theta}$ that will minimize in some weighted fashion the values of (5.11) and (5.12). Let us rewrite (5.13) and (5.14) as matrix equations

$$\|y\|_\infty = \left\| W_y \begin{bmatrix} 1 \\ \bar{M}^{-1}(Y - \underline{M\theta}) \\ \underline{\theta} \end{bmatrix} \right\|_1 = \sum_{i=1}^{m} \left| f_i(\underline{\theta}) \right| =: F_y(\theta), \tag{5.20}$$

$$\|u\|_\infty = \left\| W_u \begin{bmatrix} \bar{M}^{-1}(Y - \underline{M\theta}), \\ \underline{\theta} \end{bmatrix} \right\|_1 = \sum_{i=m+1}^{2m} \left| f_i(\underline{\theta}) \right| =: F_u(\theta) \tag{5.21}$$

where $m = n_c + 1 + n_r$ and $f_i(\underline{\theta})$ is affine in $\underline{\theta} \in \mathsf{R}^{n_r+1-n_p}$ and $W_y \in \mathsf{R}^{(n_c+1+n_r)\times 2(n_p+1)}$ and $W_u \in \mathsf{R}^{(n_c+n_r)\times(2n_p+1)}$ are as follows;

$$W_y = \begin{pmatrix}
c_0 & 0 & \cdots & \cdots & \cdots & \cdots & 0 & \cdots & 0 \\
\vdots & \ddots & \ddots & & & & \vdots & & \vdots \\
c_{n_c} & \cdots & c_0 & \ddots & & & \vdots & & \vdots \\
0 & c_{n_c} & \cdots & c_0 & \ddots & & \vdots & & \vdots \\
\vdots & \ddots & \ddots & & \ddots & \ddots & \vdots & & \vdots \\
\vdots & & \ddots & c_{n_c} & \cdots & c_0 & 0 & & \vdots \\
\vdots & & & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & \cdots & \cdots & 0 & c_{n_c} & 0 & \cdots & 0
\end{pmatrix}, \tag{5.22}$$

$$W_u = \begin{pmatrix}
0 & \cdots & 0 & c_0 & 0 & \cdots & \cdots & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \ddots & & & \vdots \\
\vdots & & 0 & c_{n_c} & \cdots & c_0 & \ddots & & \vdots \\
\vdots & & \vdots & \ddots & \ddots & & \ddots & \ddots & \vdots \\
\vdots & & \vdots & & \ddots & c_{n_c} & \cdots & c_0 & 0 \\
\vdots & & \vdots & & & \ddots & c_{n_c} & \cdots & c_0 \\
\vdots & & \vdots & & & & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & \cdots & \cdots & \cdots & \cdots & 0 & c_{n_c}
\end{pmatrix}. \tag{5.23}$$

Expressions (5.20) and (5.21) represent the maximum output signal and maximum control signal in terms of the free regulator parameters. Here there is no constraint on the values of $\underline{\theta}$. As long as (5.19) is maintained, the resulting controller stabilizes system (5.1). With these expressions, we can now seek ways to select $\underline{\theta}$ such that $\|y\|_\infty$ is minimized in some compromised manner. To do this, first observe that (5.20) and (5.21) can be written in the general affine form for $f_i(\underline{\theta})$ as

$$f_i(\underline{\theta}) = \sum_{j=1}^{n} \alpha_{ij}\underline{\theta}_j + \beta_i; \qquad i = 1, \ldots, 2m; \qquad m \geq n. \tag{5.24}$$

Now, it makes sense to use the principles of *linear programming* (LP) to show that the limits of performance curve is defined by finitely many linear equations and that its gradient is monotonically nondecreasing.

Limits-of-Performance Curve

In the context of the ℓ_∞ index used here, the limits-of-performance curve is a graphical plot of $\|y\|_\infty$ verses $\|u\|_\infty$. In this subsection, we show that this curve is described by a finite number of linear equations with gradients monotonically nondecreasing, and we propose a systematic method to construct this curve.

Let us now define points $P \in \mathsf{R}^2$ with coordinates (p_u, p_y), where $p_u = F_u(\underline{\theta})$ and $p_y = F_y(\underline{\theta})$. Let the collection of all feasible coordinate pairs of P be represented by the region □ as shown in Figure 5.3.

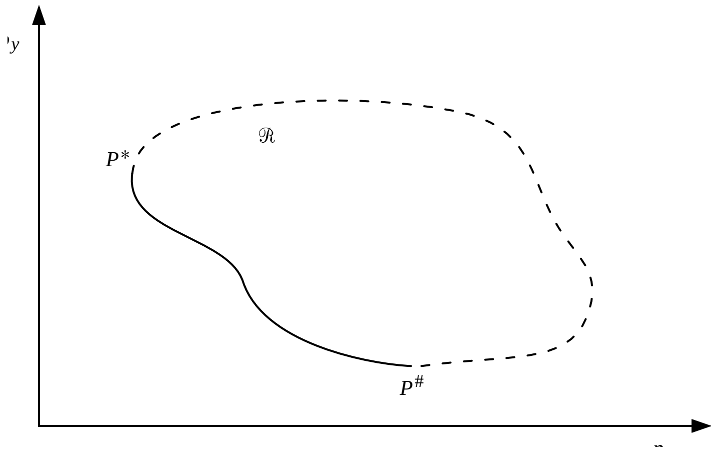


FIGURE 5.3. The region □ and the required contour line shown in solid line

Lemma 5.4. *Refering to Figure 5.3, consider the region □ and in particular the curve that defines that part of the boundary joining the point P^*, where the value of p_u is minimum, to the point $P^\#$, where the value for p_y is minimum. (This is the solid line shown in Figure 5.3). Then this section of the boundary is described by a finite number of linear equations. The vertices of this curve occur at those points when n of the $2m$ equations in* (5.24) *have intersecting solutions.*

Proof. The contour concerned can be determined by minimizing a cost function of the form

$$\min_{\underline{\theta}}(F_y(\underline{\theta}) + \lambda F_u(\underline{\theta})), \qquad 0 \le \lambda < \infty; \qquad \lambda \in \mathsf{R}. \tag{5.25}$$

For a given λ, the solution to this minimization problem will produce a set of points $(F_u(\underline{\theta}), F_y(\underline{\theta}))$ on the required curve, and thus solving (5.25) for $0 \le \lambda < \infty$ will achieve all points on this curve. To solve (5.25) for a fixed λ, we can reformulate the task as 2^{2m} sets of LP problems. The required solution is the minimum value of all the solutions to the 2^{2m} sets of LP problems. First let us define a matrix K of dimension $2^{2m} \times 2^{2m}$ with all rows distinct and the value of each of its elements either zero or one. In simpler terms, the rows of the matrix K generate all possible $2m$-bit binary numbers

$$K = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 & 1 \\ & & & & & \vdots & \vdots \\ 1 & 1 & 1 & \dots & 1 & 1 & 0 \\ 1 & 1 & 1 & \dots & 1 & 1 & 1 \end{pmatrix}. \tag{5.26}$$

Note that the minimization of (5.25) can be written as

$$\min_{\ell}\left\{\min_{\underline{\theta}}\left(\sum_{i=1}^{m}(-1)^{K_{\ell i}} f_i(\underline{\theta}) + \lambda \sum_{i=m+1}^{2m} (-1)^{K_{\ell i}} f_i(\underline{\theta})\right) \quad \text{subject to} \right.$$
$$\left. (-1)^{K_{\ell i}} f_i(\underline{\theta}) \ge 0; \qquad i = 1, \dots, 2m; \qquad \ell = 1, \dots, 2^{2m}\right\}. \tag{5.27}$$

Applying Fact 5.3, the region defined in (5.27) for each of the LP problems forms a convex region. Applying Facts 5.1 and 5.2, the minimum value of (5.27) is at one of the extreme points, which occurs when n of the constraints in (5.27) hold with equality. Note that the inequalities do not change the locations of the possible extreme points. Hence all the LP problems for $0 \le \lambda < \infty$ share some of these extreme points, and in total there are not more than ${}^{2m}C_n = (n(n-1)\dots(n-2m))/((2m)(2m-1)\dots1)$ number of extreme points.

Since the solution of (5.25), for $0 \le \lambda < \infty$, is the minimum value of all the LP problems defined implicitly in (5.27), the solution of (5.25) has to occur at one of the ${}^{2m}C_n$ intersection points. Hence we can conclude that the limits of the performance curve is formed by a finite number of straight lines, and the vertices occur at those points when n numbers of the equations in (5.25) intersect. □

Note that it is not required to solve all the 2^{2m} LP problems as mentioned above to obtain the solution of (5.25). The solution of (5.25) can be solved by evaluating the cost at all the ${}^{2m}C_n$ possible extreme points.

Lemma 5.5. *The gradients of the curve of the part of the boundary region □, as described in Lemma 5.4, are monotonically nondecreasing.*

Proof. As mentioned in the previous lemma, this curve is determined by solving the minimization problem given in (5.25) for λ between zero and infinity. To prove that the gradients of the curve are monotonically nondecreasing, we redefine the minimization problem (5.25) for a particular λ as described below. But first, let us introduce a new variable $\underline{\theta}_{n+1}$. The minimization of (5.25) is equivalent to

$$\min_{\underline{\theta},\underline{\theta}_{n+1}} \underline{\theta}_{n+1}, \tag{5.28}$$

subject to

$$\underline{\theta}_{n+1} \geq (-1)^{k_{\ell 1}} f_1(\underline{\theta}) + \cdots + (-1)^{K_{\ell m}} f_m(\underline{\theta}) + \cdots + (-1)^{k^{\ell(2m)}} f_{2m}(\underline{\theta}), \qquad \ell = 1 \ldots 2^{2m}. \tag{5.29}$$

Note that each of the inequalities defined in (5.29) is a semiplane that bounds a convex region. From Fact 5.3 the region defined by the inequalities forms a convex set and its edges are formed by the solution of linear equations. Again using Facts 5.1 and 5.2 we see that the solution space of (5.28) is a convex region. Consequently, the solutions of (5.25) for a given λ lie in a convex region.

In Lemma 5.4, we have shown that the limits-of-performance curve is formed by the solution of a finite number of linear equations. Therefore, minimization of (5.25) has either a single solution or an infinite number of solutions. Here we have shown that in the case where there are an infinite number of solutions, the solutions lie in a convex region. We conclude that the gradients of the limits-of-performance curve are strictly monotonically nondecreasing. □

From the above two lemmas, we conclude that the limits-of-performance curve are formed by the solution of a set of linear equations with gradients monotonically nondecreasing. Let us now present a systematic method for constructing this curve. The steps are given as follows:

Algorithm.

Step 0. Find all the intersection points of the $2m$ equations $(-1)^{K_{\ell i}} f_i(\underline{\theta}) \geq 0$; $i = 1, \ldots, 2m$; $\ell = 1, \ldots, 2^{2m}$ and calculate the coordinates $(\|y\|_\infty, \|u\|_\infty)$ at all the points. This involves solving ${}^{2m}C_n$ sets of n simultaneous equations and substituting the solution back to (5.20) and (5.21) to determine the coordinates of $(\|y\|_\infty, \|u\|_\infty)$. Determine the point P^* where $\|y\|_\infty$ is minimum. Label these two pairs of points as P_0 and P_1 respectively. Let $i = 0$ and $j = 1$. Go to Step 1.

Step 1. If $j \leq i$, go to Step 4, otherwise determine the line that joins P_i to P_{i+1}. Find from among the rest of the intersection points the one that has the smallest displacement from this line. Label this point as $P_{\min}$. If the minimum displacement is smaller than zero, go to Step 2, otherwise go to Step 3.

Step 2. For $k = j$ to $i + 1$ in decrements of 1, set $P_{k+1} = P_k$ and $P_{i+1} = P_{\min}$. Set $j = j + 1$. Go to Step 1.

Step 3. $i = i + 1$. Go to Step 1.

Step 4. The contour of the limits-of-performance linking the minimum achievable $\|u\|_\infty$ to the minimum achievable $\|y\|_\infty$ is now defined by the straight lines joining points P_0 to P_1, P_1 to $P_2, \ldots, P_{j-1}$ to P_j.

Remark. Due to the nature of the problem, one is required to solve ${}^{2m}C_n$ sets of n simultaneous equations. Therefore this method may not be computationally feasible for plants of very high orders ($n_p > 20$) or plants that use a relatively high order regulator. Table 5.1 shows the regulator order for a plant of varying order. Here F is the number of flops required to solve a set of n simultaneous equations, and T_F is the total number of MFLOPS required to solve ${}^{2m}C_n$ sets of n simultaneous equations.

We have not explored the possibility of using efficient combinatorial algorithms such as the so-called *branch and bound method* to solve this problem. Such techniques could prove useful, especially for large-scale problems, and the interested reader could consult Nemhauser and Wolsey (1988).

n_p	n_1	m	n	${}^{2m}C_n$	F/Flops	T_F/MFLOPS
2	11	12	10	1.96×10^6	2 500	4.9×10^3
4	12	13	9	3.12×10^6	1 840	5.8×10^3
5	13	14	9	6.9×10^6	1 840	13×10^3
8	15	16	8	10.5×10^6	1 350	14×10^3
10	16	17	7	5.3×10^6	950	5.2×10^3
15	20	21	6	5.25×10^6	620	3.2×10^3
20	25	26	6	20.3×10^6	620	13×10^3

TABLE 5.1. System and regulator order and estimated computation effort

Example. In this section, we will present the limits-of-performance curve for a second-order single-input, single-output, linear, time-invariant system. It has a pure unit delay, a nonminimum phase zero at $z = 2.0$ and two poles at $z = 1.2$, and $z = 0.7$. The system is given as

$$\left(1 - 1.2q^{-1}\right)\left(1 - 0.5q^{-1}\right) y_k = q^{-1}\left(1 - 2q^{-1}\right) u_k + w_k. \tag{5.30}$$

Let us consider regulators with a fifth- and eighth-order structure, respectively. The limits-of-performance curve for the eighth-order structure is shown in Figure 5.4. For the fifth-order regulator, the lines that make up the limits of performance curve are in effect the lowest three segments having gradients of -1.997,

-1.189 and -0.476 respectively. If λ happens to belong to this set, then the solutions that satisfy the optimization problem will be infinite in number. If, however, λ happens to be between these values, then the solution for the optimal control will remain unchanged.

From this example, it is clear that the weighted-sum method is not a suitable method for solving the optimization problem. For most of the weights chosen, the solutions remain the same. In the unlikely event where some particular weights are chosen, there are an infinite number of solutions. Given the limits-of-performance curve, one can now decide the compromise one has to make and select the appropriate operating point. For example, assuming the bound of the noise is normalized to unity and if the control effort is constrained to 3.5 units, the worst output signal will then be 6.1 units (shown as point A in Figure 5.4). In the case when the constraint on the output signal can be relaxed a little, we can choose to operate at a point such that the maximum output signal is less than 6.65 units with maximum control effort less than 2.5 units (shown as point B in Figure 5.4). In the case of unconstrained control effort, the best we can achieve for the output signal will be 5.7 units and the worst control effort is 4.4 units.

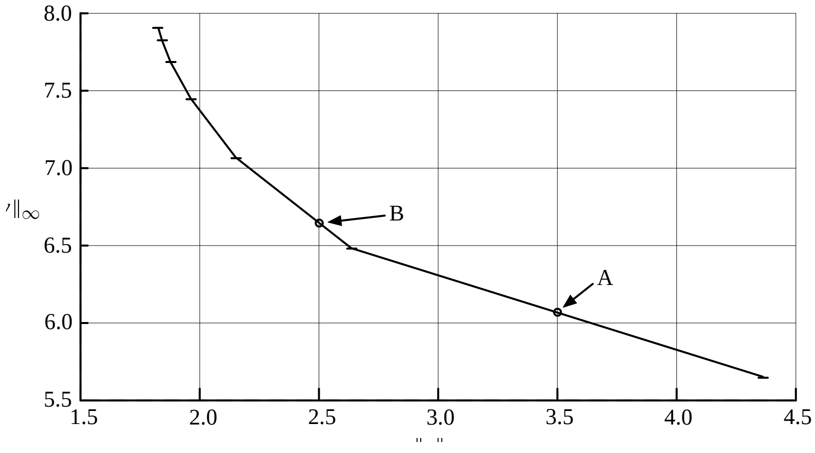


FIGURE 5.4. Limits-of-performance curve

Main Points of Section

In this section, the limitations of using the weighted-sum method or the weighted-maximum method for solving ℓ_1 optimization problems are illustrated. A limits-of-performance method to overcome this problem is described and the procedure to obtain this curve is illustrated. The advantage of this approach is that now we can do away with the trial and error method of selecting an appropriate cost function. Also the performance of the regulator structure can be observed and altered if necessary. Another advantage of this method is that with the knowledge of the maximum bound of the input disturbance, one can select the operating point so as to minimize one or more of the signals while keeping the other within its constraints.

4.6 Notes and References

This chapter aims to collect together in one place some contemporary controller design techniques. The stress is not on proving theorems, but rather on presenting the practical techniques to allow practicing engineers to quickly implement such controllers for their plants.

We begin with a discussion on performance specifications. These specifications can be found in almost every controller design based paper. However a good start would be books such as Boyd and Barratt (1991) and Vidyasagar (1985).

Linear quadratic control has been well explored. Good references are Anderson and Moore (1989) and Kwakernaak and Sivan (1972). The need for LQG/LTR was first raised in Doyle (1974) indicating a poor stability margin as a result of using state estimate feedback rather than state feedback. Subsequently stability margin recovery (LTR) methods were developed by researchers such as Doyle and Stein (1979), Zhang and Freudenberg (1987), Moore and Xia (1987), Moore, Gangsaas and Blight (1982), Lehtomaki, Sandell and Athans (1981), Stein and Athans (1987). The material of Section 4.3 is obtained from Moore and Tay (1989c), which is in many ways an advance on the previous works. It introduces the concept of sensitivity recovery, and this is done via a properly selected Q.

For ℓ_1 optimal design, the problem was first formulated by Vidyasagar (1986). Subsequently the work to solve the optimization problem for various situations were undertaken by Dahleh and Pearson (1987), Dahleh and Pearson (1988), Vidyasagar (1991) and their coworkers. The key approach in all these works is to formulate the desired closed-loop transfer function in terms of Q and then subsequently to convert the optimization into a linear programming problem by means of a dual formulation. The technique described in this chapter is based on the work by Teo and Tay (1995) and takes a polynomial setting. It takes the ℓ_1 solution a step further towards a practical design by proposing a technique to construct the entire limits-of-performance curve for all possible control energy penalties in a weighted sum performance index.

CHAPTER 5

Iterated and Nested (Q, S) Design

5.1 Introduction

In Chapter 4, we have presented controller design strategies to achieve various control performance objectives. For some of these methods the Q parameterization of Chapter 2 proved helpful. The assumption behind all the strategies is that the plant model is known. The controller is designed to reject in some optimal fashion certain classes of disturbances applied to this nominal model. There is no explicit provision to cope with plant perturbations or plant uncertainties in the designs. Of course it turns out that some of the designs are also robust to certain classes of plant perturbation or uncertainty. There can be a trade off between performance and robustness, but robustness to prescribed plant variations or uncertainties is not included explicitly in the optimization criteria, so that this trade off may not be straight forward.

In this chapter, we take a different approach to controller design from that of using the standard optimization techniques presented in Chapter 4. We will examine a controller design strategy that will take account of unmodeled dynamics in the nominal model of the plant. Here we view the controller $K(Q)$ as a nested two controller structure, consisting of the nominal controller K and an augmented controller Q, in the notation of Chapter 2, with $K = K(Q)|_{Q=0} = K(0)$. In the same vein, we also view a plant $G(S)$ as consisting of two parts; a nominal part G and an augmentation we call S in the notation of Chapter 3, with $G = G(S)|_{S=0} = G(0)$. The augmentation S is deemed to contain dynamics that are not modeled in the nominal part G of the actual plant. In fact, we have shown in Chapter 3 that the nominal model can be viewed as a simplified model of the actual plant obtained from, say, an initial identification procedure. The augmentation S then turns out to be a frequency-shaped deviation of the nominal model from the actual plant, with the frequency shaping emphasizing the cross-over frequencies of the nominal system and of the actual system.

The strategy adopted here is to first design the nominal controller K to optimally control the nominal plant G. The initially unmodeled dynamics represented by S is then identified leading to an estimate of S. This process can be carried out either on-line or off-line from measurements, as discussed in Chapter 3 and elaborated subsequently. Once an S is identified, the next stage is to design an augmented controller Q to optimally control S according to some performance measure, related to, but not necessarily identical to that of the initial controller design. This approach exploits the robust stabilization results in Chapter 3 where it is shown that when K stabilizes G, then $K(Q)$ stabilizes $G(S)$ if and only if Q stabilizes S.

In this chapter, we also take the work of Chapter 3 a step further by showing that from a performance point of view, the approach to design the nominal controller K for the nominal plant G, and the augmented controller Q for the derived plant S can be complementary. In fact there need be little danger of the augmented controller Q interacting with the nominal controller K in an adverse manner so that the original control object is compromised. In particular, we demonstrate how to achieve the complementary behavior for the familiar design methods of pole-placement, linear quadratic control and H_∞ control. We show that for these three techniques, designing the Q to control S, using the same criterion as that used in the design of the nominal controller K for G, ultimately assists in the achievement of the original design goal.

We then move on to extend the results of Chapter 3 in another direction. We show that in general, there need not be a limit to a nested two-controller structure. In fact, any plant can be represented in an m recursive fractional form via a continued fraction expansion. The results in Chapter 3 correspond to the special case where $m = 2$. Similarly, instead of the two-controller nested structure consisting of K and Q, we can generalize the two-controller structure to an m-controller nested structure for a plant represented in an m recursive fractional form. The stability results for the two-controller structure are then generalized to this multiple controller structure. With this generalization, we propose a multistage or an iterative controller design approach as a natural extension to the two-stage controller design approach. In some cases, a high order original plant may be decomposed to form a sequence of relatively simple plant models which allow successive approximation of the plant. The task of designing a complex controller for the high order plant can then be broken down into a sequence of lower order controller designs for a sequence of lower order models.

The iterated or multistage design methods of this chapter are focussed on a sequence of off-line controller improvements based on on-line identification, but the results lead naturally to on-line adaptive control as developed in the following two chapters.

5.2 Iterated (Q, S) Design

The results of Chapters 2 and 3 allow us to proceed with a controller design as a two-stage process.

The first stage is the design of a stabilizing nominal controller K for a nominal plant G, but also stabilizing the actual plant $G(S)$. In this section, we will think of this stage as an off-line design where we will do our best to obtain a controller for the plant based on the *a priori* knowledge of the actual plant, represented by the nominal model of the plant. Thus if there are no unmodeled dynamics, then this off-line designed controller should be the optimal controller for the actual plant, but if there are plant uncertainties the controller should be robust to these; maybe not achieving good performance, but at least ensuring stability.

We will view the second stage as an enhancement stage. Here we will utilize on-line measurements to identify the unmodeled dynamics in the original plant, represented by S. The matrix transfer function S is a frequency-shaped deviation of the nominal plant model from the actual plant. The augmented controller Q is then designed based on the identified S. The design method for Q is appropriately selected to ensure that there is no conflict of the original controller design goal. This can be interpreted as stating that after the implementation of Q, the implemented controller $K(Q)$ actually solves the design problem for the plant $G(S)$. Figure 2.1 illustrates the iterative-Q design process.

It is clear that the overall Q-enhancement procedure can be iterated. Indeed the identification of S may have been incomplete, or due to the new updated design deficiencies in the model may have been accentuated. Both may necessitate a repeating of the procedure.

Identification of Unmodeled Dynamics

Let us consider the problem of identifying S given that a particular controller with a Q augmentation has been implemented, as given in Figure 2.1. The matrix transfer function J and the operator Q form the feedback controller $K(Q)$ for the plant $G(S)$. For the purpose of the iterative scheme, the plant $G(S)$ and the matrix transfer function J are viewed as a single block W, such as is depicted in Figure 2.1. The signals r, s, w_1, u, v, and y are assumed to be measurable. The relationship between r, s, and w_1, w_2 can be written from the key operator equation (3.4.22) derived in Lemma 3.4.3 specialized to the representation of Figure 2.1 (see also (3.6.3)), and is given by

$$r = Ss + \tilde{M}(S)w_2 + \tilde{N}(S)w_1. \tag{2.1}$$

This is a linear equation in S and can be reformulated into a linear regression equation for a standard identification algorithm.

Besides the relationship expressed in (2.1) we also have the control relation-

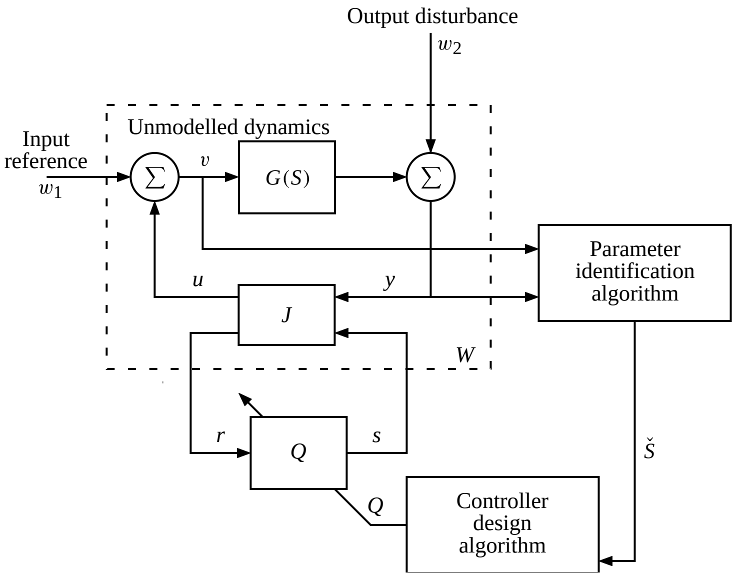


FIGURE 2.1. An iterative-Q design

ship:

$$s = Qr. \tag{2.2}$$

Recall also from (2.12) and (2.13) that, in the absence of w,

$$r = \tilde{M}y - \tilde{N}u, \qquad s = \tilde{V}u - \tilde{U}y. \tag{2.3}$$

Notice that in the first stage of the redesign process $Q = 0$ applies, as the only controller implemented is $K = K(0)$. In this situation (2.1) suffices for identification purposes. Unfortunately, due to the obvious feedback interconnection of (2.1), (2.2) it is unclear how to obtain an unbiased estimate for S, at least without adding excitation signals to s. Moreover, even without the feedback interconnection, due to the fact that in (2.1) the transfer functions between s, w_1, w_2 and r are not independently parameterized, it might be thought difficult to obtain an unbiased estimate of S. The problem is overcome without adding extra signals as follows.

Intuitively, in view of the closed loop (2.1), (2.2) it is clear that we can express both r and s in an affine manner in terms of the closed-loop transfer function of the feedback system (Q, S), and the external signals w_1 and w_2. If Q is known, we could then deduce S from knowledge of (Q, S). This is achieved in the following development, see also Figure 2.2.

In order to fix ideas, we assume here that the signal w_2 is an unmeasurable disturbance while w_1 is measurable. Any other combination can be treated in a similar fashion. Moreover, let us assume that w_1 and w_2 are uncorrelated signals, as defined in Section 3.2.

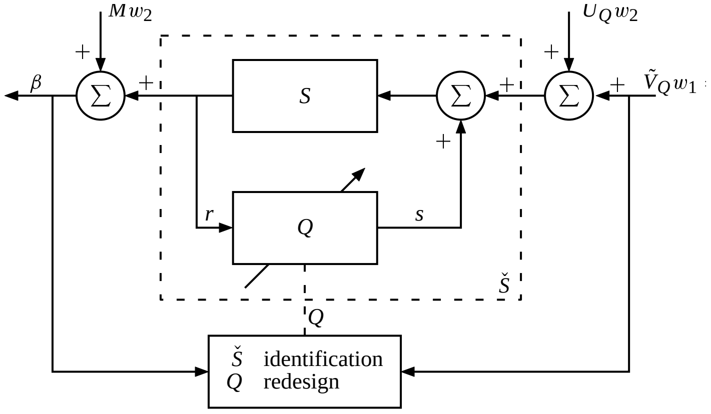


FIGURE 2.2. Closed-loop identification

Let us clarify the assumptions on $G(S)$, $K(Q)$ for the following. We assume that $(G = G(0), K = K(0))$ form a stabilizing pair, and that $(G(0), K(Q))$ and $(G(S), K(Q))$ are stabilizing. Thus we have assumed both that Q is stable and that Q stabilizes S.

With the above stability assumptions in mind, let us introduce coprime factorizations,

$$\begin{aligned} G &= G(0) = \tilde{M}^{-1}\tilde{N} = NM^{-1}, \\ K &= K(0) = \tilde{V}^{-1}\tilde{U} = VU^{-1}, \end{aligned} \tag{2.4}$$

and

$$K(Q) = U_Q V_Q^{-1} = \tilde{V}_Q^{-1}\tilde{U}_Q, \tag{2.5}$$

where

$$\begin{aligned} U_Q &= U + MQ, \qquad & V_Q &= V + NQ, \\ \tilde{U}_Q &= \tilde{U} + Q\tilde{M}, \qquad & \tilde{V}_Q &= \tilde{V} + Q\tilde{N}. \end{aligned} \tag{2.6}$$

Also,

$$G(S) = (\tilde{M} + S\tilde{U})^{-1}(\tilde{N} + S\tilde{V}) = (N + US)(M + VS)^{-1}, \tag{2.7}$$

and we introduce coprime factorizations for $G(S)$ parameterized in terms of S or $\check{S}$ as

$$G(S) = G_Q(\check{S}) = \tilde{M}_Q(\check{S})^{-1}\tilde{N}_Q(\check{S}) = N_Q(\check{S})M_Q(\check{S})^{-1}, \tag{2.8}$$

where

$$\begin{aligned} M_Q(\check{S}) &= M + U_Q\check{S}, \qquad & N_Q(\check{S}) &= N + V_Q\check{S}, \\ \tilde{M}_Q(\check{S}) &= \tilde{M} + \check{S}\tilde{U}_Q, \qquad & \tilde{N}_Q(\check{S}) &= \tilde{N} + \check{S}\tilde{V}_Q, \end{aligned} \tag{2.9}$$

(see also equation (3.4.3)). Also we have the double Bezout identity

$$\begin{bmatrix} \tilde{V}_Q & -\tilde{U}_Q \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} M & U_Q \\ N & V_Q \end{bmatrix} = \begin{bmatrix} M & U_Q \\ N & V_Q \end{bmatrix} \begin{bmatrix} \tilde{V}_Q & -\tilde{U}_Q \\ -\tilde{N} & \tilde{M} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{2.10}$$

Let us introduce the auxiliary variables α, β given from

$$\begin{bmatrix} \beta \\ \alpha \end{bmatrix} = \begin{bmatrix} -\tilde{N} & \tilde{M} \\ \tilde{V}_Q & -\tilde{U}_Q \end{bmatrix} \begin{bmatrix} v \\ y \end{bmatrix}. \tag{2.11}$$

The importance of the signals α, β for identification purposes can be inferred by expressing α, β in terms of the external signals driving the system in Figure 2.1. With reference to Figure 2.1 it is clear that, since $G(S) = G_Q(\check{S})$,

$$\begin{bmatrix} v \\ y \end{bmatrix} = \begin{bmatrix} I & -K(Q) \\ -G_Q(\check{S}) & I \end{bmatrix}^{-1} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},$$

where simple manipulations show

$$\begin{aligned}
&\begin{bmatrix} I & -K(Q) \\ -G_Q(\check{S}) & I \end{bmatrix}^{-1} \\
&\qquad = \begin{bmatrix} M_Q(\check{S}) & U_Q \\ N_Q(\check{S}) & V_Q \end{bmatrix} \begin{bmatrix} \tilde{V}_Q & 0 \\ 0 & \tilde{M}_Q(\check{S}) \end{bmatrix} \\
&\qquad = \begin{bmatrix} M_Q(\check{S}) \\ N_Q(\check{S}) \end{bmatrix} \begin{bmatrix} \tilde{V}_Q & \tilde{U}_Q \end{bmatrix} + \begin{bmatrix} 0 & -M_Q(\check{S})\tilde{U}_Q + U_Q\tilde{M}_Q(\check{S}) \\ 0 & -N_Q(\check{S})\tilde{U}_Q + V_Q\tilde{M}_Q(\check{S}) \end{bmatrix} \\
&\qquad = \begin{bmatrix} M_Q(\check{S}) \\ N_Q(\check{S}) \end{bmatrix} \begin{bmatrix} \tilde{V}_Q & \tilde{U}_Q \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}.
\end{aligned}$$

The last equality follows from the double Bezout equation (2.10) and the definition of $M_Q(\check{S})$ and $N_Q(\check{S})$. It follows then that using the definitions in (2.11), and the Bezout identity, we get

$$\begin{aligned}
\begin{bmatrix} \beta \\ \alpha \end{bmatrix} &= \begin{bmatrix} -\tilde{N} & \tilde{M} \\ \tilde{V}_Q & -\tilde{U}_Q \end{bmatrix} \left(\begin{bmatrix} M + U_Q(\check{S}) \\ N + V_Q(\check{S}) \end{bmatrix} \begin{bmatrix} \tilde{V}_Q & \tilde{U}_Q \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} \right) \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\
&= \left(\begin{bmatrix} \check{S} \\ I \end{bmatrix} \begin{bmatrix} \tilde{V}_Q & \tilde{U}_Q \end{bmatrix} + \begin{bmatrix} 0 & \tilde{M} \\ 0 & -\tilde{U}_Q \end{bmatrix} \right) \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\
&= \begin{bmatrix} \check{S}\tilde{V}_Q & \tilde{M}_Q(\check{S}) \\ \tilde{V}_Q & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},
\end{aligned}$$

from which we deduce that α is independent of w_2, and

$$\beta = \check{S}\alpha + \tilde{M}_Q(\check{S})w_2. \tag{2.12}$$

This has the interpretation that we can obtain an unbiased estimate of $\check{S}$ from measurements of β and α via, for example, output error identification schemes. For a treatment of open-loop identification methods, we refer the reader to Ljung (1987). An in depth treatise of the ideas presented above on closed-loop identification can be found in Lee (1994), Schrama (1992b) and Hansen (1989).

The above results are summarized as the following lemma.

Lemma 2.1 (Unbiased closed-loop identification). *Refer to the Figure 2.1. Assume that the controller $K(Q)$ stabilizes both the plant $G(S)$ and the nominal plant $G(0)$. Let $K(0) = UV^{-1} = \tilde{V}^{-1}\tilde{U}$, $G(0) = \tilde{M}^{-1}\tilde{N} = NM^{-1}$, $K(Q) = U_Q V_Q^{-1} = \tilde{V}_Q \tilde{U}_Q^{-1}$, as in* (2.4)*. Define the measurable signals α, β via* (2.11)*. Then as depicted in Figure 2.2*

$$\beta = \check{S}\alpha + (\tilde{M} + \check{S}\tilde{U}_Q)w_2, \qquad \alpha = \tilde{V}_Q w_1. \tag{2.13}$$

Provided the external signals w_1 and w_2 are uncorrelated, it follows that $\check{S}$ can be identified in an unbiased manner via correlation analysis of β and α.

Now $\check{S}$ and S are related by (2.8), in that

$$(N + VS)(M + US)^{-1} = (\tilde{M} + \check{S}\tilde{U}_Q)^{-1}(\tilde{N} + \check{S}\tilde{V}_Q),$$

where $\tilde{U}_Q = U + Q\tilde{M}$, $\tilde{V}_Q = \tilde{V} + Q\tilde{N}$. Hence we can deduce that

$$S = (I + \check{S}Q)^{-1}\check{S} = \check{S}(I + Q\check{S})^{-1}, \tag{2.14}$$

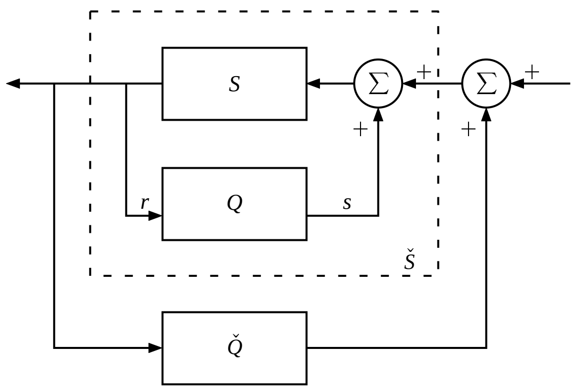
or

$$\check{S} = S(I - QS)^{-1} = (I - SQ)^{-1}S. \tag{2.15}$$

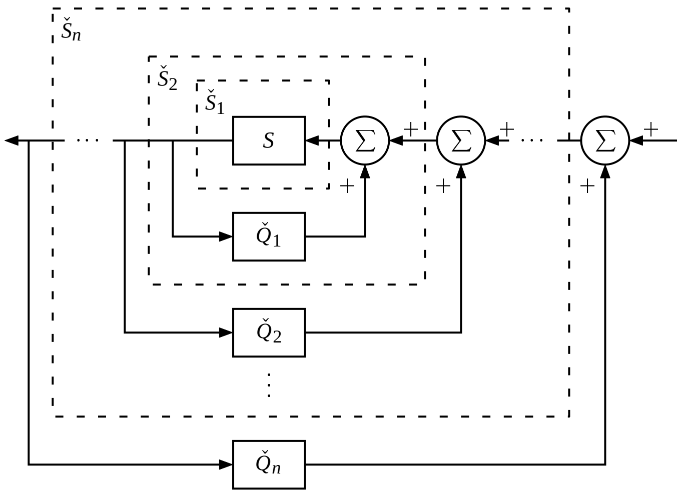
These expressions allow us to infer S from $\check{S}$ and vice versa. Reinterpreting the result of Lemma 2.1 in the light of the expressions (2.14) and (2.15) we come to the rather obvious statement that we are able to identify the closed-loop system (Q, S) from the data in an unbiased manner. This is depicted in Figure 2.2.

Iterated Control-identification Principle

Since we start with the assumption that $K(Q)$ stabilizes $G(S)$, once we obtain an estimate for $\check{S}$, we are guaranteed that the present controller $K(Q)$ will also stabilize $G_Q(\check{S})$ since $G_Q(\check{S}) = G(S)$ under (2.14), (2.15)). We are now in a position to update the controller Q so as to obtain an improved closed loop. Indeed having identified $\check{S}$ we can find an associated estimated S via equation (2.14). For this estimated S we can now design a new Q and the whole procedure can be repeated if desired.



One step iteration



Multistage iterated design

FIGURE 2.3. Iterated-Q design

We stress that it is important to base the identification on the $\check{S}$ representation rather that the S representation. This is because unbiased $\check{S}$ estimates can be obtained from standard identification algorithms. This is not the case for S estimates, at least without the addition of excitation signals in the Q, S loop.

Because we now want to redesign Q so as to stabilize the newly found S while achieving some design goal, and because of the relationship between $\check{S}$, S and Q, we have that the stabilizing pair (Q_1, S), where Q_1 denotes the new Q, is identical to the stabilizing pair $(\check{S}, \check{Q})$ where $\check{Q} = Q_1 - Q$. We may therefore proceed with designing a stabilizing controller for $\check{S}$ and *augment* the existing Q by this new controller to find the actual controller to be implemented.

The process is summarized in Figure 2.3 and detailed in the next algorithm.

Algorithm.

Step 0. • Initialize

$$G_0 = NM^{-1} = \tilde{M}^{-1}\tilde{N},$$
$$K_0 = UV^{-1} = \tilde{V}^{-1}\tilde{U},$$
$$Q_0 = \check{Q}_0 = 0, \qquad S_0 = \check{S}_0 = 0,$$
$$\ell = 0.$$

Step 1. •

$$\tilde{V}_\ell = \tilde{V} + Q_\ell \tilde{N}, \qquad \tilde{U}_\ell = \tilde{U} + Q_\ell \tilde{M},$$
$$\tilde{M}_\ell = \tilde{M} + \check{S}_\ell \tilde{U}_\ell, \qquad \tilde{N}_\ell = \tilde{N} + \check{S}_\ell \tilde{V}_\ell,$$
$$K_\ell = \tilde{V}_\ell^{-1}\tilde{U}_\ell, \qquad G_\ell = \tilde{M}_\ell^{-1}\tilde{N}_\ell,$$
$$\alpha_\ell = \tilde{V}_\ell v - \tilde{U}_\ell y = \tilde{V}_\ell w_1, \qquad \beta_\ell = -\tilde{N}v + \tilde{M}y.$$

- Identify $\check{S}$ from $\beta_\ell = \check{S}\alpha_\ell + (\tilde{M} + \check{S}\tilde{U}_\ell)w_2$ using an output error identification method.
- Update control by designing $(\check{S}, \check{Q})$.

Step 2. • Output

$$\check{S}_{\ell+1} = \check{S},$$
$$Q_{\ell+1} = Q_\ell + \check{Q}.$$

Step 3.
- Either let $\ell + 1$ be ℓ, go to Step 1.
- Or stop if control objective is achieved.

The key idea in an iterated control design is thus to first stabilize the plant with some nominal robust controller, presumably not achieving high performance. Consequently, with the stabilizing controller in the loop we reidentify the system, or better the S-factor representation. Then we redesign the controller, or better the Q parameter, to obtain improved performance. This may be iterated. The reason why iterations are necessary stems from the fact that no one iterate is capable of representing the plant accurately. The success of this method hinges on our ability to use low order approximations for $\check{S}$ as is exemplified in Chapter 3. Even then, for practical implementation, controller order reduction may be required, as each iteration augments the controller order.

We now discuss how the control design for some specific control objectives can proceed in this iterated framework.

Iterated Pole-placement Strategy

We now present a result to show the rationale behind designing Q using a pole-placement algorithm. Let us examine the relationship among the closed-loop systems formed by the pair $(G(S), K(Q))$ representing the actual closed loop, the pair (G, K) representing the nominal or design control loop and the pair (Q, S). We show that the eigenvalues of the closed-loop system formed by $(G(S), K(Q))$

consists of the set of eigenvalues of the actual closed-loop system represented by (G, K) together with the set of eigenvalues of the closed-loop system (Q, S). We summarize the result in the following theorem.

Theorem 2.2. *Let (G, K) be a stabilizing nominal plant-controller pair in that* (2.3.5), (2.3.6) *holds. Let $G(S)$ be the class of plants parameterized by S as in* (2.7) *with $G = G(0)$, and let $K(Q)$ be the class of controllers parameterized by Q as in* (2.4) *with $K = K(0)$. Under these conditions, the set of closed-loop poles of the pair $(G(S), K(Q))$ is generically the union of the set of poles of the pairs (G, K) and (Q, S).*

Proof. First recall equation (3.4.18) from Chapter 3,

$$\begin{bmatrix} I & -K(Q) \\ -G(S) & I \end{bmatrix}^{-1}$$
$$= \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} + \begin{bmatrix} M & U \\ N & V \end{bmatrix} \left\{ \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} - I \right\} \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix},$$

from which the result follows, since it turns out that the coefficient matrices in the second term introduce no additional dynamics to the dynamics of the first term. Notice that any pole/zero cancellations necessarily involve stable pole/zero cancellation by construction, as $M, U, N, V, \tilde{M}, \tilde{U}, \tilde{N}, \tilde{V}$ all represent stable rational transfer functions, as well as (Q, S). □

This theorem indicates that if the nominal controller K for the nominal plant G is designed based on a pole-placement technique, then the additional poles arising in the case of unmodeled dynamics can be assigned to appropriate locations using the ‘controller’ Q. There is no conflict with design objectives in each stage of the design.

Other important interpretations are obtained by considering the closed-loop arrangement $(G(S), K(0))$ as the plant to be controlled by Q. The closed-loop behavior, according to Theorem 2.2, is characterized by the actual desired design behavior (G, K) together with that of $(S, 0)$. In other words, S models the difference between the desired and the actual closed loop. Therefore, Theorem 2.2 implies that in order to achieve a desired control objective such as a certain control bandwidth, one should design the nominal controller K so as to achieve this design goal on the nominal plant model G and iterate the same design goal for the control loop (Q, S).

The limitation with the above methodology is that in the presence of significant model errors, our control objective must be a cautious one. Otherwise, achieving high performance on a nominal design could lead to an unstable closed-loop response for $(G(S), K(0))$, or equivalently, an unstable S. Indeed, the above theory allows for this since an unstable S can be stabilized by an appropriate Q. Nominally however, it makes sense to proceed with a cautious design objective, and iterate to an improved model for the plant keeping the same control objective. When the nominal model response and the actual closed-loop response are in close agreement, we can then envisage changing the control objective, that is, changing the desired closed-loop poles. The whole process can then be started over, until such time that we are satisfied with the closed-loop response.

From the previous discussion on iterated design and Theorem 2.2, it follows that at any stage of an iterated pole placement design, the closed-loop poles of the system are the poles of (G, K) and $(\check{S}_n, \check{Q}_n) = (S, \check{Q}_1 + \check{Q}_2 + \cdots + \check{Q}_n)$. Therefore, at no stage in the iterative design process is there a conflict in the design process, even when we work with $\check{S}$, $\check{Q}$ variables.

Example. Let G be a second order plant with poles at $2.348\,4 \pm j1.628\,2$ and K be a pole placement controller that assigns the closed-loop poles to the locations $(0.8, 0.8, 0.7, 0.7)$. The parameters of the various transfer functions are given in Table 2.1. Let us work with a first order S and let Q be the controller that assigns the closed-loop poles of (Q, S) to $(0.75, 0.75)$.

We now construct $G(S)$ and $K(Q)$. Notice that the closed-loop transfer function $G(S)(1 - K(Q)G(S))^{-1}$ as given in Table 2.1 has identical poles to those $G(1 - KG)^{-1}$ and $S(1 - QS)^{-1}$, as predicted by Theorem 2.2.

Iterated Linear Quadratic Design Strategy

We first present a result which explains the rationale behind the design of controller $K(Q)$ using the linear quadratic technique. We refer to Figure 2.1. All external disturbances are assumed to be zero, i.e. $w_1 \equiv w_2 \equiv 0$. (In this case, $u = v$ in Figure 2.1).

Theorem 2.3. *With (G, K) a stabilizing plant-controller pair, consider a plant $G(S)$ with a controller $K(Q)$ applied for some Q, as in Figure 2.1, see also factorizations* (2.4), (2.7). *Consider also a linear quadratic index, penalizing the controller internal signals r and s (being the inputs and outputs of Q, respectively), as*

$$J_{LQ} = \lim_{k\to\infty} \sum_{i=1}^{k} (r_i' r_i + s_i' R s_i). \tag{2.16}$$

where R is a symmetric positive definite weighting matrix. Then this index can be expressed in terms of a frequency-shaped penalty on the plant outputs and inputs, y, u as

$$J_{LQ} = \lim_{k\to\infty} \sum_{i=1}^{k} \begin{bmatrix} y_i' & u_i' \end{bmatrix} \begin{bmatrix} (\tilde{M}^*\tilde{M} + \tilde{U}^*R\tilde{U}) & -(\tilde{M}^*\tilde{N} + \tilde{U}^*R\tilde{V}) \\ -(\tilde{N}^*\tilde{M} + \tilde{V}^*R\tilde{U}) & (\tilde{N}^*\tilde{N} + \tilde{V}^*R\tilde{V}) \end{bmatrix} \begin{bmatrix} y_i \\ u_i \end{bmatrix}, \tag{2.17}$$

*where * denotes conjugate transpose.*

Transfer functions	Poles
$G = \frac{1.389\,5z^{-1}-2.879\,3z^{-2}}{1-4.696\,9z^{-1}+8.166\,2z^{-2}}$	$2.348\,4 \pm j1.628\,2$
$K = \frac{-7.100\,8z^{-1}+19.090\,9z^{-2}}{1+1.696\,9z^{-1}-6.692\,8z^{-2}}$	$-3.571\,1,\ 1.874\,2$
$G(1-KG)^{-1} = \frac{1.389\,5z^{-1}-0.521\,4z^{-2}-14.185\,5z^{-3}+19.270\,3z^{-4}}{1-3z^{-1}+3.37z^{-2}-1.68z^{-3}+0.313\,63z^{-4}}$	$0.8,\ 0.8,\ 0.7,\ 0.7$
$S = \frac{z^{-1}}{1-0.9z^{-1}}$	0.9
$Q = \frac{-0.022\,5z^{-1}}{1-0.6z^{-1}}$	0.6
$S(1-QS)^{-1} = \frac{z^{-1}-0.6z^{-2}}{1-1.5z^{-1}+0.562\,5z^{-2}}$	$0.75,\ 0.75$
$G(S) = \frac{2.389\,5z^{-1}-2.433z^{-2}-4.101\,4z^{-3}}{1-5.596\,9z^{-1}+5.292\,6z^{-2}+11.741\,3z^{-3}}$	$3.294\,6 \pm j0.988\,4,$ $-0.992\,4$
$K(Q) = \frac{-7.123\,3z^{-1}+23.457z^{-2}-11.638\,3z^{-3}}{1-1.096\,9z^{-1}-7.742\,2z^{-2}+4.804z^{-3}}$	$-3.578\,8,\ 1.873\,3,$ $0.608\,7$
$G(S)(1-K(Q)G(S))^{-1} = \frac{2.389\,5z^{-1}+0.188\,1z^{-2}-25.087\,8z^{-3}+24.087\,8z^{-4}+21.826\,3z^{-5}-16.735\,63z^{-6}}{1-4.5z^{-1}+8.432\,5z^{-2}-8.422\,5z^{-3}+4.729\,2z^{-4}-1.415\,4z^{-5}+0.176\,4z^{-6}}$	$0.8,\ 0.8,\ 0.75,$ $0.75,\ 0.7,\ 0.7$

TABLE 2.1. Transfer functions

Proof. We have that with $w_1 = w_2 = 0$:

$$\begin{bmatrix} s \\ r \end{bmatrix} = \begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix}. \tag{2.18}$$

Substituting this expression into (2.16), the result follows. □

Example. We use the nominal plant G given in Table 2.1. An LQG controller penalizing the index

$$J_1 = \lim_{k\to\infty} \sum_{i=1}^{k} (y_i^2 + \lambda u_i^2), \qquad \lambda = 0.1,$$

is used to design the nominal controller K. Similarly, the transfer function S of Table 2.1 is used to design Q using the index

$$J_2 = \lim_{k\to\infty} \sum_{i=1}^{k} (r_i^2 + \lambda s_i^2), \qquad \lambda = 0.1.$$

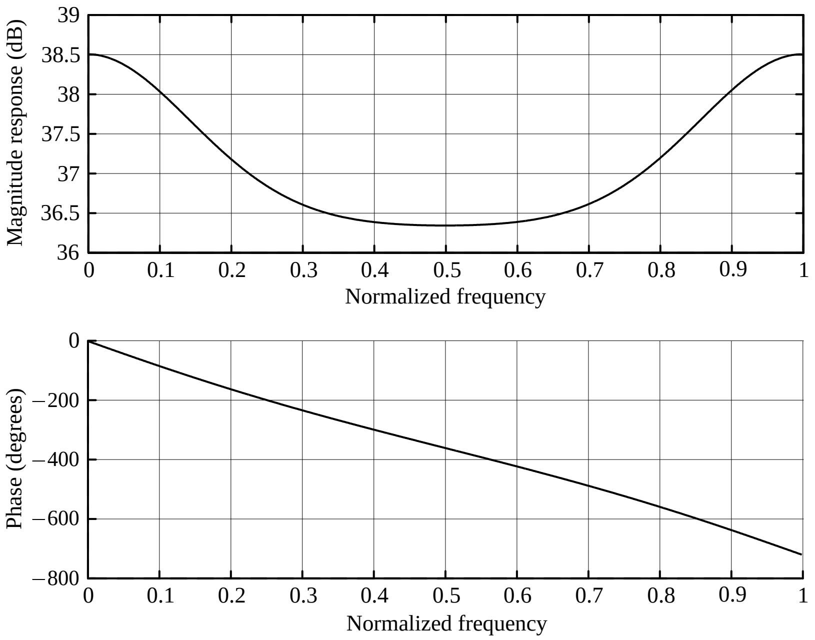


FIGURE 2.4. Frequency shaping for *y*

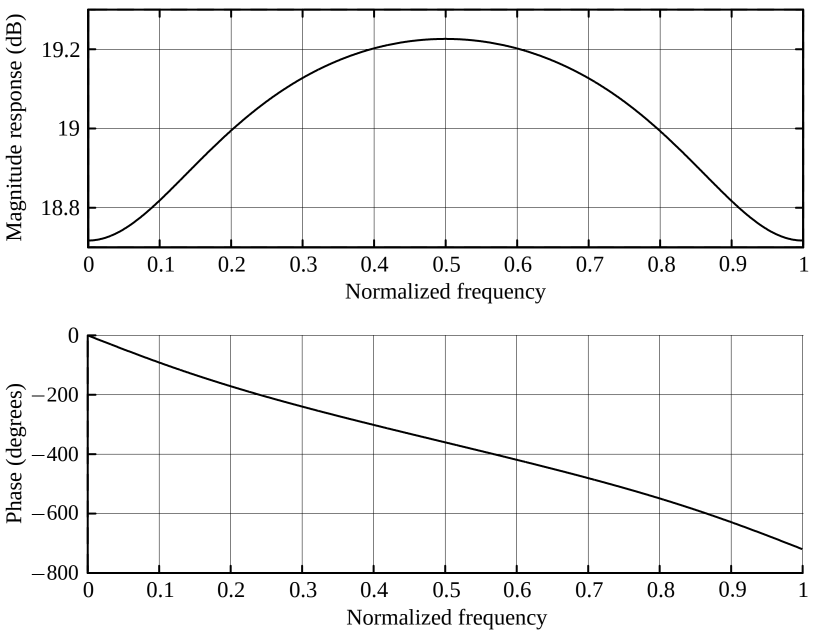


FIGURE 2.5. Frequency shaping for *u*

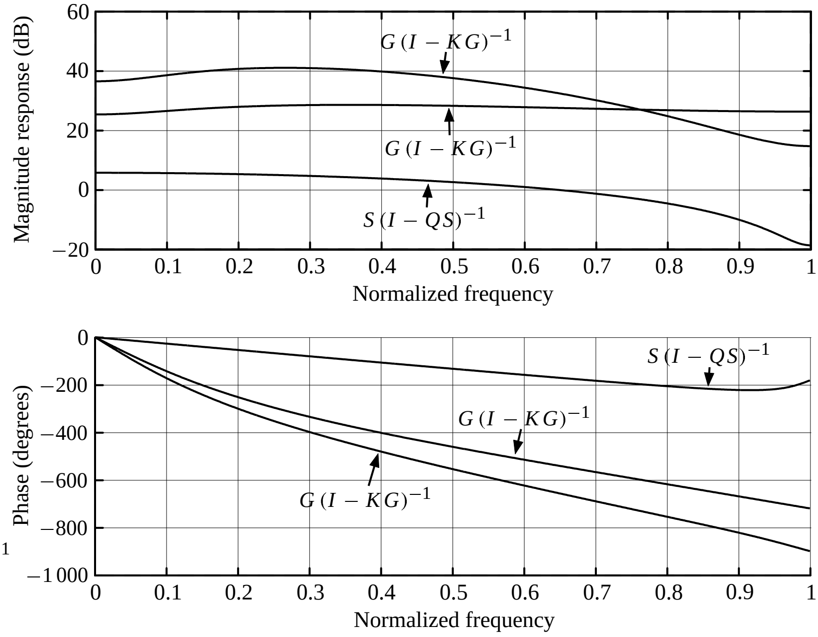


FIGURE 2.6. Closed-loop frequency responses

Figure 2.4 and Figure 2.5 show the magnitude/phase plots of $(\tilde{M}^*\tilde{M} + \tilde{U}^*\lambda^2\tilde{U})$ and $(\tilde{N}^*\tilde{N} + \tilde{V}^*\lambda^2\tilde{V})$ which are, respectively, the frequency shaped penalties for y and u as a result of using the index J_2. The frequency shapings exhibit a higher penalty at the low frequency range for y, and conversely for u. Figure 2.6 shows the frequency responses for the closed-loop transfer functions.

Theorem 2.3 tells us that performing an LQ design based on the penalty of r and s is equivalent to performing an LQ design based on a frequency shaped penalty of y and u as in (2.17). We recall that $\tilde{M}$, $\tilde{N}$, $\tilde{U}$, $\tilde{V}$ reflect the closed-loop poles of the pair (G, K). Thus in this case, we can interpret the frequency shaping given in (2.17) as an emphasis on y, u in the pass band of the nominal design, hence in the frequency bands of interest. It is therefore a meaningful index to minimize.

Iterated H_∞ Design Strategy

In order to fix ideas, we discuss here an iterative H_∞ design in the case where the performance variable is simply $e = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$. The generalized plant description then takes the form, referring to Figure 2.1, with $\bar{G} = G(S)$,

$$\begin{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} \\ y \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 0 \\ \bar{G} & I \end{bmatrix} & \begin{bmatrix} I \\ \bar{G} \end{bmatrix} \\ \begin{bmatrix} \bar{G} & I \end{bmatrix} & \bar{G} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ u \end{bmatrix} = \bar{P} \begin{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ u \end{bmatrix}. \tag{2.19}$$

The standard H_∞ problem for (2.19) is one of designing a controller $\bar{K}$ as to minimize the H_∞ norm of the transfer function matrix from the disturbances $w = \left[\begin{smallmatrix} w_1 \\ w_2 \end{smallmatrix}\right]$ to the performance variable $e = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$.

Here we would like to approach the H_∞ optimization problem in an iterated fashion. Let us first solve the H_∞ problem for the nominal plant, and next perform the H_∞ optimization problem for the unmodeled dynamics. The first subproblem is solved in terms of K and G, and the second is solved in terms of S and Q.

It remains to be seen how this suboptimal H_∞ control design method for the actual plant $G(S)$ is related to the complete H_∞ control design problem. Working with our standard notation, let $\bar{G} = G(S)$, $\bar{K} = K(Q)$. Assume that K solves the H_∞ problem for the nominal plant model P, in that K minimizes the H_∞ norm of the transfer function matrix $\Box(P, K)$ from the disturbances $w = \left[\begin{smallmatrix} w_1 \\ w_2 \end{smallmatrix}\right]$ to the performance variable $e = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ in the nominal system model. This transfer function is given by:

$$\begin{aligned} \Box(P, K) &= \begin{bmatrix} 0 & 0 \\ G & I \end{bmatrix} + \begin{bmatrix} I \\ G \end{bmatrix} K \begin{bmatrix} I - GK \end{bmatrix}^{-1} \begin{bmatrix} G & I \end{bmatrix} \\ &= \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1} - \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}. \end{aligned} \tag{2.20}$$

Here P is given by

$$P = \begin{bmatrix} \begin{bmatrix} 0 & 0 \\ G & I \end{bmatrix} & \begin{bmatrix} I \\ G \end{bmatrix} \\ \begin{bmatrix} G & I \end{bmatrix} & G \end{bmatrix}.$$

The difference between the actual transfer function linking disturbances and performance variable and its nominal version is given by

$$\Box(\bar{P}, \bar{K}) - \Box(P, K) = \begin{bmatrix} I & -\bar{K} \\ -\bar{G} & I \end{bmatrix}^{-1} - \begin{bmatrix} I & -K \\ -G & I \end{bmatrix}^{-1}.$$

Using the expression derived in Chapter 3, see (3.4.18) Theorem 3.4.2, this may be re-expressed as

$$\Box(\bar{P}, \bar{K}) - \Box(P, K) = \begin{bmatrix} M & U \\ N & V \end{bmatrix} \left\{ \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} - I \right\} \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix}.$$

Clearly, this can be reinterpreted as a frequency weighted left fractional representation for the system:

$$\bar{S} = \begin{bmatrix} \begin{bmatrix} 0 & 0 \\ S & 0 \end{bmatrix} & \begin{bmatrix} I \\ S \end{bmatrix} \\ \begin{bmatrix} S & I \end{bmatrix} & S \end{bmatrix} \tag{2.21}$$

as indeed

$$\begin{aligned} \square(\bar{S}, Q) &= \begin{bmatrix} 0 & 0 \\ S & 0 \end{bmatrix} + \begin{bmatrix} I \\ S \end{bmatrix} Q\,(I - SQ)^{-1} \begin{bmatrix} S & I \end{bmatrix} \\ &= \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} - \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \end{aligned} \tag{2.22}$$

The above expressions lead to the following theorem.

Theorem 2.4. *Consider the block diagram of Figure 2.1. Let the model be as defined in* (2.19) *and let the controller be* $\bar{K} = K(Q)$ *of* (2.5). *Under these conditions we have*

$$\square(\bar{P}, \bar{K}) - \square(P, K) = \begin{bmatrix} M & U \\ N & V \end{bmatrix} \square\left(\bar{S}, Q\right) \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix}.$$

It follows that the H_∞ design of $\bar{K}$ for the plant $\bar{P}$ can be approximated by solving two H_∞ design problems. The first H_∞ design is to find an optimal controller K for the nominal plant P, the next H_∞ design is a frequency weighted H_∞ design of the Q-factor on the plant $\bar{S}$. Clearly, this leads to a suboptimal H_∞ design for the plant, as indeed:

$$\begin{aligned} &\min_{\bar{K}} \left\| \square\left(\bar{P}, \bar{K}\right) \right\|_\infty \\ &\qquad \le \min_{K} \left\| \square\,(P, K) \right\|_\infty + \min_{Q} \left\| \begin{bmatrix} M & U \\ N & V \end{bmatrix} \square\left(\bar{S}, Q\right) \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix} \right\|_\infty . \end{aligned}$$

Even so, the additional cost may well be acceptable since the iterated design process may at every stage be much less complex than solving the overall problem at once. The following example illustrates the principle.

Example. Consider a true plant $\bar{G}$ as

$$\bar{G} : \left[\begin{array}{cccccc|c} 0.9976 & 0.0464 & -0.0002 & 0.0066 & 0.0007 & -0.0002 & -0.0012 \\ -0.0938 & 0.8575 & -0.0162 & 0.2174 & 0.0376 & -0.0106 & -0.0441 \\ -0.0036 & -0.0036 & 0.8138 & -0.3731 & 0.0639 & -0.0086 & 0.0083 \\ -0.0088 & -0.0088 & -0.1600 & 0.3598 & 0.1964 & -0.0395 & 0.0202 \\ -0.1242 & -0.1242 & 0.1024 & -1.4255 & 0.7347 & 0.0988 & 0.2937 \\ -0.0351 & -0.0351 & -0.0774 & 0.7071 & -0.8844 & 0.3290 & 0.0883 \\ \hline -1.5000 & -5.0000 & -3.0000 & 2.5000 & -2.0000 & 0 & 0 \end{array}\right] \tag{2.23}$$

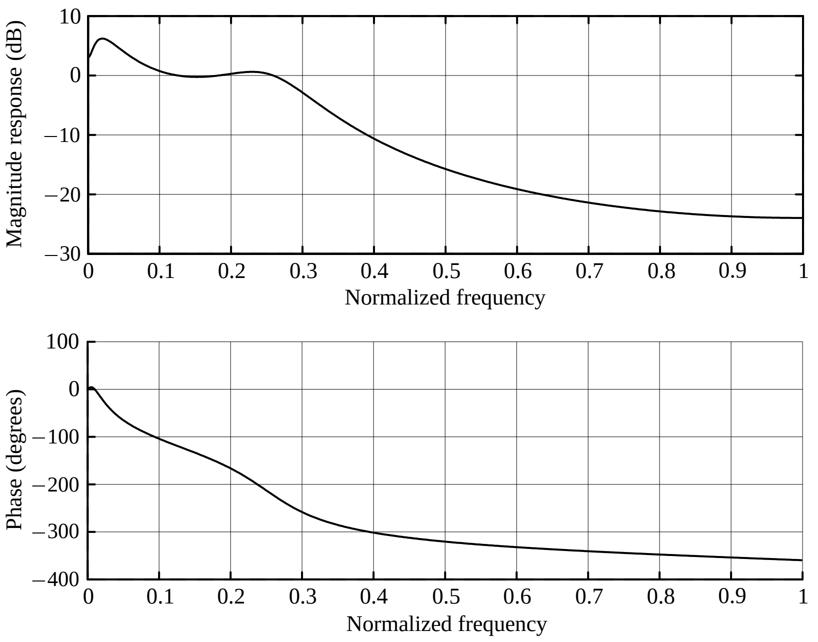


FIGURE 2.7. Modeling error $\left\|\bar{G} - G\right\|$

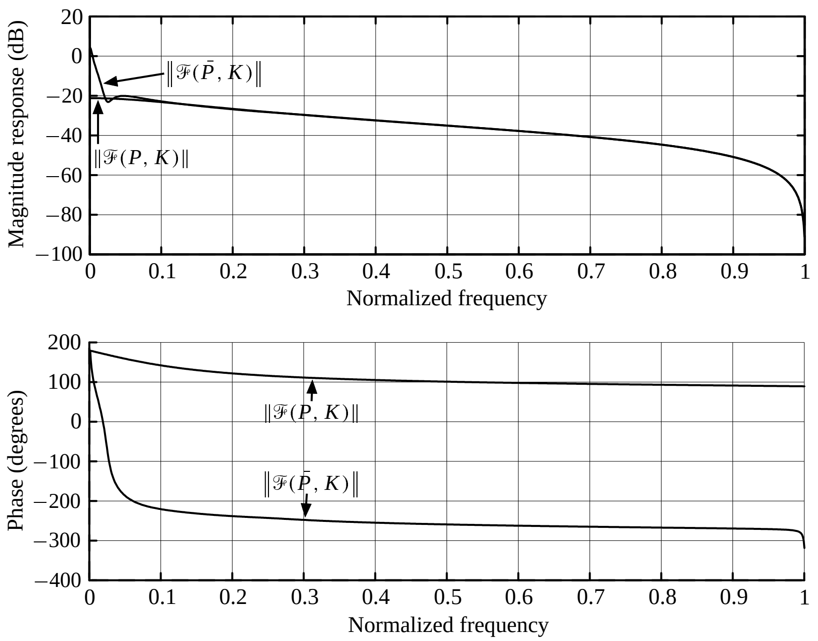


FIGURE 2.8. Magnitude and phase plots of $\square(P, K)$, $\square(\bar{P}, K)$

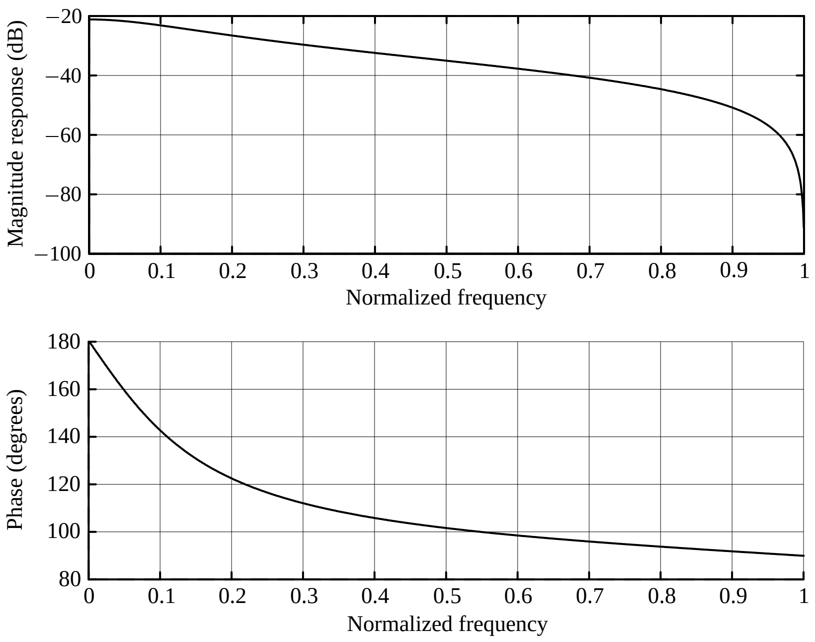


FIGURE 2.9. Magnitude and phase plots of $\square\left(\bar{P}, K(Q)\right)$

and a nominal generalized plant $P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$ with $P_{22} = G$ as

$$P : \left[\begin{array}{cccc|cc} 0.8584 & -0.0928 & 0 & 0 & 0.0464 & 0 \\ 0.0464 & 0.9976 & 0 & 0 & 0.0012 & 0 \\ 0 & 0 & 0.8584 & -0.0928 & 0 & 0.0464 \\ 0 & 0 & 0.0464 & 0.9976 & 0 & 0.0012 \\ \hline -1 & 3 & -12 & -20 & 0 & 1 \\ 1 & -1 & -5 & -2 & 1 & 0 \end{array}\right]. \tag{2.24}$$

The resulting modeling error $\bar{G} - G$ is depicted in Figure 2.7. An optimal H_∞ controller is found to be

$$K : \left[\begin{array}{cccc|c} 0.9524 & -0.0538 & -0.0105 & 0.0141 & -0.0232 \\ 0.0061 & 0.6539 & -0.1889 & -0.0587 & -0.1476 \\ 0 & 0.1900 & 1.0561 & 0.0978 & 0.1101 \\ 0 & 0 & 0.1549 & 0.7222 & 0.0495 \\ \hline 0 & 0 & 0 & 0.2035 & -0.0150 \end{array}\right]. \tag{2.25}$$

It turns out that K stabilizes both $\bar{G}$ and G. The magnitude plots of $\square(P, K)$, $\square(\bar{P}, K)$ are depicted in Figure 2.8. The H_∞-norm of the closed-loop transfer function $\square(\bar{P}, K)$ is well above that for $\square(P, K)$, although in most of the frequency spectrum $\left\|\square(\bar{P}, K)\right\|$ is virtually identical to $\|\square(P, K)\|$, see Figure 2.8. The difference at low frequencies is due to unmodeled dynamics.

To design the additional controller Q, we choose $\tilde{V}$, $\tilde{U}$, $\tilde{N}$ and $\tilde{M}$ based on G and K. Thereby the frequency-shaped modeling error S can be generated. But, we only use its second order balanced-truncation model to form the generalized error model $\bar{S}$ via (2.21). Then an optimal H_∞ controller Q for $\bar{S}$ is designed.

The controller $K(Q) = (U + MQ)(V + NQ)^{-1}$ evidently stabilizes $\bar{G}$. Not unexpectedly, $\left\| \square(\bar{P}, K(Q)) \right\|$ is dramatically reduced at low frequencies relative to $\left\| \square(\bar{P}, K) \right\|$, see Figure 2.9.

Main Points of Section

Off-line controller design using any of the available methods may not lead to controllers that perform well on an actual plant. Reidentification can be used to achieve improved controllers. The methods proposed here do not reidentify the plant itself, but rather identify a version of the difference between the plant and its model used for an initial design, denoted S. A controller Q is now designed for S, working with performance objectives which do not conflict with those of the initial design, but rather support these objectives. The controller Q is then a 'plug-in' controller for augmentation of the initial feedback control closed-loop.

We observe that, in general, it is not possible to identify S accurately from closed-loop data. Rather, we identify $\check{S} = (I - SQ)^{-1}S$, which is in one-to-one correspondence with S, given Q. In the case of pole placement design, we could proceed with an iterated control-identification design using $\check{S}$, $\check{Q}$. However, for H_∞ and LQ control design, it is important to recover S and then design Q accordingly.

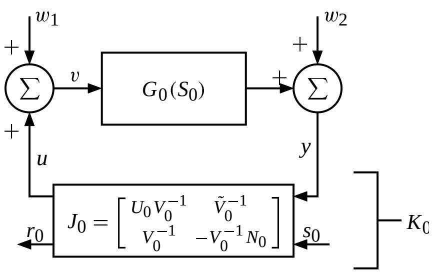
5.3 Nested (Q, S) Design

In this section, we redevelop the ideas of the previous section. The central idea is to realize that given a plant-controller pair $(G(S), K(Q))$, the design of a controller may be viewed as a design involving the pair (Q, S). Clearly it is then possible to view this as a problem of the form $(Q(Q_1), S(S_1))$, etc. This heuristic is made more precise in the sequel. The presentation here builds on the results established in Section 5.2.

First we justify the nested (Q, S) identification-control cycle in a heuristic manner. Then we develop a number of representation results to establish that the nested (Q, S) procedure is capable of achieving arbitrary control objectives.

Heuristics

To fix the ideas, let $\bar{G}$ represent the plant and denote $G = G_0$ as an initial model. Let K_0 denote an initial stabilizing controller, stabilizing both the nominal plant model G_0 as well as the plant $\bar{G}$. Consider factorizations $G_0 = N_0 M_0^{-1} =$



w_1
w_2
$+$
v
$G_0(S_0)$
$+$
$+$
$+$
u
y
$J_0 = \begin{bmatrix} U_0V_0^{-1} & \tilde{V}_0^{-1} \\ V_0^{-1} & -V_0^{-1}N_0 \end{bmatrix}$
r_0
s_0
K_0

FIGURE 3.1. Step 1 in nested design

$\tilde{M}_0^{-1}\tilde{N}_0$ and $K_0 = U_0V_0^{-1} = \tilde{V}_0^{-1}\tilde{U}_0$ satisfying the double Bezout equation

$$\begin{aligned}\begin{bmatrix} M_0 & U_0 \\ N_0 & V_0 \end{bmatrix}\begin{bmatrix} \tilde{V}_0 & -\tilde{U}_0 \\ -\tilde{N}_0 & \tilde{M}_0 \end{bmatrix} &= \begin{bmatrix} \tilde{V}_0 & -\tilde{U}_0 \\ -\tilde{N}_0 & \tilde{M}_0 \end{bmatrix}\begin{bmatrix} M_0 & U_0 \\ N_0 & V_0 \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}.\end{aligned} \tag{3.1}$$

We have for some S_0:

$$\begin{aligned}\bar{G} = G_0(S_0) &= (\tilde{M}_0 + S_0\tilde{U}_0)^{-1}(\tilde{N}_0 + S_0\tilde{V}_0) \\ &= (N_0 + V_0S_0)(M_0 + U_0S_0)^{-1}.\end{aligned}$$

In order to identify S_0, we will now inject a signal s_0 into the control loop (see Figure 3.1), which we assume to be uncorrelated to the input reference signal w_1 and output disturbance w_2. This yields, according to equation (2.1)

$$r_0 = S_0s_0 + \tilde{M}(S_0)w_2 + \tilde{N}(S_0)w_1. \tag{3.2}$$

Assuming that s_0 is measurable and not correlated to w_1 and w_2, it follows that we can obtain an unbiased estimate for S_0. Denote this estimate as $\hat{S}_1 = N_1M_1^{-1} = \tilde{M}_1^{-1}\tilde{N}_1$. Our new model for the plant now becomes

$$\begin{aligned}G_1 &= (\tilde{M}_0 + \hat{S}_1\tilde{U}_0)^{-1}(\tilde{N}_0 + \hat{S}_1\tilde{V}_0) \\ &= (\tilde{M}_1\tilde{M}_0 + \tilde{N}_1\tilde{U}_0)^{-1}(\tilde{M}_1\tilde{N}_0 + \tilde{N}_1\tilde{V}_0) \\ &= (N_0M_1 + V_0N_1)(M_0M_1 + U_0N_1)^{-1}.\end{aligned}$$

As in the previous section, it makes sense to update the controller K by designing a controller Q_1 for $\hat{S}_1$ such that the closed loop $(Q_1, \hat{S}_1)$ achieves some desired objective. In particular, we require that (Q_1, S_0) is also stable. Given

$Q_1 = U_1 V_1^{-1} = \tilde{V}_1^{-1} \tilde{U}_1$ and the double Bezout equation

$$\begin{bmatrix} M_1 & U_1 \\ N_1 & V_1 \end{bmatrix} \begin{bmatrix} \tilde{V}_1 & -\tilde{U}_1 \\ -\tilde{N}_1 & \tilde{M}_1 \end{bmatrix} = \begin{bmatrix} \tilde{V}_1 & -\tilde{U}_1 \\ -\tilde{N}_1 & \tilde{M}_1 \end{bmatrix} \begin{bmatrix} M_1 & U_1 \\ N_1 & V_1 \end{bmatrix}$$
$$= \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix},$$

we have that for some S_1

$$\begin{aligned} S_0 = \hat{S}_1(S_1) &= (\tilde{M}_1 + S_1 \tilde{U}_1)^{-1} (\tilde{N}_1 + S_1 \tilde{V}_1) \\ &= (N_1 + V_1 S_1)(M_1 + U_1 S_1)^{-1}. \end{aligned} \tag{3.3}$$

Our new controller becomes

$$\begin{aligned} K_1(Q_1) &= (U_0 + M_0 Q_1)(V_0 + N_0 Q_1)^{-1} \\ &= (U_0 V_1 + M_0 U_1)(V_0 V_1 + N_0 U_1)^{-1}. \end{aligned} \tag{3.4}$$

Also the plant model is now

$$\begin{aligned} \bar{G} = G_1(S_1) = {} & [(N_0 M_1 + V_0 N_1) + (N_0 U_1 + V_0 V_1) S_1] \\ & \times [(M_0 M_1 + U_0 N_1) + (U_0 V_1 + M_0 U_1) S_1]^{-1}. \end{aligned} \tag{3.5}$$

If the actual closed loop does not respond as hoped for, we can repeat the above procedure starting from the system model G_1 and controller K_1.

In order to identify S_1, we inject a signal s_1 in the control loop, see Figure 3.2. Again, s_1 is generated independently from w_1 and w_2. We now have

$$r_1 = S_1 s_1 + \tilde{M}(S_1) w_2 + \tilde{N}(S_1) w_1.$$

This allows us to find an estimate for S_1. Denote this estimate as $\hat{S}_2 = N_2 M_2^{-1} = \tilde{M}_2^{-1} \tilde{N}_2$ etc.

In this manner we proceed with the identification step of $\hat{S}_i,\ i = 1, 2, \ldots$ followed by the control design step Q_i. In the control, we are only concerned with the control loop $(\hat{S}_i, Q_i)$. The crucial assumption in order to keep nesting the design step is that at any one step, $(\hat{S}_{i-1}, Q_i)$ is stable. This amounts to stating that we always stabilize the original system. (See Figure 3.3). Clearly, the advantage of this nesting approach over the iteration approach in the previous section is that at any one step in the design, we have a much easier identification task. What is not immediately clear in the present procedure is whether or not we have a potential to lose out by a poor design, say at the ith nested loop. Using the iterated scheme, we can always recover from a poor ith iteration design. The following more formal derivations serve to clarify the situation, and in particular demonstrate that in using nesting, no freedom of design is lost, as long as overall stability is maintained, regardless of intermediate identification and/or control mishaps.

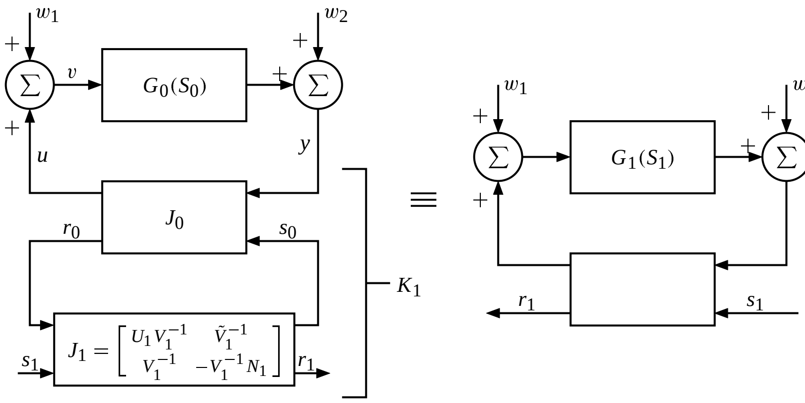

w_1
w_2
$+$
v
$G_0(S_0)$
u
y
J_0
r_0
s_0
$J_1 = \begin{bmatrix} U_1V_1^{-1} & \tilde{V}_1^{-1} \\ V_1^{-1} & -V_1^{-1}N_1 \end{bmatrix}$
s_1
r_1
K_1
$\equiv$
$G_1(S_1)$

FIGURE 3.2. Step 2 in nested design

Equivalent Representation[*]

Consider a sequence of models $G_i \in R_p$ for $i = 0, \ldots, m-1$. Let $K_i \in R_p$ represent a sequence of associated stabilizing controllers, so that $\left(G_i, K_i\right)$ are stabilizing pairs. Let $G_i = N_i M_i^{-1}$ and $K_i = U_i V_i^{-1}$, as usual, with all the transfer functions $N_i, M_i, U_i, V_i \in RH_\infty$ for all $i = 0, \ldots, m-1$, satisfying the double Bezout equation

$$\begin{aligned}\begin{bmatrix} M_i & U_i \\ N_i & V_i \end{bmatrix}\begin{bmatrix} \tilde{V}_i & -\tilde{U}_i \\ -\tilde{N}_i & \tilde{M}_i \end{bmatrix} &= \begin{bmatrix} \tilde{V}_i & -\tilde{U}_i \\ -\tilde{N}_i & \tilde{M}_i \end{bmatrix}\begin{bmatrix} M_i & U_i \\ N_i & V_i \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}.\end{aligned} \tag{3.6}$$

For each $\bar{G} \in R_p$ there exists a unique $S \in R_p$ such that

$$\begin{aligned} G_i(S) &= \left(\tilde{M}_i + G_{i+1}(S)\tilde{U}_i\right)^{-1}\left(\tilde{N}_i + G_{i+1}(S)\tilde{V}_i\right) \\ &= \left(N_i + V_i G_{i+1}(S)\right)\left(M_i + U_i G_{i+1}(S)\right)^{-1}, \end{aligned} \tag{3.7}$$

for $i = 0, \ldots, m-1$ with extremes

$$G_0(S) = \bar{G}, \qquad G_m(S) = S. \tag{3.8}$$

Conversely, each $S \in R_p$ defines via the backward iterations (3.7) a unique $\bar{G} \in R_p$.

It is important to observe that the models G_i may be completely arbitrary. Also neither S nor $\bar{G}$ needs to be stable. This is captured in the following two results.

[*]This material is more technical than that of Chapters 2 and 3. It is not required for subsequent developments. On first reading one need but seek to grasp the key insights without being convinced of the detailed formulations.

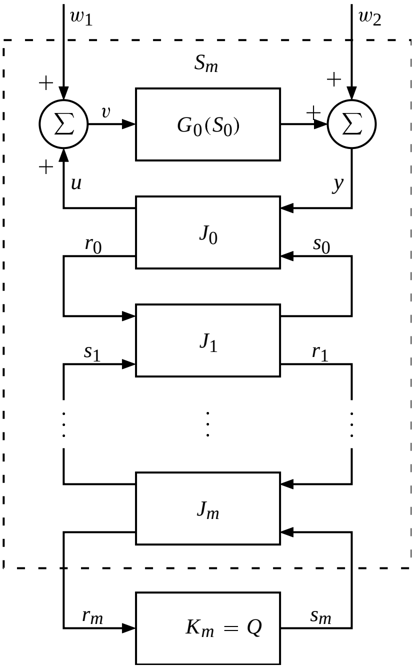


FIGURE 3.3. Step m in nested design

Lemma 3.1. *Suppose we are given a sequence of models $G_i \in R_p$ and strictly proper stabilizing controllers $K_i \in R_p$ for G_i, with (N_i, M_i) and (U_i, V_i) right coprime factorizations of G_i and K_i, respectively, $i = 0, 1, \dots, m-1$. Then any transfer matrix $\bar{G} \in R_p$ can be expressed in the following recursive manner in terms of a unique $G_m(S) = S \in R_p$:*

$$\bar{G} = G_0(S), \qquad G_i(S) = (N_i + V_i G_{i+1}(S))(M_i + U_i G_{i+1}(S))^{-1}, \tag{3.9}$$

for $i = 0, 1, \dots, m-1$. Moreover, each $G_i(S)$ belongs to R_p. Conversely, any given $G_m = S \in R_p$ can recursively yield $G_{m-1}(S), \dots, G_0 = G(S)$ in R_p via (3.9).

Proof. See problems. □

Notice that for any given plant, there always exists a strictly proper stabilizing controller, see Vidyasagar (1985). Hence without loss of generality we can assume that U_i is strictly proper.

For a matrix transfer function $\bar{G} = G(S) \in R_p$ in the recursive form of (3.9), a parameterization of all rational proper controllers for $\bar{G}$ is summarized by the following lemma.

Lemma 3.2. *Given a matrix transfer function $\bar{G} = G(S) \in R_p$ with the recursive representation* (3.9)*, assume that $N_i, M_i, U_i, V_i \in RH_\infty$ satisfy the double Be-*

zout identity of (3.6) *with* N_i, U_i *being strictly proper. Then any rational proper controller for* $\bar{G}$ *can be recursively parameterized by*

$$K(Q) = K_0(Q), \qquad K_i(Q) = (U_i + M_i K_{i+1}(Q))(V_i + N_i K_{i+1}(Q))^{-1}, \tag{3.10}$$

for $i = 0, 1, \ldots, m-1$, *in terms of a unique* $K_m = Q \in R_p$.

The practical importance of these lemmas (Lemma 3.1 and Lemma 3.2) is twofold. First, there is no loss of information or control design freedom in the iterative procedure, since Lemma 3.1 holds for arbitrary G_i, K_i. This provides strong justification for the nested (Q, S) design method. Second, we may expect that the iterative design/identification method explained in the previous subsection, see also Figure 3.2, leads to stepwise improved models and control performance. Each step provides diminishing returns, hence leading to a natural termination of the iterative process, which in principle could indeed be continued *ad infinitum*.

Stability Results†

In this subsection, we extend the robust stabilization results for the nested two-controller case in Chapter 3 to the nested multicontroller case as depicted in Figure 3.3. We present conditions for the multicontroller closed-loop system of Figure 3.4 to be stable. Figure 3.4 is a further generalization of Figure 3.3, allowing us to discuss internal stability more precisely. First, let us consider what well-posedness and internal stability means for the multicontroller case. The multicontroller scheme of Figure 3.4 is *well-posed* and *internally stable* if and only if each matrix transfer function from u_i to e_j exists and belongs to RH_∞ for $i, j = 1, \ldots, 2m+3$. We proceed to find the necessary and sufficient conditions for this.

Let us begin with the known case when $n = 1$. Consider first the arrangements of Figures 3.5 and 3.6. We then have the following mild generalization (to cope with the additional external signals) of the stability results of Section 2.2 and Section 3.4, see also Francis (1987).

Lemma 3.3. *Given a* 2×2*-block matrix transfer function* P *of the form* (2.2.1), *assume that* P *is stabilizable with respect to the controller arrangement of Figure 3.5. Then the matrix transfer function from* u_1, u_2, u_3 *to* e_1, e_2, e_3 *is in* RH_∞ *if and only if* K *stabilizes* P_{22}.

Proof. Directly from the definitions of stability. □

Theorem 3.4. *Consider Figure 3.6 with* P *being stabilizable with respect to the controller arrangement of Figure 3.5. Then the system in Figure 3.6 is stable if and only if* Q *stabilizes* S.

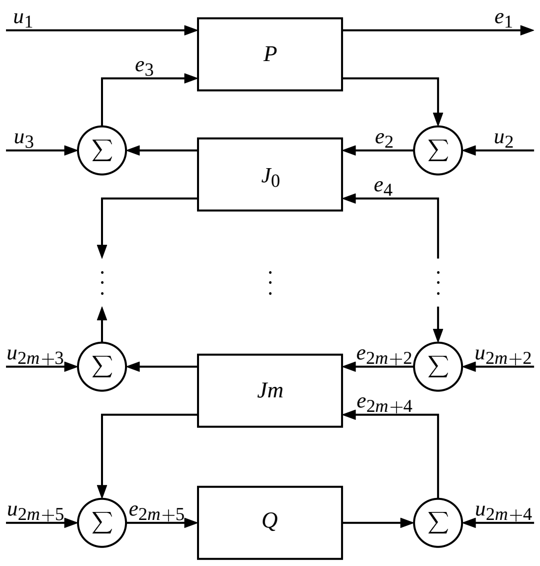


FIGURE 3.4. The class of all stabilizing controllers for P

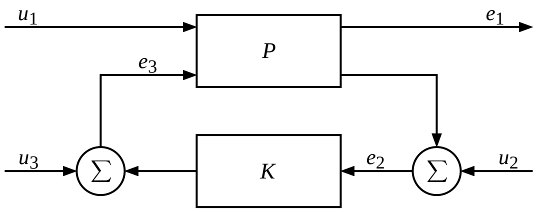


FIGURE 3.5. The class of all stabilizing controllers for $P,\ m = 1$

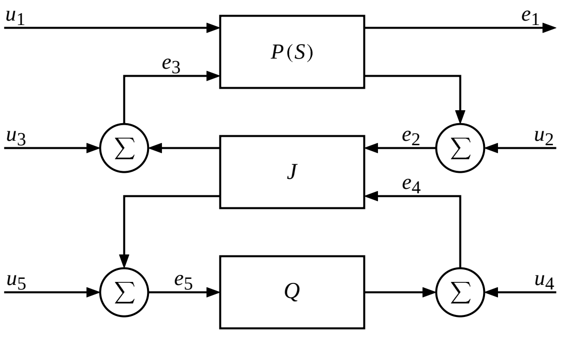


FIGURE 3.6. Robust stabilization of $P,\ m = 1$

Proof. Mild generalizations of results in Section 3.4. The key step is to show that the subblock of Figure 3.6 with inputs $[\,u_1\ u_2\ u_3\ (e_4{-}u_4)\,]'$ and outputs $[\,e_1\ e_2\ e_3\ (e_5{-}u_5)\,]'$ has the form

$$P_1(S) = \left[\begin{array}{ccc|c} P_{11}(S) + P_{12}(S)M(S)\tilde{U}P_{21}(S) & P_{12}(S)M(S)\tilde{U} & P_{12}(S)M(S)\tilde{V} & P_1(S)M(S) \\ V\tilde{M}(S)P_{21} & V\tilde{M}(S) & N(S)\tilde{V} & N(S) \\ U\tilde{M}(S)P_{21} & U\tilde{M}(S) & M(S)\tilde{V} & M(S) \\ \hline \tilde{M}(S)P_{21} & \tilde{M} & \tilde{N} & S \end{array}\right]$$

where

$$P(S) = \begin{bmatrix} P_{11}(S) & P_{12}(S) \\ P_{21}(S) & P_{22}(S) \end{bmatrix}, \qquad P_{22}(S) = G(S),$$

$$G(S) = N(S)M(S)^{-1}, \qquad N(S) = N + VS, \qquad M(S) = M + US.$$

□

With these results as the background, let us now move to the multicontroller case, with first the following lemma, which can be proved by an induction argument.

Lemma 3.5. *The stability properties of the control scheme in Figure 3.4 with P replaced by $P(S)$ are equivalent to those of the control scheme in Figure 3.7 for $i = 1, \ldots, m$. In addition, if P is stabilizable with respect to the controller arrangement of Figure 3.5, then P_i, $i = 2, \ldots, m$ are also stabilizable.*

Proof. See problems for an inductive proof. Start with proof results of Theorem 3.4. □

From this lemma, we can immediately see the following result.

Corollary 3.6. *Consider the diagram in Figure 3.4. Then the following relations hold:*

$$e_{2i+2} = P_{i,22}(S)e_{2i+3} + P_{i,21}\begin{bmatrix} u_1 \\ \vdots \\ u_{2i+1} \end{bmatrix} + u_{2i+2}, \qquad i = 1, \ldots. \tag{3.11}$$

An important identification implication of the relations (3.11) is that $P_{i,22}(S) = \hat{S}_i(S)$, $i = 0, 1, \ldots, m$, can be identified successively from measurements. More specifically, once the estimate $\hat{S}_i = N_i M_i^{-1}$ of $\hat{S}_i(S)$ has been obtained through identification and a corresponding augmented controller J_i together with Q has been constructed, then $\hat{S}_{i+1}(S)$, which is a frequency-shaped difference between

[†]This material is more technical than that of Chapters 2 and 3. It is not required for subsequent developments. On first reading one need but seek to grasp the key insights without being convinced of the detailed formulations.

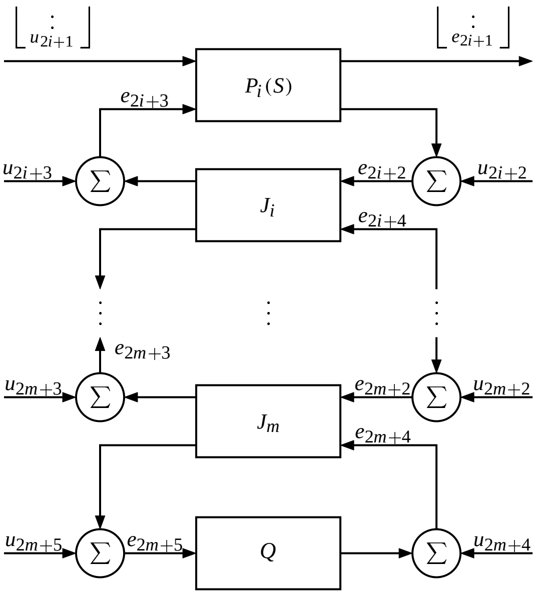


FIGURE 3.7. The $(m - i + 2)$-loop control diagram

$\hat{S}_i(S)$ and $\hat{S}_i$, can be further identified on line from measurement. Notice that at every stage of the nested design an "open-loop" identification problem has to be solved. This is a clear advantage of the nested design over the iterated design of the previous section.

Theorem 3.7. *Assume that P is stabilizable with respect to the controller arrangement of Figure 3.5. Then the multiple control system shown in Figure 3.4 with P replaced by $P(S)$, see also Figure 3.7, is internally stable if and only if Q stabilizes S.*

Proof. By Lemma 3.5, the multiple control diagram is equivalent to the two-loop control diagram depicted in Figure 3.7 specialized to the case $i = m$, $P_{m,22} = \hat{S}_{m-1}(S)$ and P_m is stabilizable with respect to the controller arrangement of Figure 3.5. Thus, applying Theorem 3.4 to this two-loop diagram immediately yields Theorem 3.7. $\square$

In the special case where $\hat{S}_m(S) = S = 0$, Theorem 3.7 tells us that the nested multiple control system of Figure 3.4 is stable provided $Q \in RH_\infty$. This implies that the multiple control system is always stable with any given $Q \in RH_\infty$ whenever the mth frequency-shaped plant model error S is sufficiently 'small'. It is also obvious that if Q is both stable and stabilizes S, that is Q strongly stabilizes S, then the multiple control system is simultaneously stable for $\hat{S}_m(S) = 0$ and $\hat{S}_m(S) = S$.

Lemma 3.8. *Consider the* $(m+1)$*-controller strategy of Figure 3.4, see also Figure 3.7 with* $u_i = 0,\ i = 1, \dots, 2m+3$. *Assume that* $\hat{S}_i(0),\ i = 0, \dots, m-1$, *are strictly proper. Then the matrix transfer function from* e_2 *to* e_3 *equals* $K(Q)$ *given as in* (3.10) *with* $K_m = Q$.

Proof. First, the following relations are immediate from the controller strategy of Figure 3.4.

$$\begin{bmatrix} e_{2i+3} \\ e_{2i+5} \end{bmatrix} = \begin{bmatrix} U_i V_i^{-1} & \tilde{V}_i^{-1} \\ V_i^{-1} & -\tilde{N}_i \tilde{V}_i^{-1} \end{bmatrix} \begin{bmatrix} e_{2i+2} \\ e_{2i+4} \end{bmatrix}, \tag{3.12}$$

$$e_{2m+2} = Q e_{2m+5}, \qquad i = 1, \dots, m. \tag{3.13}$$

Let T_i denote the matrix transfer function from e_{2i+4} to e_{2i+5}. Then it is straightforward to check that (3.12) implies

$$e_{2i+3} = (U_i + M_i T_i)(V_i + N_i T_i)^{-1} e_{2i+2}. \tag{3.14}$$

Consequently,

$$T_{i-1} = (U_i + M_i T_i)(V_i + N_i T_i)^{-1}. \tag{3.15}$$

Noting that (3.13) gives $T_m = Q$, one can conclude that $T_0 = K_0$ with $K_m = Q$. This proves the lemma. □

Corollary 3.9. *Suppose* N_i *is strictly proper for* $i = 0, 1, \dots, m-1$. *Then the* $(m+1)$*-controller scheme of Figure 3.4 is internally stable if and only if the matrix transfer function from* u_1, u_2, u_3 *to* e_1, e_2, e_3 *belongs to* RH_∞.

Proof. We only need to prove sufficiency. Assume that the matrix transfer function from u_1, u_2, u_3 to e_1, e_2, e_3 belongs to RH_∞. Quite evidently, this assumption implies the stabilizability of $P(S)$ with respect to the controller arrangement of Figure 3.5. From Lemmas 3.8 and 3.3, it follows that K stabilizes G where K is defined as in (3.10) with $K_n = Q$. But this is equivalent to Q stabilizing S by Lemma 3.2; hence the internal stability of the $(m+1)$-controller strategy follows directly from Theorem 3.7. □

Main Points of Section

This section presents a nested controller structure and a dual nested plant representation. The nested plant representation can be viewed as successive approximations to the plant model, and the nested controller as a successive approximation to the 'optimal' controller for the actual plant.

The key stability result is a generalization of the results of Chapter 3 for the two controller/plant nested representations. Namely the stability of the nested scheme depends on the stability of that of the successive designs, based on the successive approximations to the plant.

The nested approach to controller design appears to be a very natural way to improve on previous designs.

5.4 Notes and References

The material for this chapter has arisen from the authors earlier works recorded in Tay et al. (1989), Yan and Moore (1992), and Yan and Moore (1994). It makes connection with the concepts of iterated design as developed in Zang et al. (1991). The identification method proposed in Section 5.2 is due to Hansen (1989). It is often referred to as identification in dual-Youla parameterization format. The method has been extensively used in the context of identification for control (Lee, 1994; Partanen, 1995; Schrama, 1992b).

Problems

1. Verify Theorem 2.2 using the factorizations (2.4.18), (2.4.19) and also for (2.4.22), (2.4.23).

2. Verify the lemmas and theorems of the chapter.

CHAPTER **6**

Direct Adaptive-Q Control

6.1 Introduction

It should by now be evident that the parameterization of all stabilizing controllers via a stable matrix transfer function Q is an interesting and powerful control design vehicle. In the previous two chapters we indicated how one could optimize Q over a number of different control performance objectives, either in a purely off-line or iterative identification and control design approach.

In the off-line method, we start from a given plant model, and our attention is focussed on rejecting disturbances or maximizing robustness with respect to unmodeled dynamics. In the iterative method, our premise at the outset is that the plant model is inadequate and may prevent us from obtaining the desired performance. In this situation, model identification via the dual S parameterization of all plants stabilized by a given controller, and control via a 'plug-in' controller Q is the natural way to proceed.

In the previous chapter we have considered iterative or nested methods, alternating identification and control design steps. We demonstrated that, when care is exercised, these iterative methods are capable of achieving any desired control objective.

In this chapter, we introduce the first of two adaptive methods. Here, we discuss direct adaptive-Q control. In this approach, Q is adjusted without identification of S. This method is suited for the situation where the signal model uncertainty is limited. For example, the plant model itself could be adequate but the model for the external signals is not, as would be the case when the control objective is to track an unknown signal or a signal with uncertain time-varying spectral content. Another application of direct adaptive-Q control is for retuning an existing controller, either because the original (stabilizing) controller is inefficient to obtain good performance or because the design criterion is altered. In this chapter we limit ourselves to optimization criteria based on rms signal measures, but in principle, the methodology is applicable to any optimization based control design. Whenever control performance is inadequate due to severe model-plant mismatch, direct adaptive-Q design is unlikely to be sufficiently powerful, as is demonstrated. The analysis of the adaptive-Q method uses averaging ideas. The main concepts of averaging as they are required in the analysis are collected in Appendix C.

Unlike traditional adaptive control methods, we start here always from the premise that we know how to stabilize the plant, but may not know how to achieve the desired control performance. Adaptation is seen here as a mechanism to improve performance on-line. The analysis we present is aimed at this objective. The fact that we use slow adaptation is not a real restriction for our purposes. Indeed, adaptive performance enhancement requires adaptation to be slow, at least slow as compared to the normal control dynamics of the closed loop.

The chapter is organized as follows. First we introduce the direct adaptive-Q control algorithm. The algorithm is completely developed in state space notation. This facilitates the analysis. Indeed, the adaptive-Q control leads to a nonlinear and time-varying control system. Despite this observation, and due to the time scale separation properties, frequency domain ideas play an important role in understanding the behavior of the adaptive system. The direct adaptive-Q method is first analyzed under the premise of a perfect plant model. It is then established that *optimal* control is achieved. Next we analyze how the adaptive mechanism breaks down under (severe) model-plant mismatch, but with a graceful degradation. The generic scheme is discussed. The chapter ends with the discussion of a scalar example.

6.2 Q-Augmented Controller Structure: Ideal Model Case

The plant signal model is represented as:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Bw_{1,k}; \qquad x_0, \\ y_k &= Cx_k + Du_k + w_{2,k}, \end{aligned} \tag{2.1}$$

where $x_k \in \mathsf{R}^n$ is the state vector, $u_k \in \mathsf{R}^p$ is the input, $y_k \in \mathsf{R}^m$ is the output, $w_{1,k}$ is an input disturbance and $w_{2,k}$ is an output reference signal. The only signals available for control purposes are the input u_k and the output y_k. Neither the input disturbance $w_{1,k}$ nor the reference signal $w_{2,k}$ are available. The control objective is to obtain an internally stable closed-loop system with output regulation, that is regulating y_k to zero. Otherwise interpreted, we want to have the actual system response $(Cx_k + Du_k)$ track the negative of the reference signal $w_{2,k}$ as closely as possible.

Let us work with a controller structure based on a state estimate feedback arrangement as depicted in Figure 2.5.4. The controller structure is based on a full state observer. (See Chapter 2 for the development of the state space realization):

$$\begin{aligned} \hat{x}_{k+1} &= A\hat{x}_k + Bu_k - Hr_k; \qquad \hat{x}_0, \\ r_k &= (y_k - Du_k) - C\hat{x}_k, \\ u_k &= F\hat{x}_k + s_k. \end{aligned} \tag{2.2}$$

Here $\hat{x}_k$ is the estimate for the state x_k, the estimation residual is r_k and s_k is the extra input to be generated via the Q filter. The nominal controller (with $Q = 0$) provides stability but may not necessarily attain satisfactory disturbance rejection. A nominal controller based solely on the plant model, not taking into account the disturbances, could hardly be expected to achieve optimum disturbance rejection. The Q filter is introduced to enhance disturbance rejection. In order to obtain an implementable adaptively updated Q, we restrict the dynamic complexity of Q as follows:

$$\begin{aligned} z_{k+1} &= A_q z_k + B_q r_k; \qquad z_0, \\ s_k &= \Theta_k z_k. \end{aligned} \tag{2.3}$$

Here $z_k \in \mathsf{R}^{n_q}$ is the state of the adaptive Q filter and A_q is a stable matrix, chosen by the designer. Typically, one would select the eigenvalues of the matrix A_q to be comparable in magnitude with the eigenvalues of the matrix $A + HC$, which determines the observer dynamics. The output equation is to be adaptively adjusted. In this equation, Θ_k is the adaptation parameter being a $p \times n_q$ matrix of adjustable parameters.

In order to focus the development, we restrict the desired signal used to formulate the control objectives as $e_k = \left[\begin{smallmatrix} y_k \\ u_k \end{smallmatrix}\right]$. Our control objective is to achieve fast regulation with disturbance rejection, while containing the control effort.

As pointed out in (2.5.18), the transfer function T_Θ from the disturbance signal $w_k := \left[\begin{smallmatrix} w_{1,k} \\ w_{2,k} \end{smallmatrix}\right]$ to e_k is affine in the transfer function Q. It follows that for $\Theta_k = \Theta$ (no adaptation), this transfer function is affine in Θ. This can also be directly observed from the following state space realization for the mapping from w_k to $e_k = \left[\begin{smallmatrix} y_k \\ u_k \end{smallmatrix}\right]$:

$$\left[\begin{array}{c} x_{k+1} \\ z_{k+1} \\ \tilde{x}_{k+1} \\ \hline y_k \\ u_k \end{array}\right] = \left[\begin{array}{ccc|cc} A+BF & B\Theta_k & -BF & B & 0 \\ 0 & A_q & -B_qC & 0 & -B_q \\ 0 & 0 & A+HC & B & H \\ \hline C+DF & D\Theta_k & -DF & 0 & I \\ F & \Theta_k & -F & 0 & 0 \end{array}\right] \left[\begin{array}{c} x_k \\ z_k \\ \tilde{x}_k \\ \hline w_{1,k} \\ w_{2,k} \end{array}\right]. \tag{2.4}$$

Here $X_k = [\, x_k' \; z_k' \; \tilde{x}_k' \,]'$ is a closed-loop system state variable and the state estimation error for the plant is $\tilde{x}_k = x_k - \hat{x}_k$. It follows from the state space representation (2.4) that the closed-loop dynamics are in essence determined by the block designed elements $A + BF$, A_q, $A + HC$, so that the adaptation parameter Θ_k can not seriously affect the stability of the closed loop. Indeed, BIBO stability of the nominal control design, that is (2.4) with $\Theta_k \equiv 0$, informs us that for any bounded sequence Θ_k and any bounded disturbance signal w_k, the state X_k of the closed-loop equation (2.4) is bounded. More formally, we can state:

Lemma 2.1. *Consider the system* (2.4). *Let* $|w_k| \le W$ *and* $|\Theta_k| \le \theta$ *for all* k. *Assume that the matrices* $A + BF$, $A + HC$ *and* A_q *are stable in that:*

$$\max |\mathrm{eig}(A + BF)| < \lambda_1 < 1, \tag{2.5}$$

$$\max(|\mathrm{eig}(A + HC)|, |\mathrm{eig}(A_q)|) < \lambda_0 < \lambda_1 < 1. \tag{2.6}$$

Then there exist positive constants $C_0, C_1 \ge 1$ *such that:*

$$(|\tilde{x}_k|, |z_k|) \le C_0 \lambda_0^k (|\tilde{x}_0|, |z_0|) + C_0(1 - \lambda_0^k) W, \tag{2.7}$$

and

$$|x_k| \le C_1 \lambda_1^k (|x_0| + |\tilde{x}_0| + \theta |z_0|) + C_1(1 + \theta)(1 - \lambda_1^k) W. \tag{2.8}$$

Proof. Follows directly from the *variation of constants formula* applied to (2.4). □

Before continuing with the introduction of the adaptive algorithm, it is instructive to see how the plug in controller achieves tracking. Let us consider the case in which all signals y, u, w_1 and w_2 are scalar valued. From the system equation (2.4), it is clear that we obtain a transfer function from w_1, w_2 to y of the form:

$$y = \frac{L_1(z) + L_\Theta(z) L_2(z)}{P_C(z)} w_1 + \frac{B_1(z) + B_\Theta(z) B_2(z)}{P_C(z)} w_2. \tag{2.9}$$

Here L_1, L_Θ, L_2, P_C, B_1, B_Θ, B_2 are polynomials. The polynomial $P_C(z)$ has as its roots the eigenvalues of the matrices $A+BF$, A_q and $A+HC$. The polynomials $L_\Theta(z)$ and $B_\Theta(z)$ have coefficients which are linear functions of the Θ parameters. From (2.9), but also from (2.4), we observe that the poles of the system are completely unaffected by the plug in controller Θ. However, by appropriate selection of Θ, we can minimize $\|e\|_{\mathrm{rms}}$. In particular, if w_1, w_2 have a finite spectrum containing say n_w different frequency lines (counting negative frequency lines!) then exact disturbance rejection and tracking can be achieved if Θ contains at least n_w free parameters which place appropriate zeros in the transfer function. Hence we are discussing a notch filter structure, which leads us to an adaptive notch filter.

6.3 Adaptive-Q Algorithm

In order to highlight the principle underlying Q filter adaptation, we restrict ourselves to the simplest of algorithms. Lemma 2.1 in Section 6.2 greatly simplifies our analysis. Here we are in the pleasant situation where stability is guaranteed *a priori* and so our focus can be on performance issues within an adaptive context, rather than on closed-loop stability normally of foremost concern in the analysis of adaptive systems. After all, performance enhancement is exactly the main motivation for using an adaptive algorithm and the less concern there is on stability issues the better.

Let us adjust Θ_k as to minimize the criterion:

$$J(\Theta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} e_k' R e_k; \qquad R = R' \geq 0. \tag{3.1}$$

An approximate *steepest descent* algorithm may be used to update Θ_k in a recursive manner as follows:

$$\Theta_{ij,k+1} = \Theta_{ij,k} - \mu \left.\frac{\partial e_k'}{\partial \Theta_{ij}}\right|_{\Theta_k} R e_k,$$

or in short hand:

$$\Theta_{ij,k+1} = \Theta_{ij,k} - \mu \gamma_{ij,k}' R e_k; \qquad i = 1 \dots p, \qquad j = 1 \dots n_q. \tag{3.2}$$

Here $\Theta_{ij,k}$ is the ijth entry in the matrix Θ_k, and $\gamma_{ij,k}$ is an $m + p$ column vector of sensitivity functions, obtained from:

$$\left[\begin{array}{c} \Gamma_{ij,k+1} \\ \hline \gamma_{ij,k} \end{array}\right] = \left[\begin{array}{c|c} A + BF & BE_{ij} \\ \hline C + DF & DE_{ij} \\ F & E_{ij} \end{array}\right] \left[\begin{array}{c} \Gamma_{ij,k} \\ \hline z_k \end{array}\right]; \qquad \Gamma_{ij,0} = 0, \tag{3.3}$$

where E_{ij} is a matrix of zero elements except for a unity at the ijth position. Notice that the gradient vector $\gamma_{ij,k}$ is indeed independent of $\Theta_{ij,k}$, thus confirming the earlier observation that the transfer function from w_k to e_k is affine in Θ.

In the update algorithm (3.2), the design parameter μ is a small positive constant which scales the adaptation speed. The equations (3.2) and (3.3) describe the "adaptive" mechanism. *A priori*, (3.2) does not guarantee that Θ_k will be bounded. Although, it turns out that in the ideal case, the algorithm does have this property. Boundedness may be guaranteed by either projecting Θ_k back into a bounded set or by introducing some *leakage* in the update equation (3.2). *Projection* onto a ball (with center 0 and radius θ denoted as $B(0, \theta)$) leads to an algorithm of the form:

$$\begin{aligned} \Theta_{ij,k+1}^* &= \Theta_{ij,k} - \mu \gamma_{ij,k}' R e_k, \\ \Theta_{k+1} &= \Theta_{k+1}^* \quad \text{if } \Theta_{k+1}^* \in B(0, \theta), \\ &= \Theta_{k+1}^* \frac{\theta}{\left\| \Theta_{k+1}^* \right\|} \quad \text{otherwise.} \end{aligned} \tag{3.4}$$

Leakage may be implemented as:

$$\Theta_{ij,k+1} = (1-\mu\lambda)\Theta_{ij,k} - \mu\gamma'_{ij,k} Re_k, \tag{3.5}$$

where $\lambda \in (0,1)$ is the leakage factor. Leakage contracts Θ_k towards zero, that is, it prefers an unmodified controller design. It will be shown that in the presence of *sufficiently rich signals* neither of these modifications is required, at least in the ideal case. In general it is good practice to implement either one.

6.4 Analysis of the Adaptive-Q Algorithm: Ideal Case

In order to obtain some insight into the behavior of the adaptive algorithm, we analyze the closed-loop system described by (2.4), (3.2) and (3.3), not considering the projection or the leakage modification.

Also, in order that the "criterion minimization" task attempted by the adaptive algorithm be well posed we assume that the disturbance signal w_k is stationary. More precisely, we assume that:

Assumption 4.1. *Stationary signals: w_k is a bounded signal such that there exist constant matrices E_w, $C_w(\ell)$, $\ell = 0, 1, \ldots$ such that for all integers $N \geq 1$*

$$\begin{aligned} \left| \sum_{k=k_0}^{k_0+N-1} (w_k - E_w) \right| &\leq \square_2 N^{\gamma}, \\ \left| \sum_{k=k_0}^{k_0+N-1} (w_k w'_{k-\ell} - C_w(\ell)) \right| &\leq \square_3 N^{\gamma}, \qquad \ell = 0, 1, \ldots, \end{aligned} \tag{4.1}$$

for some positive constants $\square_2, \square_3$ independent of k_0 and $\gamma \in (0,1)$.

From the condition (4.1) it follows that the criterion (3.1) is well defined for any fixed Θ. In particular as N goes to infinity the *Cesáro mean* in (3.1) converges to $J(\Theta)$ at the same rate as $N^{-1+\gamma}$ converges to 0.

We also assume that the disturbance is sufficiently rich. In particular we assume that:

Assumption 4.2. *Exciting signals: w_k is such that for any stable, rational, matrix transfer function $\square \in RH_{\infty}^{(m+p)}$ the signal $w_f = \square w$ satisfies:*

$$\left| \sum_{k=k_0}^{k_0+N-1} (w_{f,k} w'_{f,k} - C_{w_f}) \right| \leq \square_4 N^{\gamma}, \tag{4.2}$$

for some $\square_4 > 0$, and some symmetric positive definite C_{w_f}, and $\gamma \in (0,1)$.

Under Assumption 4.2, there exists a unique Θ^* such that $J(\Theta^*) \leq J(\Theta)$ for all Θ, indeed $J(\Theta)$ is quadratic in Θ with positive definite Hessian.

Remark. Assumption 4.2 not only guarantees the existence of a unique Θ^* minimizer of $J(\Theta)$ but unfortunately, also excludes the possibility of exact tracking. Indeed, Assumption 4.2 implies *inter alia* that the spectral content of w is not finite.

Under Assumption 4.1 and provided μ is sufficiently small we can use averaging results to analyze the behavior of the adaptive system. For an overview of the required results from averaging theory, see Appendix C. The following result is established.

Theorem 4.3. *Consider the adaptive system described by* (2.4), (3.2) *and* (3.3). *Let $A + BF$, $A + HC$, A_q be stable matrices in that* (2.5) *and* (2.6) *are satisfied. Let the disturbance signal $w_k = \begin{bmatrix} w_{1k} \\ w_{2k} \end{bmatrix}$ satisfy Assumptions 4.1 and 4.2, and $\|w_k\| \leq W$. Then there exists a positive μ^* such that for all $\mu \in (0, \mu^*)$ the adaptive system has the properties:*

1. *The system state is bounded; for all initial conditions x_0, $\tilde{x}_0$, z_0, Θ_0:*

$$\begin{aligned} \|\Theta_k\| &\leq \theta \quad \textit{for some } \theta > 0, \\ (|\tilde{x}_k|, |z_k|) &\leq C_0\lambda_0^k(|\tilde{x}_0|, |z_0|) + C_0(1-\lambda_0^k)W, \\ |x_k| &\leq C_1\lambda_1^k(|x_0| + |\tilde{x}_0| + |z_0|\theta) + C_1(1+\theta)(1-\lambda_1^k)W, \end{aligned}$$

   for all $k = 0, 1, \ldots$ for some constants C_1, C_0 independent of μ, W, θ. See also (2.7) *and* (2.8).

2. *Near optimality:*

$$\limsup_{k\to\infty} \left\|\Theta_k - \Theta^*\right\| \leq C_5\mu^{(1-\gamma)/2}, \tag{4.3}$$

   where Θ^* is the unique minimizer of the criterion (3.1). *The constant C_5 is independent of μ.*

3. *Convergence is exponential:*

$$\left\|\Theta_k - \Theta^*\right\| \leq C_6(1 - L\mu)^k \left\|\Theta_0 - \Theta^*\right\| \quad \textit{for all } k = 0, 1, \ldots, \tag{4.4}$$

   for some $C_6 > 1$ and $L > 0$ independent of μ.

Proof[*]**.** Define $X_k(\Theta)$ as the state solution of the system (2.4), denoted $X_k(\Theta_k)$, with $\Theta_k \equiv \Theta$. In a similar manner define $e_k(\Theta)$ as its output. Such correspond to the so-called frozen system state and output. These are zero adaptation approximations for the case when $\Theta_k \approx \Theta$. Because Θ_k is slowly time varying and because of the stability of the matrices $A + BF$, A_q and $A + HC$ it follows that for sufficiently small μ, provided $|\Theta_k| \leq \theta$,

$$\begin{aligned} |X_k - X_k(\Theta_k)| &\leq C_7\mu; &&\text{some } C_7 > 0, &&\text{for all } k : |\Theta_k| \leq \theta, \\ |e_k - e_k(\Theta_k)| &\leq C_7\mu; &&\text{some } C_7 > 0, &&\text{for all } k : |\Theta_k| \leq \theta. \end{aligned}$$

[*]This proof may be omitted on first reading.

Now, (3.2) may be written as:

$$\Theta_{ij,k+1} = \Theta_{ij,k} - \mu\gamma'_{ij,k} Re_k(\Theta_k) + O(\mu^2),$$

which is at least valid on a time interval $k \in (0, M/\mu)$, for some $M > 0$.

The averaged equation becomes:

$$\Theta^{av}_{ij,k+1} = \Theta^{av}_{ij,k} - \mu \left.\frac{\partial J(\Theta)}{\partial \Theta_{ij}}\right|_{\Theta=\Theta^{av}_k}.$$

Here we observe that for all finite Θ:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=k_0}^{k_0+N-1} \gamma'_{ij,k} Re_k(\Theta) = \frac{\partial J(\Theta)}{\partial \Theta_{ij}}.$$

This follows from Assumption 4.1. Provided μ is sufficiently small, it follows that Θ^{av}_k is bounded and converges to Θ^*. Indeed, the averaged equation is a steepest descent algorithm for the cost function $J(\Theta)$. In view of Assumption 4.2, $J(\Theta)$ has a unique minimum, which is therefore a stable and attractive equilibrium, see Hirsch and Smale (1974).

The result then follows from the standard averaging theorems presented in Appendix C, in particular, Theorem C.4.2. □

Remarks.

1. The same result can be derived under the weaker signal assumption:

   Assumption 4.4. *The external signal w is such that there exists a unique minimizer Θ^* for the criterion* (3.1).

   Assumption 4.4 allows for situations where exact output tracking can be achieved.

2. Generally, the adaptive algorithm achieves near optimal performance in an exponential manner. Of course, the convergence is according to a large time constant, as μ is small.

3. The real advantage of the adaptive algorithm is its ability to track near optimal performance in the case w_k is not stationary. Indeed, we can infer from Theorem 4.3 that provided the signal w_k is well approximated by a stationary signal over a time horizon of the order of $1/\mu$, the adaptive algorithm will maintain near optimal performance regardless of the time-varying characteristics of the signal.

If we consider the adaptive algorithm with leakage, we may establish a result which no longer requires sufficiently rich external signals. In this situation, there is not necessarily a unique minimizer for the criterion (3.1). The following result holds:

Theorem 4.5. *Consider the adaptive system described by* (2.4), (3.3) *and* (3.5). *Let $A + BF$, $A + HC$ and A_q be stable matrices satisfying the conditions* (2.5) *and* (2.6). *Let the external signal be stationary, satisfying Assumption 4.1, and $\|w_k\| \leq W$. Then there exists a positive μ^* such that for all $\mu \in (0, \mu^*)$ the adaptive system has the properties*

1. *The system state is bounded for all possible initial conditions x_0, $\tilde{x}_0$, z_0, Θ_0:*

$$\begin{aligned} \|\Theta_k\| &\leq \theta, \quad \text{for some } \theta > 0, \\ (|\tilde{x}_k|, |z_k|) &\leq C_0\lambda_0^k(|\tilde{x}_0|, |z_0|) + C_0(1 - \lambda_0^k)W, \\ |x_k| &\leq C_1\lambda_1^k(|x_0| + |\tilde{x}_0| + |z_0|\,\theta) + C_1(1 + \theta)(1 - \lambda_1^k)W, \end{aligned}$$

for all $k = 0, 1, \ldots$ and constants $C_0,\ C_1 > 0$ independent of μ, W and θ.

2. *Writing $\partial J/\partial\Theta = 0$ in the form $\Gamma \operatorname{vec}\Theta - E = 0$[†]; then there exists a Θ^*, $\operatorname{vec}\Theta^* = (\Gamma + \lambda I)^{-1}E$ such that for some C_2 independent of μ*

$$\limsup_{k\to\infty} \left\|\Theta_k - \Theta^*\right\| \leq C_2\mu^{(1-\gamma)/2}.$$

3. *Convergence is exponential (for some $C_3 > 0,\ L > 0$):*

$$\left\|\Theta_k - \Theta^*\right\| \leq C_3(1 - L\mu)^k \left\|\Theta_0 - \Theta^*\right\|; \qquad k = 0, 1, \ldots.$$

Proof. The proof follows along the same lines as the proof of Theorem 4.3. It suffices to observe that the averaged version of equation (3.4) governing the update for the estimate Θ_k becomes:

$$\Theta^{av}_{ij,k+1} = \Theta^{av}_{ij,k} - \mu\left(\left.\frac{\partial J(\Theta)}{\partial\Theta_{ij}}\right|_{\Theta=\Theta^{av}_k} + \lambda\Theta^{av}_{ij,k}\right). \tag{4.5}$$

The existence of the averages is guaranteed by Assumption 4.1. It follows that there exists a unique equilibrium for (4.5) given by Θ^*. Because $\Gamma = \Gamma' \geq 0$, and $\Gamma + \lambda I > 0$ for all $\lambda > 0$, then Θ^* is a locally exponentially stable solution of (4.5), that is for sufficiently small μ, such that $|\text{eig}(I - \mu(\Gamma + \lambda I))| < 1$. This establishes the result upon invoking Theorem C.4.2. □

Remarks.

1. The result of Theorem 4.5 establishes the rationale for having the exponential forgetting factor $(1 - \mu\lambda)$ in (3.5) satisfying $0 < 1 - \mu\lambda < 1$. The *exponential forgetting* of past observations should not dominate the adaptive mechanism, otherwise the only benefit which can be derived from the adaptation will be due to the most recent measurements.

[†]vec Θ denotes the column vector obtained by collecting all columns of the matrix Θ from left to right and stacking them under one another.

2. The minimizers of the criterion (3.1) are of course the solutions of $\Gamma \operatorname{vec} \Theta - E = 0$. In the case of Γ being only positive semi-definite and not positive definite, there is a stable linear subspace of Θ parameter space, achieving optimal performance. In this case the exponential forgetting is essential to maintain bounded Θ. Without it, the adaptation mechanism will force Θ_k towards this linear subspace of minimizers, and subsequently Θ_k will wander aimlessly in this subspace. Under such circumstances, there are combinations of w signals that will cause $\|\Theta_k\|$ to become unbounded. The forgetting factor λ prohibits this.

3. The forgetting factor guarantees boundedness, but with the cost of not achieving optimality. Indeed, Θ^* (as established in Theorem 4.5 Item 2) solves $(\Gamma + \lambda I) \operatorname{vec} \Theta = E$, not $\Gamma \operatorname{vec} \Theta = E$. The penalty for this is however a small one. If Γ were invertible, then for sufficiently small λ,

$$\operatorname{vec} \Theta^* = \Gamma^{-1} E - \lambda \Gamma^{-2} E + \lambda^2 \Gamma^{-3} E + \cdots$$

Hence there is but an order λ error between the obtained Θ^* and the optimal one, $\Gamma^{-1} E$. More generally, Γ and E may be expressed as $\Gamma = \sum_i^k \Gamma_i \Gamma_i'$ and $E = \sum_i^k \Gamma_i a_i$, with $\Gamma_i' \Gamma_i = 1$, $i = 1, \ldots, k$ and $\Gamma_i' \Gamma_j = 0$ for $i \neq j$, $i, j = 1, \ldots, k$. The Γ_i may be interpreted as those directions in parameter space in which information is obtained from the external signal w. If $k < \dim(\operatorname{vec} \Theta)$, that is Γ is singular, then we have that

$$\begin{aligned} \Lambda_j \operatorname{vec} \Theta^* &= 0; & j &= k+1, \ldots, \dim(\operatorname{vec} \Theta), \\ \Gamma_i' \operatorname{vec} \Theta^* &= \frac{a_i}{1+\lambda}; & i &= 1, \ldots, k, \end{aligned}$$

where the Λ_j, $j = k+1, \ldots, \dim(\operatorname{vec} \Theta)$ complement the Γ_i, $i = 1, \ldots, k$ to form an orthonormal basis for the parameter space. Again we see that near optimal performance is obtained.

6.5 Q-augmented Controller Structure: Plant-model Mismatch

In Sections 6.2, 6.3 and 6.4, the ideal situation where the plant is precisely modeled has been discussed. Normally, we expect the model to be an approximation. Let the controller be as defined in (2.2) and (2.3). The nominal controller corresponds to $\Theta = 0$. As usual, we assume that the nominal controller stabilizes the actual plant. Using the representations developed in Chapters 2 and 3 the plant is here represented by:

$$\left[\begin{array}{c} x_{k+1} \\ v_{k+1} \\ \hline y_k \end{array}\right] = \left[\begin{array}{cc|ccc} A & -HC_s & B & B & 0 \\ -B_s F & A_s & B_s & B_s & 0 \\ \hline C & C_s & D & 0 & I \end{array}\right] \left[\begin{array}{c} x_k \\ v_k \\ \hline u_k \\ w_{1k} \\ w_{2k} \end{array}\right], \tag{5.1}$$

where

$$G : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

is the realization for the model (see also (2.1)), and

$$S : \left[\begin{array}{c|c} A_s & B_s \\ \hline C_s & 0 \end{array}\right] \tag{5.2}$$

represents a stable system characterizing the plant model mismatch. The vector $v_k \in \mathrm{R}^s$ is the state associated with the unmodeled dynamics. The matrices H and F are respectively the gain matrices used to stabilize the observer and the nominal system model. The representation (5.1) is derived from the earlier result (3.5.1), with the assumption $D_s = 0$ and using a nominal stabilizing controller in observer form, that is, with the Z of (3.5.4) having the form

$$Z := \begin{bmatrix} M & U \\ N & V \end{bmatrix} := \left[\begin{array}{c|cc} A + BF & B & -H \\ \hline F & I & 0 \\ C + DF & D & I \end{array}\right]. \tag{5.3}$$

From the results of Chapter 3, we recall that the controller (2.2) with $s_k \equiv 0$, that is, the nominal controller stabilizes any system of the form (5.1) as long as A_s is a stable matrix. It is also clear that any system of the form (5.1) with A_s stable and $C_s = 0$ is stabilized by the controller (2.2) with the stable Q-filter (2.3). More importantly it is established in Theorem 3.4.2 that the controlled system described by equation (5.1), (2.2) and (2.3) with $\Theta_k \equiv \Theta$ is stable if and only if the matrix

$$\begin{bmatrix} A_s & B_s\Theta \\ B_q C_s & A_q \end{bmatrix} \tag{5.4}$$

is a stable matrix. It has all its eigenvalues in the open unit disk, since this condition is equivalent to requiring that the pair (Q, S) is stabilizing. It is clear that some prior knowledge about the 'unmodeled dynamics' S is required in order to be able to guarantee that (Q, S) is a stabilizing pair. (Also recall that this result does not require A_s to be stable!)

The closed loop, apart from the adaptation algorithm, may be represented in state space format as follows:

$$\begin{bmatrix} x_{k+1} \\ v_{k+1} \\ z_{k+1} \\ \tilde{x}_{k+1} \\ y_k \\ u_k \end{bmatrix} = \left[\begin{array}{cccc|cc} A+BF & -HC_s & B\Theta_k & -BF & B & 0 \\ 0 & A_s & B_s\Theta_k & -B_sF & B_s & 0 \\ 0 & -B_qC_s & A_q & -B_qC & 0 & -B_q \\ 0 & 0 & 0 & A+HC & B & H \\ \hline C+DF & C_s & D\Theta_k & -DF & 0 & I \\ F & 0 & \Theta_k & F & 0 & 0 \end{array}\right] \begin{bmatrix} x_k \\ v_k \\ z_k \\ \tilde{x}_k \\ w_{1,k} \\ w_{2,k} \end{bmatrix}. \tag{5.5}$$

The state variable is now $x_k = (\, x_k' \; v_k' \; z_k' \; \tilde{x}_k' \,)'$, where again $\tilde{x}_k$ is the state estimation error $\tilde{x}_k = x_k - \hat{x}_k$.

Obviously the stability of the above closed loop (5.5) system hinges on the stability of the matrix

$$\begin{bmatrix} A_s & B_s\Theta_k \\ -B_qC_s & A_q \end{bmatrix}.$$

Due to the presence of the unmodeled dynamics, it is also obvious that the interconnection between the disturbance signal w_k and our performance signal $e_k = \left[\begin{smallmatrix} y_k \\ u_k \end{smallmatrix}\right]$ is no longer affine in Θ_k. $C_s \neq 0$ is the offending matrix.

In order to present the equivalent of Lemma 2.1 for the system (5.5) we make the following assumption concerning the effect of the unmodeled dynamics:

Assumption 5.1. *There exists a positive constant Θ_s such that for all $\|\Theta\| \leq \Theta_s$*

$$\left| \operatorname{eig} \begin{bmatrix} A_s & B_s\Theta \\ -B_qC_s & A_q \end{bmatrix} \right| < \lambda_s < \lambda_0 < 1 \tag{5.6}$$

for some $\lambda_s > 0$. (λ_0 is defined in (2.6)*).*

In essence we are requiring that the unmodeled dynamics are well outside the bandwidth of the nominal controller, to the extent that for any Θ modification of the controller with “gain” bounded by Θ_s the dominant dynamics are determined by the nominal model and nominal controller system. The bound Θ_s can be conservatively estimated from a small gain argument. In order to obtain a reasonable margin for adaptive control, with Θ_s not too small, it is important that the nominal controller has small gain in the frequency band where the unmodeled dynamics are significant.

The small gain argument is obtained as follows. Denote $S(z) = C_S(zI - A_S)^{-1}B_S$ and $H_\Theta(z) = (zI - A_q)^{-1}B_q$. Then, the interconnection of (Q, S)

will be stable provided that

$$\|\Theta H_\Theta(z)S(z)\| < 1 \quad \text{on } |z| = 1,$$

or

$$\|\Theta\| < \frac{1}{\|H_\Theta(z)S(z)\|} \quad \text{on } |z| = 1. \tag{5.7}$$

From these inequalities we deduce that in order to have an effective plug in controller Q, the controller transfer function $H_\Theta(z)$ should be negligible in the frequency band where significant model-plant mismatch is expected. As explained before, due to the frequency weighting, $S(z)$ is only significant outside the passband of the nominal controller.

With the above assumption (5.6) we are now in a position to state the equivalent of Lemma 2.1 for the system (5.5).

Lemma 5.2. *Consider the system* (5.5). *Let Assumption 5.1 hold. Let* $\|w_k\| \le W$ *and let condition* (2.5) *and* (2.6) *from Lemma 2.1 hold. There exists a* $\Delta > 0$ *such that for all sequences* Θ_k *such that* $\|\Theta_k\| \le \Theta_s$ *and* $\|\Theta_{k+1} - \Theta_k\| \le \Delta$, *the state of the system* (5.5) *is bounded. In particular, there exists positive constants* C_0, C_1 *such that:*

$$\begin{aligned}(|z_k|, |v_k|, |\tilde{x}_k|) &\le C_0\lambda_0^k(|z_0|, |v_0|, |\tilde{x}_0|) + C_0(1 - \lambda_0^k)W,\\ |x_k| &\le C_1\lambda_1^k(|x_0| + |\tilde{x}_0| + |v_0| + \Theta_s |z_0|)\\ &\quad + C_1(1 + \Theta_s)(1 - \lambda_1^k)W.\end{aligned}$$

Lemma 5.2 is considerably weaker than Lemma 2.1. Due to the presence of the unmodeled dynamics we not only need to restrict the amount of adaptation of Θ_s, but also need to restrict the adaptation speed; at least if we want to guarantee that adaptation is *not* going to alter the stability properties of the closed loop. We argue that the requirement that adaptation improves the performance of a nominal control design is essential in practical applications. From this perspective, Lemma 5.2 established the minimal (be it conservative) requirements that need to be imposed on any adaptation algorithm we wish to implement. It provides yet another pointer for why slow adaptation is essential.

6.6 Adaptive Algorithm

Let us adjust Θ_k so as to minimize the criterion (3.1) within the limitation imposed by Lemma 5.2. An approximate steepest descent algorithm for updating Θ_k looks like:

$$\Theta_{ij,k+1} = \Theta_{ij,k} - \mu\gamma'_{ij,k} Re_k; \qquad i = 1 \dots p, \qquad j = 1 \dots n_q. \tag{6.1}$$

Here $\gamma_{ij,k}$ is a column vector of approximate sensitivity functions given by equation (3.3). When used in conjunction with system (2.4) the $\gamma_{ij,k}$ are asymptotically exact estimates for the sensitivity functions, whereas in the presence of the unmodeled dynamics v_k, they are only approximate estimates.

As indicated in Lemma 5.2 the basic algorithm (6.1) needs to be modified using projection to guarantee that $\|\Theta\| \leq \Theta_s$ and μ needs to be sufficiently small so as to guarantee that $\|\Theta_{k+1} - \Theta_k\| < \Delta$. Notice that in essence the adaptive algorithm is identical to the algorithm proposed in Section 6.3.

Before proceeding with the analysis, we provide here the state space realization for the exact sensitivity functions with respect to a change in Θ_{ij}, denoted $g_{ij,k}$. Directly from (5.5) we obtain:

$$\left[\begin{array}{c} G_{ij,k+1} \\ \hline g_{ij,k} \end{array}\right] = \left[\begin{array}{ccc|c} A+BF & -HC_s & B\Theta & BE_{ij} \\ 0 & A_s & B_s\Theta & B_sE_{ij} \\ 0 & -B_qC_s & A_q & 0 \\ \hline C+DF & C_s & D\Theta & DE_{ij} \\ F & 0 & \Theta & E_{ij} \end{array}\right] \left[\begin{array}{c} G_{ij,k} \\ \hline z_k \end{array}\right]; \qquad G_{ij,0} = 0. \tag{6.2}$$

Clearly with $C_s = 0$ we recover up to exponentially decaying terms the expression for the sensitivity functions as given in equation (3.3). The difference between the true nonimplementable sensitivities $g_{ij,k}$ and the approximate implementable sensitivities $\gamma_{ij,k}$ is governed by:

$$\left[\begin{array}{c} \tilde{G}_{ij,k} \\ \hline g_{ij,k} - \gamma_{ij,k} \end{array}\right] = \left[\begin{array}{ccc|c} A+BF & -HC_s & B\Theta & 0 \\ 0 & A_s & B_s\Theta & B_sE_{ij} \\ 0 & -B_qC_s & A_q & 0 \\ \hline C+DF & C_s & D\Theta & 0 \\ F & 0 & \Theta & 0 \end{array}\right] \left[\begin{array}{c} \tilde{G}_{ij,k} \\ \hline z_k \end{array}\right]; \qquad \tilde{G}_{ij,0} = 0. \tag{6.3}$$

From the above expression it follows that the implemented sensitivity function $\gamma_{ij,k}$ is a good approximation for the exact sensitivity function $g_{ij,k}$ under the now familiar conditions: unmodeled dynamics are effective only outside the passband of the nominal controller, and the disturbances w_k to be rejected are limited in frequency content to the passband of the nominal controller. Notice also that the linear gain from z_k to $g_{ij,k} - \gamma_{ij,k}$ may be over-estimated by a constant proportional to $|C_s|$, which again confirms that for $C_s = 0$ the approximate sensitivity function is asymptotically exact.

6.7 Analysis of the Adaptive-Q Algorithm: Unmodeled Dynamics Situation

In this section, in studying the unmodeled dynamics situation, we proceed along the same lines used for the analysis of the adaptive algorithm under ideal modeling, see Section 6.4. In particular we assume that the external signals w_k are both stationary and sufficiently rich.

Unfortunately, due to the presence of the unmodeled dynamics we can no longer conclude that $J(\Theta)$ is quadratic in Θ. Indeed, we have for Θ constant, using Parseval's Theorem:

$$\begin{aligned} J(\Theta) &= \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} e_k' R e_k \\ &= \frac{1}{2\pi} \int_0^{2\pi} W^*(e^{i\theta}) T_\Theta^*(e^{i\theta}) R T_\Theta(e^{i\theta}) W(e^{i\theta}) d\theta. \end{aligned} \tag{7.1}$$

Here the transfer function from the disturbance $w_k = \left[\begin{smallmatrix} w_{1,k} \\ w_{2,k} \end{smallmatrix}\right]$ to $e_k = \left[\begin{smallmatrix} y_k \\ u_k \end{smallmatrix}\right]$, denoted T_Θ, has a state space realization as indicated in (5.5) and $W(z)$ is the z-transform corresponding to the external signal w. Obviously T_Θ is not affine in Θ, hence $J(\Theta)$ is not quadratic in Θ.

Under these circumstances, we can establish the following result:

Theorem 7.1. *Consider the adaptive system described by* (3.1), (3.2) *and* (5.5). *Let* $\theta \le \Theta_S$. *Assume that the conditions* (2.5), (2.6) *and* (5.6) *are met. Let the external signal* w *satisfy the Assumptions 4.1 and 4.2. Then there exists a* $\mu^* > 0$ *such that for all* $\mu \in (0, \mu^*)$ *and* $\|\Theta_0\| < \theta$ *and all* $\tilde{x}_0$, x_0, z_0 *and* v_0, *the system state is bounded in that* $\|\Theta_k\| < \Theta_S$ *and Lemma 5.2 holds. Consider the difference equation*

$$\Theta_{ij,k+1}^{av} = \Theta_{ij,k}^{av} - \mu \left. \frac{\partial J(\Theta)}{\partial \Theta_{ij}} \right|_{\Theta=\Theta_k^{av}} + \mu\epsilon b_{ij}(\Theta_k^{av}), \tag{7.2}$$

with $\epsilon = \|C_S\|$ *and*

$$b_{ij}(\Theta) = \lim_{N\to\infty} \frac{1}{N\epsilon} \sum_{k=1}^{N} (g_{ij,k}(\Theta) - \gamma_{ij,k}(\Theta))' R e_k(\Theta), \tag{7.3}$$

where $g_{ij,k}(\Theta) - \gamma_{ij,k}(\Theta)$ *is described in* (6.3) *and*

$$e_k(\Theta) = \begin{bmatrix} y_k(\Theta) \\ u_k(\Theta) \end{bmatrix} \tag{7.4}$$

follows from (5.5) *with* $\Theta_k = \Theta$. *Provided* (7.2) *has locally stable equilibria* $\Theta^* \in B(0, \Theta_S)$, *then* Θ_k *converges for almost all initial conditions* $\|\Theta_0\| < \theta$ *to a* $\mu^{(1-\gamma)/2}$ *neighborhood of such an equilibrium.*

Proof. Follows along the lines of Theorem 4.3. □

Equation (7.2) is crucial in understanding the limiting behavior of Θ_k. If no locally stable equilibria exists inside the ball $B(0, \Theta_S)$, a variety of dynamical behaviors may result, all characterized by bad performance. The offending term in (7.2) is the bias term $b_{ij}(\Theta)$. Provided $\epsilon b_{ij}(\Theta)$ is small in some sense, good performance will be achieved asymptotically.

As explained, the bias $b_{ij}(\Theta)$ will be minimal when the disturbance signal has little energy in the frequency band of the unmodeled dynamics. One conclusion is that the adaptive algorithm provides good performance enhancement under the reasonable condition that the model is a good approximation in the frequency band of the disturbances.

Notice that these highly desirable properties can be directly attributed to the exploitation of the Q-parameterization of the stabilizing controllers. Standard implementations of direct adaptive control algorithms are not necessarily robust with respect to unmodeled dynamics! (See for example Rohrs, Valavani, Athans and Stein (1985) and Anderson et al. (1986).)

The above result indicates that the performance of the adaptive algorithm may be considerably weaker than the performance obtained under ideal modeling, embodied in Theorem 4.3. Alternatively, Theorem 7.1 explores the robustness margins of the basic adaptive algorithm. Indeed, in the presence of model mismatch, the algorithm fails gracefully. Small model-plant mismatch (small ε) implies small deviations from optimality. Notice that the result in Theorem 7.1 recovers Theorem 4.3, by setting $\varepsilon = 0$! However, for significant model-plant mismatch we may expect significantly different behavior. The departure from the desired optimal performance is governed by the $b_{ij}(\Theta)$ term in (7.3). It is instructive to consider how this bias term comes about. A frequency domain interpretation is most appropriate:

$$b_{ij}(\Theta) = \frac{1}{2\pi\varepsilon}\int_0^{2\pi} W^*(e^{i\theta})T^*_{z(g-\gamma)_{ij}}(e^{i\theta})T^*_{wz}(e^{i\theta})RT_{we}(e^{i\theta})W(e^{i\theta})d\theta,$$

where $T_{z(g-\gamma)_{ij}}$ has a state space realization as given in (6.3) and T_{wz}, T_{we} have state space realizations given in (5.5), and are respectively the transfer functions from r to $(g-\gamma)_{ij}$, w to z and w to e.

In Wang (1991) a more complete analysis of how the adaptive algorithm fails under model plant mismatch conditions is presented. It is clear however that under severe model mismatch the direct adaptive Q mechanism is bound to fail. Under such conditions reidentification of a model has to be incorporated in the adaptive algorithm. This is the subject of the next chapter.

Example. Having presented the general theory for direct adaptive-Q control, we develop now a simple special case were the external signal, $w_1 = 0$, $w_{2k} = \cos \omega_1 k$ and the adaptive-Q filter contains a single free parameter. Whereas the development of the general theory by necessity proceeded in the state space domain, due to the slow adaptation the main insight could be obtained via transfer

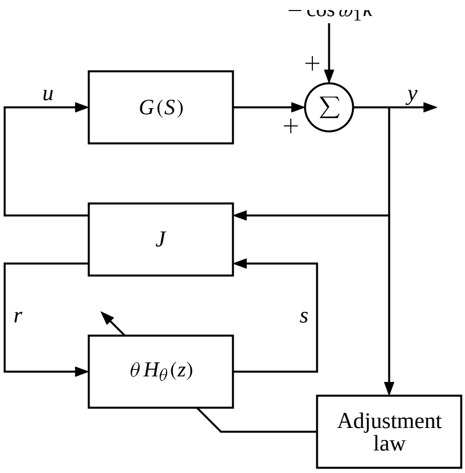


FIGURE 7.1. Example

functions and frequency domain calculations. For the example we proceed by working with transfer functions.

Consider the control loop as in Figure 7.1. All signals u, y, r and s are scalar valued. Let the control performance variable be simply $e = y$. We are thus interested in minimizing the rms value of y:

$$J(\theta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} y_k^2(\theta).$$

The plant is given by

$$G(z) = \frac{N(z) + S(z)V(z)}{M(z) + S(z)U(z)}.$$

The controller is given by

$$K(z) = \frac{U(z) + \theta H(z)M(z)}{V(z) + \theta H(z)N(z)}.$$

with $M(z)V(z) - N(z)U(z) = 1;\ N, S, V, M, U, H \in RH_\infty$.

For constant θ we have

$$\begin{aligned} y_k(\theta) &= -\frac{(M(z) + S(z)U(z))(V(z) + \theta H(z)N(z))}{1 - S(z)\theta H(z)} \cos\omega_1 k, \\ u_k(\theta) &= -\frac{M(z) + S(z)U(z))(U(z) + \theta H(z)M(z))}{1 - S(z)\theta H(z)} \cos\omega_1 k, \\ \gamma_k(\theta) &= H(z)N(z)(M(z)y_k(\theta) - N(z)u_k(\theta)). \end{aligned} \tag{7.5}$$

The update algorithm for θ (with exponential forgetting $\lambda \in (0, 1)$) is thus given by (compare with (3.5))

$$\theta_{k+1} = (1 - \mu\lambda)\theta_k - \mu\gamma_k y_k. \tag{7.6}$$

Following the result of Theorem 7.1 the asymptotic behavior of (7.6) is governed by the equation

$$\theta_{k+1}^{av} = (1 - \mu\lambda)\theta_k^{av} - \mu g(\theta_k^{av}), \tag{7.7}$$

where $g(\theta)$ is given by

$$g(\theta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \gamma_k(\theta) y_k(\theta).$$

When $S(z) = 0$, that is the ideal model case, $g(\theta)$ can be evaluated using (7.5) as

$$g_i(\theta) = \left|M(e^{j\omega_1})\right|^2 \left(\operatorname{Re} V(e^{-j\omega_1})H(e^{j\omega_1})N(e^{j\omega_1}) + \theta \left|H(e^{j\omega_1})N(e^{j\omega_1})\right|^2\right), \tag{7.8}$$

where the index i reflects the situation that $S(z) = 0$.

In general we obtain the rather more messy expression:

$$\begin{aligned} g(\theta) = &\left| \frac{M(e^{j\omega_1}) + S(e^{j\omega_1})U(e^{j\omega_1})}{1 - S(e^{j\omega_1})H(e^{j\omega_1})\theta} \right| \\ &\cdot \left(\operatorname{Re}\left\{H(e^{-j\omega_1})N(e^{-j\omega_1})V(e^{j\omega_1})\right\} + \theta \left|H(e^{-j\omega_1})N(e^{-j\omega_1})\right|^2 \right). \end{aligned} \tag{7.9}$$

Let us discuss the various scenarios described in, respectively, Theorems 4.3, 4.5 and 7.1 using the above expressions (7.8) or (7.9).

1. **Ideal case:** $\lambda = 0$ **(Theorem 4.3), expression** (7.8)**.**
   There is a unique, locally stable equilibrium; $g(\theta^*) = 0$, or

$$\theta^* = -\frac{\operatorname{Re}\left\{H(e^{-j\omega_1})N(e^{-j\omega_1})V(e^{j\omega_1})\right\}}{\left|H(e^{-j\omega_1})N(e^{-j\omega_1})\right|^2}. \tag{7.10}$$

   This equilibrium achieves the best possible control. The adaptation approximates this performance. Indeed, it is easily verified that

$$J(\theta^*) \le J(\theta) \quad \text{for all } \theta.$$

   In this case the performance criterion $\|y\|^2 = J(\theta)$ is given by

$$\|y\|_{\text{rms}}^2 = \left|M(e^{j\omega_1})\right|^2 \left|V(e^{j\omega_1}) + \theta H_\theta(e^{j\omega_1})N(e^{j\omega_1})\right|^2.$$

   In general, we can not expect to zero the output, unless the actual signal happened to be a constant $\omega_1 = 0$, in which case we indeed obtain $\theta^* = -V(1)/(H(1)N(1))$ and $J(\theta^*) = 0$.

2. **Ideal case: $\lambda \neq 0$ (Theorem 4.5).**
   Again there is a unique, locally stable equilibrium, now given by

$$\theta^*_\lambda = -\frac{\operatorname{Re}\{H(e^{-j\omega_1})H(e^{j\omega_1})N(e^{j\omega_1})\}\left|M(e^{j\omega_1})\right|^2}{\lambda + \left|M(e^{j\omega_1})\right|^2 \left|H_\theta(e^{j\omega_1})N(e^{j\omega_1})\right|^2}. \tag{7.11}$$

   In this case, the performance of θ^*_λ is no longer the best possible, but clearly for small λ, we have that $\left|\theta^*_\lambda - \theta^*\right| = O(\lambda)$, see (7.10) and (7.11).

3. **Plant-model mismatch: $\lambda = 0$ (Theorem 7.1).**
   Remarkably, there is again a locally stable equilibrium θ^*, but also there may exist an equilibrium at ∞ as $g(\theta) \to 0$ for $\theta \to \pm\infty$. Clearly in the presence of unmodeled dynamics, the adaptive algorithm loses the property of global stability.

   Despite the fact that θ^* is always a locally stable equilibrium of (2.2), it may not be an attractive solution for the adaptive system. Indeed Theorem 7.1 requires that the closed-loop system $(S(z), H_\theta(z)\theta)$ be stable, a A property that may be violated for $\theta = \theta^*$ if $S(z)$ is not small. Theorem 7.1 requires at least that $\left|S(e^{j\omega_1})H(e^{j\omega_1})\theta^*\right| < 1$. If this is not the case, the adaptive system will undergo a phenomenon known as bursting. See Anderson et al. (1986) or Mareels and Polderman (1996) for a more in-depth discussion of this phenomenon. Let it suffice here to state that whenever θ^*, the equilibrium of (2.2), is such that $(S(z), H(z)\theta^*)$ is unstable, the adaptive system performance will be undesirable.

   The performance of θ^* is also not optimal with respect to our control criterion. When $S(z) \neq 0$, the criterion becomes:

$$\|y\|^2_{\text{rms}} = \frac{\left|V(e^{j\omega_1}) + \theta H(e^{j\omega_1})N(e^{j\omega_1})\right|^2 \left|M(e^{j\omega_1}) + S(e^{j\omega_1})U(e^{j\omega_1})\right|^2}{\left|1 - S(e^{j\omega_1})H(e^{j\omega_1})\theta\right|^2}.$$

   It can be verified that the performance at the equilibrium θ^* will be better than the performance of the initial controller $\theta = 0$ if and only if

$$\begin{aligned}\operatorname{Re}\left\{N(e^{-j\omega_1})V(e^{j\omega_1})H(e^{j\omega_1})\right\}\operatorname{Re}\left\{S(e^{j\omega_1})H(e^{j\omega_1})\right\}& \\ \geq -\frac{1}{2}\left(\operatorname{Re}\left\{N(e^{-j\omega_1})V(e^{j\omega_1})H(e^{j\omega_1})\right\}\right)^2 & \\ \cdot\left(\frac{1}{\left|V(e^{j\omega_1})\right|} + \left|\frac{S(e^{j\omega_1})}{N(e^{j\omega_1})}\right|^2\right). &\end{aligned}$$

   This condition is always satisfied for sufficiently small $\left|S(e^{j\omega_1})\right|$. The above expression for this example is the precise interpretation of our earlier observation that the direct adaptive Q filter can achieve good performance provided S, the plant-model mismatch, is small in the passband of the controller and provided the external signals are inside the passband of the nominal control loop.

4. **Plant-model mismatch: $\lambda \neq 0$.**
   Due to the presence of λ as well as model-plant mismatch, we now end up with the possibility of either 3 or 1 equilibria. Indeed the equilibria are the solutions of $\lambda\theta + g(\theta) = 0$ which leads to a third order polynomial in θ. For small values of $\lambda > 0$, there is a locally stable equilibrium θ_λ^* close to θ^*. Global stability is of course lost, and the same stability difficulties encountered in the previous subsection persist here.

6.8 Notes and References

Direct adaptive-Q control has been studied in some detail in Wang (1991), building on the earlier work of Tay and Moore (1991) and Tay and Moore (1990). The paper Wang and Mareels (1991) contains the basic ideas on which the present theory is built. Most of the material in this chapter has not been published before.

For a more complete presentation of averaging techniques in continuous-time setting, we refer the reader to Sanders and Verhulst (1985). Averaging ideas have been used extremely successfully in the analysis of adaptive systems. We refer the reader to Anderson et al. (1986), Mareels and Polderman (1996) and Solo and Kong (1995) for more in depth treatises. Much of the presentation here could have been presented using a stochastic framework. The book of Benveniste, Metivier and Priouret (1991) is a good starting point. The treatment of the actual dynamical behavior of adaptive systems has, by necessity, been superficial. One should not lose sight of the fact that in general, an adaptive scheme leads to a nonlinear and time-varying control loop, the behavior of which is all but simple. We have identified conditions under which the adaptive system response can be understood from the point of view of frequency domain ideas using the premise that the adaptation proceeds slowly compared to the actual dynamics of the controlled loop. Whenever this condition is violated, adaptive system dynamics become difficult to comprehend. The interested reader is referred to Mareels and Polderman (1996, Chapter 9).

Problems

Problems 1, 2 and 3 are intended to give the reader some insight into averaging.

1. Consider

$$x_{k+1} = (1 - a\mu)x_k + \mu(1 + \cos k). \tag{8.1}$$

   Here x_k a, μ are scalar variables, $\mu > 0$ and small, $a \in \mathsf{R}$. Also, consider the 'averaged' difference equation

$$x_{k+1}^a = (1 - a\mu)x_k^a + \mu. \tag{8.2}$$

(a) Compare the solution of (8.1) and (8.2), both initialized with the same initial condition x_0. Show in particular that $\left|x_k - x_k^a\right| \leq C\mu,\ k = 0, 1, \ldots, L/\mu$ for any $L > 0$ and some constant C independent of μ, but dependent on L.

(b) If $a < 0$, show that the error $\left|x_k - x_k^a\right| \leq C\mu$ for all k.

In a sense, averaging is a technique that formalizes the 'low pass' characteristics of a difference equation of the form $x_{k+1} = x_k + \mu f_k(x_k)$.

2. Show that any signal of the form $\Sigma_{\ell=1}^{M} \cos \omega_l k$ has a uniform zero average.

3. Show that any signal w_k that converges to zero has zero mean. (Averaging eliminates transients!)

The following problem illustrates the theory of direct adaptive-Q design and allows the reader to venture beyond the theory through simulation.

4. Refer to Figure 7.1. Let

$$G(z) = \frac{2-\sqrt{2}}{2}\,\frac{z+1}{z^2-\sqrt{2}z+1},$$

$$K(z) = \frac{-\frac{1}{2+\sqrt{2}}z + \frac{1+\sqrt{2}}{2+\sqrt{2}}}{z^2+\sqrt{2}z+\frac{1+\sqrt{2}}{2+\sqrt{2}}}\,\frac{2}{2-\sqrt{2}}.$$

Let the signal to be tracked be $\cos((\pi/4)k)$, $\cos 1.75k$ or $\cos((\pi/4)k) + \cos 1.75k$. Explain for each of these reference signals the performance that the adaptive algorithm achieves; use a single scalar adaptive θ. Let $\mu = 0.01,\ H_\theta(z) = z^{-1}$. (One could also utilize $H_\theta(z) = 1$ in this case). Consider the following questions:

(a) Show that the original controller achieves dead beat response.

(b) Show that the original controller has perfect plant output tracking for any signal of the form $A\cos((\pi/4)k + \varphi)$.

(c) Show that the adaptive-Q filter will not deteriorate the performance of the loop (if μ is small).

(d) Despite the fact that a single θ can not achieve sinusoidal tracking, show that significant improvement is obtained for either $w_k = \cos 1.75k$ or $w_k = \cos((\pi/4)k) + \cos 1.75k$.

(e) Using averaging ideas, find the optimal performance that can be achieved for $w_k = \cos \omega k$. (That is, plot $J(\theta^*)$ as a function of $\omega \in (0, \pi)$). What do you learn from this plot?

(f) Using the adaptive $Q(z) = \theta_1 + \theta_2 z^{-1}$, show that exact tracking can be achieved for $w_k = \cos((\pi/4)k) + \cos \omega k$, any ω.

(g) Introduce $S(z) = \tau z^{-1}$. show that this amounts to a significant perturbation in the low frequency response; that is, this is a significant perturbation, and in particular, the plant's resonance disappears.

(h) Continuing with the scenario ending Problem 4g, introduce forgetting and use a scalar adaptive $Q(z) = \theta$. Find the equilibria as a function of λ, and τ for $w_k = \cos 1.75k$.

(i) When does the closed-loop adaptive system lose stability? This last part should not be attempted unless you have a few spare hours. A complete analysis comprising (a complete range of) λ, τ has never been completed, let alone an analysis considering ranges of the parameters μ, λ, τ and ω.

CHAPTER 7

Indirect (Q, S) Adaptive Control

7.1 Introduction

Direct adaptive-Q design is applicable in those situations that a reasonably good model for the plant dynamics is available. It is geared towards tuning of controllers, and in particular for tuning controllers for new design criteria without sacrificing stability, and also towards disturbance rejection. In some situations, our limited knowledge of the plant dynamics may be the greatest obstacle on the route to better control performance. In such situations, the iterative and nested (Q, S) designs are needed.

In this chapter, we will present an adaptive version of the nested (Q, S) methodology in that identification and control are both adjusted on-line as new data becomes available. Adaptive methods where the controller design is based on an on-line identified model are often referred to as *indirect adaptive control* methods. Hence the title, indirect adaptive (Q, S) design.

In the present case, where the lack of a sufficiently accurate model of the plant is the major issue in obtaining satisfactory control performance, we need to gather information about the plant. In the spirit of the book, this is achieved by identification of an S model. As before, we assume that a stabilizing controller is available, and hence we can parameterize a class of systems which contains the plant. In order to identify the S parameter and thus the actual plant on-line, in a closed-loop control context, we inject an external signal into the closed loop at the output of the plug-in controller Q. This probing signal is required to gain information about S and as a consequence, will necessarily frustrate to some extent any control action for as long as it is present. This is the penalty we must be prepared to pay for our lack of knowledge of S and our desire to obtain better control performance. As pointed out before in Chapter 5, identification of S in closed-loop operation is nontrivial. The methods proposed in Chapter 5 to circumvent the problem in iterated-Q design are not particularly well suited for an adaptive-Q approach. Indeed, these methods invariably rely on the time invariance of Q, a property which is lost by the very nature of an adaptive design. One solution could be to use two time scales in the adaptive algorithm; one, the fast learning time scale, for the adaptation of a model for S and a second much slower time scale for the adaptation of Q. This way, we would recover more or less the same behavior as in the nested design. If desired, one could alternate time scales between S identification and Q design at preset intervals to recover a full adaptive version of a multiple nested (Q, S) design method. The idea has been worked out in detail, and analyzed using averaging techniques in the PhD thesis of Wang (1991).

Here we explore a different method for the case where plant-model mismatch S is significant only in the frequency range above the passband of the nominal control loop. Probing signals are thus best utilized if their frequency content is located around and past the cut off frequency of the closed loop. The probing signals will only marginally affect the control performance. In order for Q to be effective, it has to shape the response in the same frequency range where S is important. The presence of Q frustrates the identification of S in so far as the probing signals affect Q. The idea is to augment Q with a filter at its input that filters out the probing signals. Of course, this limits our control ability, but ensures good identification, and thereby control response improvement. These ideas are worked out in this chapter, making use of ℓ_2-optimization design criteria, as these fit most naturally with the averaging techniques for the system analysis.

The chapter is organized as follows. First we discuss the basic framework. As in the previous chapter to facilitate our analysis techniques, the development proceeds in the state space framework. Next we discuss the adaptive mechanisms and present some results exploring the time scale separation between the adaptive mechanism and the control dynamics. The chapter is concluded with a discussion of an example.

7.2 System Description and Control Problem Formulation

Our starting point is a stable plant-controller configuration which is characterized by unacceptable performance due to an inaccurate nominal plant model for the controller design. Again G represents the nominal plant, K the nominal controller, $\bar{G}$ the actual plant, S embodies the plant-nominal plant mismatch and Q is the plug-in controller to be designed.

Since the nominal controller K stabilizes $\bar{G}$, the mismatch system S is stable. For estimation purposes, we introduce the following parameterized class of stable rational transfer functions:

$$\sum_{i=1}^{N} \Xi_i B_i(z) = \Xi B(z). \tag{2.1}$$

The $B_i(z)$ form a collection of basis functions, being stable rational transfer functions. The precise choice of $B_i(z)$ should reflect any prior knowledge of the plant uncertainty $S(z)$. The matrices Ξ_i, collected into the coefficient matrix Ξ, have to be estimated from on-line data.

We refer to the situation where $S(z)$ is indeed of the form (2.1) as the ideal case. In this situation there exists a unique coefficient matrix Ξ^* such that $S(z) = \Xi^* B(z)$. When no such representation exists we speak of the nonideal case. For future reference, let $B(z)$ possess a state space realization:

$$B(z) : \left[\begin{array}{c|c} A_s & B_s \\ \hline C_s & 0 \end{array}\right], \tag{2.2}$$

with A_s a stable matrix. As indicated before, the dominant eigenvalues of A_s must be compatible with the closed-loop bandwidth of the system $(\bar{G}, K)$.

In the ideal case we have for $S(z)$

$$S(z) : \left[\begin{array}{c|c} A_s & B_s \\ \hline \Xi^* C_s & 0 \end{array}\right]. \tag{2.3}$$

In the nonideal case we represent $S(z)$ as

$$S : \left[\begin{array}{cc|c} A_\Delta & 0 & B_\Delta \\ 0 & A_s & B_s \\ \hline C_\Delta & \Xi^* C_s & 0 \end{array}\right]. \tag{2.4}$$

Here A_Δ is a stable matrix, and $\|C_\Delta\|$ can be considered as a measure of nonidealness, being zero in the ideal case.

Remark. A linear parameterization for $S(z)$ such as $\Xi^* B(z)$ may not be the most economical way to parameterize the unmodeled dynamics. A rational fraction parameterization may require quite fewer parameters, but has the difficulty that somehow the parameters must be restricted to represent a stable $S(z)$. This is nontrivial, as the set of stable rational transfer functions does not have a simple representation in parameter space. The present approach avoids this complication and gives transparency to the associated performance analysis.

Let the nominal plant G be represented as:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Bw_{1,k}; \qquad x_0, \\ y_k &= Cx_k + Du_k + w_{2,k}, \end{aligned}$$

where $w_{1,k}$ is an input disturbance, $w_{2,k}$ is an output disturbance. Also, $x_k \in \mathsf{R}^n$, $u_k \in \mathsf{R}^p$, $y_k \in \mathsf{R}^m$. The nominal controller K is represented as:

$$\begin{aligned} \hat{x}_{k+1} &= A\hat{x}_k + Bu_k - Hr_k; \qquad \hat{x}_0, \\ r_k &= y_k - (C\hat{x}_k + Du_k), \\ u_k &= F\hat{x}_k + s_k, \end{aligned}$$

where r_k is the observer innovation and s_k is the auxiliary control signal.

The plug-in controller Q takes the special form:

$$\begin{aligned} z_{f,k+1} &= A_f z_{f,k} + B_f r_k; \qquad z_{f,0}, \\ r_{f,k} &= C_f z_{f,k}, \\ z_{k+1} &= A_q z_k + \Lambda_k z_k + B_q r_{f,k}; \qquad z_0, \\ s_k &= \Theta_k z_k + d_k. \end{aligned}$$

The signal d_k is an external signal added to aid identification of Ξ. The parameter matrices Λ_k and Θ_k are to be updated adaptively on the basis of Ξ_k, the present estimate of Ξ^*. As earlier, the remaining Q parameters are chosen *a priori*. The matrices A_f, A_q are stable. The system (A_f, B_f, C_f) is designed to have the specific property that the steady state response of the filter subject to the input d_k as input is zero. Thus for the system

$$\begin{aligned} z_{f,k+1} &= A_f z_{f,k} + B_f d_k, \\ d_{f,k} &= C_f z_{f,k}, \end{aligned} \tag{2.5}$$

$\lim_{k\to\infty} d_{f,k} = 0$. As far as the probing signal goes, the controlled system looks like an open-loop system.

Finally, the actual plant $\bar{G}$, can now be represented as:

$$\left[\begin{array}{c} x_{k+1} \\ v_{\Delta,k+1} \\ v_{k+1} \\ \hline y_k \end{array}\right] = \left[\begin{array}{ccc|ccc} A & -HC_\Delta & -H\Xi^* C_s & B & B & 0 \\ -BF & A_\Delta & 0 & B_\Delta & B_\Delta & 0 \\ -B_s F & 0 & A_s & B_s & B_s & 0 \\ \hline C & C_\Delta & \Xi^* C_s & D & 0 & I \end{array}\right] \left[\begin{array}{c} x_k \\ v_{\Delta,k} \\ v_k \\ \hline u_k \\ w_{1,k} \\ w_{2,k} \end{array}\right]. \tag{2.6}$$

The complete closed-loop equations are:

$$\left[\begin{array}{c} x_{k+1} \\ v_{\Delta,k+1} \\ v_{k+1} \\ z_{f,k+1} \\ z_{k+1} \\ \tilde{x}_{k+1} \\ \hline y_k \\ u_k \end{array}\right] = \left[\begin{array}{cccccc|ccc} A+BF & -HC_\Delta & -H\Xi^* C_s & 0 & B\Xi_k & BF & B & 0 & B \\ 0 & A_\Delta & 0 & 0 & B_\Delta \Theta_k & B_\Delta F & B & 0 & B \\ 0 & 0 & A_s & 0 & B_s \Theta_k & B_s F & B_\Delta & 0 & B_\Delta \\ 0 & -B_f C_\Delta & -B_f \Xi^* C_s & A_f & 0 & B_f C & 0 & -B_f & 0 \\ 0 & 0 & 0 & B_q C_f & A_q + \Lambda_k & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & A+HC & B & H0 & \\ \hline C+DF & C_\Delta & \Xi^* C_s & 0 & D\Theta_k & DF & 0 & I & D \\ F & 0 & 0 & 0 & \Theta_k & F0 & 0 & I & \end{array}\right] \left[\begin{array}{c} x_k \\ v_{\Delta,k} \\ v_k \\ z_{f,k} \\ z_k \\ \tilde{x}_k \\ \hline w_{1,k} \\ w_{2,k} \\ d_k \end{array}\right]. \tag{2.7}$$

In order to be able to develop a meaningful adaptively controlled closed loop we impose the following assumption.

Assumption 2.1. *The plug-in controller structure is such that for almost any* Ξ *the eigenvalues of the matrix*

$$A_c(\Xi, \Theta, \Lambda) = \begin{bmatrix} A_s & 0 & B_s\Theta \\ -B_f\Xi C_s & A_f & 0 \\ 0 & B_qC_f & A_q + \Lambda \end{bmatrix}, \tag{2.8}$$

can be assigned arbitrarily by the appropriate selection of the matrices Θ *and* Λ*. (Arbitrary eigenvalue assignment under the restriction that for any complex eigenvalue its complex conjugate is also to be assigned.)*

Remark. Assumption 2.1 implies generically that we have enough freedom in the controller structure for pole assignment. That is the best we can hope for.

The system matrix (2.8) corresponds to the interconnection of (Q, S), assuming that S belongs to the model class defined by (2.1).

A sufficient condition for arbitrary pole assignment of A_c is that the matrix pair

$$\begin{bmatrix} A_s & 0 \\ -B_f\Xi C_s & A_f \end{bmatrix}, \begin{bmatrix} B_s \\ 0 \end{bmatrix} \tag{2.9}$$

be controllable, the matrix pair

$$\begin{bmatrix} A_s & 0 \\ -B_f\Xi C_s & A_f \end{bmatrix}, \begin{bmatrix} 0 & B_qC_f \end{bmatrix} \tag{2.10}$$

be observable and that the dimension of A_q be at least as large as dim A_s+dim A_f. In this case, we could use the following special choices:

$$\begin{aligned} A_q &= \begin{bmatrix} A_s & 0 \\ 0 & A_f \end{bmatrix} \\ \Lambda &= \begin{bmatrix} 0 & \Lambda_{12}B_qC_f \\ -B_f\Xi C_s & \Lambda_{22}B_qC_f \end{bmatrix}. \end{aligned} \tag{2.11}$$

Hence Λ_{12}, Λ_{22} are chosen such as to arbitrarily assign the eigenvalues of $A_q+\Lambda$, and Θ chosen to place the eigenvalues of

$$\begin{bmatrix} A_s & 0 \\ -B_f\Xi C_s & A_f \end{bmatrix} + \begin{bmatrix} B_f \\ 0 \end{bmatrix}\Theta. \tag{2.12}$$

Clearly the above observability and controllability conditions are generically satisfied.

To complete this section, we now formulate the particular indirect adaptive control problems we discuss.

Problems

1. **Adaptive pole assignment.**
   Consider the system (2.7). Let $C_\Delta = 0$. Assume that all matrices are known except for Ξ^*. The matrices $A + BF$, A_Δ, A_s, A_f, $A + HC$ and A_q are stable. Assume that for $\Xi = \Xi^*$ the eigenvalues of A_c in (2.8) can be arbitrarily assigned by appropriate selection of Θ and Λ. Assume that the external signals $w_{1,k}$, $w_{2,k}$ and d_k are stationary and mutually uncorrelated. The indirect adaptive control objective is to design a controller using the information available in closed loop, being the signals y_k, u_k, z_k, r_k, s_k and d_k such that asymptotically the eigenvalues of $A_c(\Xi^*, \Theta, \Lambda)$ are placed at preassigned stable locations.

2. **Adaptive LQ control.**
   Consider the system (2.7). Let $C_\Delta = 0$, being the ideal S model case. Assume that all system matrices are known except for Ξ^* and that the matrices $A + BF$, $A + HC$, A_Δ, A_s, A_f, and A_q are stable. Assume that for the plant $\bar{G}$, with $\Xi = \Xi^*$, the eigenvalues of (2.8) can be arbitrarily assigned by appropriate selection of Θ and Λ. Assume that the external signals $w_{1,k}$, $w_{2,k}$ and d_k are stationary and mutually uncorrelated. Design an adaptive controller such that the performance index

$$J(\Theta, \Lambda) = \lim_{N\uparrow\infty} \frac{1}{N} \sum_{k=1}^{N} \left(r_k' r_k + s_k' s_k \right)$$

   is minimized.

Remark.

Both problems have received a lot of attention in the adaptive control literature. The important distinction with the classical literature on these topics is the starting assumption of an initial stabilizing controller. This brings a lot of structure to the problem. This structure has been exploited to the maximal extent possible in our system description (2.7).

In setting up the adaptive control problem the designer has much freedom to exploit prior knowledge about the system dynamics. The most important instances of this are the appropriate selection of $B(z)$, the spectrum of the probing signal d_k together with the associated filter (A_f, B_f, C_f).

Main Points of Section

Starting from the assumption that a stabilizing controller is available for the plant to be controlled, we provided a complete description of the closed-loop system, suitable for adaptation (2.7). The mismatch S between the particular nominal plant G and the actual plant $\bar{G}$ has been represented via a parameterized class of stable transfer functions with parameter Ξ. The corresponding parameterized class of plug-in controllers Q, with parameters (Θ, Λ), has special structure to ensure a stable (Q, S) and to filter out the probing signal d used for identification of S, or Ξ. The control objective is to either obtain pole placement for the loop (Q, S) or achieve an LQ control goal.

In the next few sections we introduce the adaptive algorithms, which will be based on the identification of Ξ^*, and then continue with their analysis.

7.3 Adaptive Algorithms

As indicated, we proceed with the introduction of an indirect adaptive control algorithm. First an estimate for Ξ^* is constructed, denoted Ξ_k. This estimate is then used in a typical certainty equivalence manner to determine the control parameter Θ_k and Λ_k. Given an estimate Ξ_k, there exists a mapping $\square(\Xi_k)$ that determines Θ_k, Λ_k. The mapping $\square$ reflects the particular control objective of interest. The problem is that $\square$ is not continuous on all of the parameter space. In the event that the estimate Ξ_k leads to a system description for which either the pair (2.9) fails to be controllable, or the pair (2.10) fails to be observable, $\square$ may not be well defined. This is a manifestation of the so-called pole/zero cancellation problem in adaptive control. Either event is rather unlikely, but not excluded by the identification algorithm we are about to propose. Here, as is traditional in adaptive control, we simply make the assumption that the above event will not occur along the sequence of estimates Ξ_k that the adaptive algorithm produces.

Assumption 3.1. *Along the sequence of estimates Ξ_k, $k = 0, 1, \ldots$, the matrix pair* (2.9) *is controllable and the matrix pair* (2.10) *is observable, in that the smallest singular value of the controllability matrix, respectively observability matrix, is larger than some constant $\sigma > 0$.*

We propose the following adaptive control algorithms:

1. **Filtered Excitation Algorithm.**

$$
\begin{aligned}
\hat{v}_{k+1} &= A_S \hat{v}_k + B_S \Theta_k z_k + B_S d_k; && \hat{v}_0, \\
\hat{z}_{f,k+1} &= A_f \hat{z}_{f,k} - B_f \Xi_k C_S \hat{v}_k; && \hat{z}_{f,0}, \\
g_{k+1} &= A_S g_k + B_S d_k; && g_0 = 0, \\
\gamma_{ij,k+1} &= A_f \gamma_{ij,k} - B_f E_{ij} C_s g_k; && \gamma_{ij,0} = 0, \\
\Xi_{ij,k+1} &= \Xi_{ij,k} - \mu(\hat{z}_{f,k} - z_{f,k})' \gamma_{ij,k}; && \Xi_0, \\
(\Theta_k, \Lambda_k) &= \square(\Xi_k).
\end{aligned}
\tag{3.1}
$$

2. **Classic Adaptation Algorithm.**

$$\begin{aligned}
\hat{v}_{k+1} &= A_s\hat{v}_k + B_s\Theta_k z_k + B_s d_k; && \hat{v}_0,\\
\hat{z}_{f,k+1} &= A_f\hat{z}_{f,k} - B_f\,\Xi_k C_s\hat{v}_k; && \hat{z}_{f,0},\\
\gamma_{ij,k+1} &= A_f\gamma_{ij,k} - B_f E_{ij} C_s\hat{v}_k; && \gamma_{ij,0} = 0,\\
\Xi_{ij,k+1} &= \Xi_{ij,k} - \mu\frac{(\hat{z}_{f,k} - z_{f,k})'\gamma_{ij,k}}{1 + \sqrt{(\hat{z}_{f,k} - z_{f,k})'(\hat{z}_{f,k} - z_{f,k})} + \sqrt{\gamma'_{ij,k}\gamma_{ij,k}}}; && \\
& && \Xi_{ij,0},\\
(\Theta_k, \Lambda_k) &= \Box(\Xi_k). &&
\end{aligned} \tag{3.2}$$

Remarks.

1. In the filtered excitation algorithm, the presence of the probing signal d_k is essential. If $d_k = 0$ then there is simply no adaptation. This provides an interesting mechanism to switch on or off the adaptation. By simply removing the probing signal the controller becomes frozen.

2. In the filtered excitation algorithm the gradient vector $\gamma_{ij,k}$ is only affected by the probing signal itself. The intention is that the other signals will be orthogonal to it, hence on average not affect the identification at all. Due to the particular set up involving a probing signal with an associated filter in the Q loop, the identification process is on average unbiased and similar to an open-loop identification process. This will be demonstrated.

3. It is not a good idea to start with $\Xi_0 = 0$. In this case Assumption 3.1 is automatically violated. An alternative is to have $(\Theta_k, \Lambda_k) = 0$ until such time that Ξ_k satisfies Assumption 3.1. This is a good method to start the adaptive algorithm since for $(\Theta_k, \Lambda_k) = 0$ the closed loop is stable by construction.

4. The update algorithm for Ξ_k is a typical least squares algorithm. The step size $\mu > 0$. In the literature one often finds a normalized update algorithm:

$$\Xi_{ij,k+1} = \Xi_{ij,k} - \mu\frac{\left(\hat{z}'_{f,k} - z_{f,k}\right)'\gamma_{ij,k}}{1 + \gamma'_{ij,k}\gamma_{ij,k}}.$$

   In the filtered excitation algorithm this normalization is superfluous because $\gamma_{ij,k}$ is bounded by construction, regardless of the state of the system. Indeed the normalization is only necessary when it is not clear whether the gradient $\gamma_{ij,k}$ is bounded or not, which is the case when $\gamma_{ij,k}$ is determined by

$$\gamma_{ij,k+1} = A_f\gamma_{ij,k} - B_f E_{ij} C_s\hat{v}_k. \tag{3.3}$$

This is the classical update algorithm. It has the advantage of not requiring a probing signal. In the ideal case one can even demonstrate that it will suffice to solve a weak version of Problem 2. Nevertheless, (3.3) introduces some nontrivial nonlinearities in the identification and control loop, which lead to biased estimates for Ξ^*. The algorithm in (3.1) circumvents this problem altogether. The classical algorithm allows for $0 < \mu < 1$. Here we restrict ourselves, as before, to small values of μ, that is slow adaptation.

5. A poor selection of Ξ_0, with associated $(\Theta_0, \Lambda_0) = \square(\Xi_0)$ may well lead to an initially destabilizing control loop. Our analysis will not deal with this situation explicitly, although we provide evidence, short of a complete proof, to show that in general the adaptive algorithm may recover from this situation.

6. The function $\square$ mapping the identification parameter Ξ to the control parameter Θ, Λ, will be specified later in the discussion of the results we are about to present. For the time being, the main property of importance is that $\square$ leads to a stable closed-loop system, either via pole placement or via LQ design. In order to be able to apply the averaging result we also need the mapping $\square$ to be Lipschitz continuous in a neighborhood of the estimates produced by the adaptive algorithm which follows from Assumption 3.1.

Main Points of Section

We have described more completely adaptive-Q methods by introducing two adaptation algorithms, the classical adaptive algorithm and the filtered excitation algorithm. The control system we are about to analyze consists of (2.7) together with either (3.1) or (3.2).

7.4 Adaptive Algorithm Analysis: Ideal case

Let us consider the adaptive algorithm consisting of (2.7) and (3.1) or (3.2) under the condition that $C_\Delta = 0$. To fix the ideas we focus on the pole placement problem, but it will transpire that the analysis and main results apply to any reasonable design rule. First we study the possible steady state behavior of the closed-loop system under the condition that the external disturbances are zero ($w_k = 0$) and that the probing d_k signal is stationary and sufficiently rich. The steady state analysis indicates that the algorithm may be successful in an adaptive context. Next we consider an averaging analysis, first without signal perturbation, next with signal perturbation. The results provide a fairly complete picture of the cases where the algorithm can be used with success and provide pointers to how it may fail. This topic is taken up in the next section, where we study the influence of plants not belonging to the assumed model class.

Steady State Analysis

Let us consider the situation $w_{1,k} \equiv 0$, $w_{2,k} \equiv 0$ and $\hat{z}_{f,k} - z_{f,k} \equiv 0$. The latter condition implies for either algorithm (3.1) or (3.2) that $\Xi_k \equiv \Xi$ and $(\Theta_k, \Lambda_k) = \square(\Xi) = (\Theta, \Lambda)$. There is no adaptation. Obviously, we want the closed-loop system to behave well under this condition for as there is no adaptation, there is no way the control algorithm can improve the performance. It can be argued that this property is almost necessary for any adaptive algorithm. In the literature it is often referred to as *tunability* or the *tuning property*, see Mareels and Polderman (1996). We show that either algorithm, the classic adaptive algorithm (3.2) as well as the filtered excitation (3.1), possesses the tuning property.

Indeed, given $\hat{z}_{f,k} - z_{f,k} \equiv 0$, we describe the adaptive closed-loop system via the following time-invariant system, making use of the fact that $w_{1,k} \equiv 0$ and $w_{2,k} \equiv 0$.

$$\begin{bmatrix} v_{k+1} \\ z_{f,k+1} \\ z_{k+1} \\ \hat{v}_{k+1} \\ \hat{z}_{f,k+1} \end{bmatrix} = \begin{bmatrix} A_s & 0 & B_s\Theta & 0 & 0 \\ -B_f\Xi^*C_s & A_f & 0 & 0 & 0 \\ 0 & B_qC_f & A_q+\Lambda & 0 & 0 \\ 0 & 0 & B_s\Theta & A_s & 0 \\ 0 & 0 & 0 & -B_f\Xi C_s & A_f \end{bmatrix} \begin{bmatrix} v_k \\ z_{f,k} \\ z_k \\ \hat{v}_k \\ \hat{z}_{f,k} \end{bmatrix} + \begin{bmatrix} B_s \\ 0 \\ 0 \\ B_s \\ 0 \end{bmatrix} d_k. \tag{4.1}$$

Because of the stability of $A + HC$ we have that $\tilde{x}_k$ converges to zero exponentially, and therefore, this state is omitted from (4.1). Moreover, x_k is also omitted, as its stability is determined by the stability of the system (4.1). If the above system is stable, so is the complete system. We now exploit the fact $\hat{z}_{f,k} \equiv z_{f,k}$ to rewrite (4.1) as:

$$\left[\begin{array}{c} v_{k+1} \\ z_{f,k+1} \\ \hline z_{k+1} \\ \hat{v}_{k+1} \\ \hat{z}_{f,k+1} \end{array}\right] = \left[\begin{array}{cc|ccc} A_s & 0 & B_s\Theta & 0 & 0 \\ -B_f\Xi^*C_s & A_f & 0 & 0 & 0 \\ \hline 0 & 0 & A_q+\Lambda & 0 & B_qC_f \\ 0 & 0 & B_s\Theta & A_s & 0 \\ 0 & 0 & 0 & -B_f\Xi C_s & A_f \end{array}\right] \left[\begin{array}{c} v_k \\ z_{f,k} \\ \hline z_k \\ \hat{v}_k \\ \hat{z}_{f,k} \end{array}\right] + \left[\begin{array}{c} B_s \\ 0 \\ \hline 0 \\ B_s \\ 0 \end{array}\right] d_k. \tag{4.2}$$

Observe now that the block diagonal matrix

$$\begin{bmatrix} A_s & 0 \\ -B_f\Xi^*C_s & A_f \end{bmatrix} \tag{4.3}$$

is stable by construction. Moreover, the matrix

$$\begin{bmatrix} A_q+\Lambda & 0 & B_qC_f \\ B_s\Theta & A_s & 0 \\ 0 & -B_f\Xi C_s & A_f \end{bmatrix} \tag{4.4}$$

is stable by virtue of the design with $(\Theta, \Lambda) = \square(\Xi)$. It follows thus from equation (4.2) that z_k, $\hat{v}_k$, $\hat{z}_{f,k}$, v_k and $z_{f,k}$ are all bounded. More importantly, from the observable signals point of view, it appears that the control objective has been achieved for the closed-loop system (2.7) with (3.1) or (3.2). Indeed, the closed-loop stability and performance hinges on the eigenvalues of the matrices:

- A_s, the unmodeled dynamics, which are outside the control bandwidth
- A_f, the filter, free for the designer to choose, as long as it is stable
- $A + HC$, the observer eigenvalues for the nominal control design
- $A + BF$, the controlled nominal plant, and
- (4.4), the controlled model for the plant-model mismatch system, which are the poles of the closed loop (Q, S).

The above observation is independent of the nature of d_k, and would even be true for $d_k \equiv 0$. However, if d_k is sufficiently rich in that it satisfies a condition like Assumption 6.4.2, then we have the additional result that the only steady state parameters are the true system parameters, that is, $\Xi = \Xi^*$ and $(\Theta, \Lambda) = \square(\Xi^*) = (\Theta^*, \Lambda^*)$. Actually, we desire that the spectrum of d_k contains at least as many distinct frequency lines as there are parameters in Ξ to identify. This follows from the following construction. Introduce $\tilde{v}_k = v_k - \tilde{v}_k$ and $\tilde{z}_{f,k} = z_{f,k} - \hat{z}_{f,k}$. Then from (4.1) we have:

$$\begin{aligned} \tilde{v}_{k+1} &= A_s \tilde{v}_k \\ \tilde{z}_{f,k+1} &= A_f \tilde{z}_{f,k} - B_t(\Xi^* - \Xi) C_s \hat{v}_k - B_f \Xi^* \tilde{v}_k. \end{aligned}$$

Also,

$$C_s \hat{v}_k = \begin{pmatrix} 0 & C_s & 0 \end{pmatrix} \left(zI - \begin{bmatrix} A_q + \Lambda & 0 & B_q C_f \\ B_s \Theta & A_s & 0 \\ 0 & -B_q \Xi C_s & A_f \end{bmatrix} \right)^{-1} \begin{bmatrix} 0 \\ B_s \\ 0 \end{bmatrix} d_k$$

from which it follows that $\tilde{z}_{f,k} \equiv 0$ can only occur when $\Xi^* = \Xi$.

We summarize our observations as follows:

Theorem 4.1 (Tuning Property). *Consider the adaptive systems described by either* (2.7) *with the algorithm* (3.1) *or adaptive algorithm* (3.2). *Let the external disturbances be zero* ($w_k \equiv 0$). *When the tuning error* $\hat{z}_{f,k} - z_{f,k}$ *is identically zero, the algorithm's stationary points* ($\Xi_k \equiv \Xi$, $(\Theta_k, \Lambda_k) \equiv \square(\Xi)$) *are such that the closed-loop system is stable and the desired control objective is realized.*

Theorem 4.2 (Tuning property with excitation). *Consider the adaptive systems described by either* (2.7) *with algorithm* (3.1) *or* (3.2). *Let the external disturbances be zero* ($w_k \equiv 0$). *Let the probing signal be sufficiently rich in that the spectrum of* d_k *contains as many distinct frequency lines as there are elements in* Ξ. *The algorithm's stationary point is unique.* $\Xi_k \equiv \Xi^*$ *and the desired control objective is realized.*

Remark. The difference between Theorem 4.1 and Theorem 4.2 is significant. Polderman (1989) shows that only in the case of pole placement one can conclusively infer from Theorem 4.1 that the control achieved in the adaptive algorithm equals the control one would have implemented if the system were completely known. In the case of LQ control, the achieved LQ performance is only optimal for the model, that is optimal for Ξ, not for the plant Ξ^*. Due to lack of excitation we are unable to observe this in the adaptively controlled loop. However, in the presence of excitation, due to the correct identification of Ξ^* asymptotically optimal performance is obtained, for any control design. This goes a long way in convincing us why excitation, via the probing signal d_k, is indeed important in an adaptive context. It is one of the main motivations for preferring the filtered excitation algorithm above the classical algorithm. Further motivation will emerge in the subsequent analysis.

Transient Analysis: Ideal Case

Exploiting standard results in adaptive control one can show that the classical algorithm (Algorithm 2), under the assumptions

- the plant belongs to the model class $(C_\Delta = 0)$
- the external disturbances are zero $(w_k \equiv 0)$
- along the solutions of the adaptive algorithm $\Box$ is well defined

indeed realizes the desired control objective in the limit. Moreover, if the probing signal is sufficiently rich, the actual plant will be correctly identified. The interested reader is referred to Mareels and Polderman (1996), Chapter 4, for a complete proof.

From a control performance perspective, which is the topic of this book of course, this result is however not very informative. Indeed the classical adaptive control results do not make any statements about important questions such as:

- How long do the transients take?
- How large is a bounded signal?

Indeed, a moment of reflection indicates that a general result can not make any statements about problems of the above nature. A result valid for (almost) all possible initial conditions must allow for completely destabilizing controllers. For such cases it is not possible to limit either the size of the signals encountered in a transient nor the time it takes to reach the asymptotic performance. In general this situation is aggravated by imposing the condition of slow adaptation. However, in the present situation, we can avoid the above disappointments, because we start from the premise that the nominal controller, however unsatisfactory its performance, is capable of stabilizing the actual plant. We proceed therefore with Algorithm 1 (Filtered excitation), exploiting explicitly the fact that our initial information suffices to stabilize the system. By injecting a sufficiently rich probing signal, which is conveniently filtered, we show that the adaptation improves (slowly) our information about the actual plant to be controlled, hence improving our ability to control it. We regard this control strategy as one where we exploit the robustness margin of a robust stabilizing controller to such an extent that we learn the plant to be controlled in such a way as to improve the control performance. The existence of a robustness margin is crucial throughout the adaptation process. The more robust the controller, it turns out, the easier the adaptation process becomes. The algorithm is clearly achieving a successful symbiosis of robust control and adaptive control. Averaging techniques are exploited to establish the above results.

Let us be explicit about our stabilization premise:

Hypothesis 4.3. *Along the sequence of estimates Ξ_k, $k = 0, 1, \ldots$, the design rule $\square$ is such that $(\Theta_k, \Lambda_k) = \square(\Xi_k)$ is a stabilizing controller for the actual plant to be controlled.*

In the ideal scenario, the validity of this hypothesis is based on the following observations.

Introduce $\tilde{v}_k = v_k - \hat{v}_k$, $\tilde{z}_{f,k} = z_{f,k} - \hat{z}_{f,k}$ and $\tilde{\Xi}_k = \Xi_k - \Xi^*$. Along the solutions of the adaptive algorithms we have then, up to terms in w_k:

$$\left[\begin{array}{c} \tilde{v}_{k+1} \\ \tilde{z}_{f,k+1} \\ \hline z_{k+1} \\ \hat{v}_{k+1} \\ \hat{z}_{f,k+1} \end{array}\right] = \left[\begin{array}{cc|ccc} A_s & 0 & 0 & 0 & 0 \\ -B_f \Xi^* & A_f & 0 & B_f \tilde{\Xi}_k C_s & 0 \\ \hline 0 & B_q C_f & A_q + \Lambda_k & 0 & B_q C_f \\ 0 & 0 & B_s \Theta_k & A_s & 0 \\ 0 & 0 & 0 & -B_f \Xi_k C_s & A_f \end{array}\right] \begin{bmatrix} \tilde{v}_k \\ \tilde{z}_{f,k} \\ z_k \\ \hat{v}_k \\ \hat{z}_{f,k} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ B_s \\ 0 \end{bmatrix} d_k. \tag{4.5}$$

By construction we have that the matrices

$$\begin{bmatrix} A_s & 0 \\ -B_f \Xi^* & A_s \end{bmatrix} \tag{4.6}$$

and

$$\begin{bmatrix} A_q + \Lambda_k & 0 & B_q C_f \\ B_s \Theta_k & A_s & 0 \\ 0 & -B_f \Xi_k C_s & A_s \end{bmatrix} \tag{4.7}$$

are stable (in that the eigenvalues for each instant k are less than 1 in modulus). Hence, provided Ξ_k is slowly time varying in that $\| \Xi_{k+1} - \Xi_k \|$ is sufficiently small and $\tilde{\Xi}_k$ is sufficiently small the overall system will be stable. As we will show $\| \tilde{\Xi}_k \|$ is monotonically nonincreasing, and $\| \Xi_{k+1} - \Xi_k \|$ is governed by μ.

Hence, assuming $\| \tilde{\Xi}_k \|$ is sufficiently small, and μ is sufficiently small, we have that Hypothesis 4.3 is satisfied along the solutions of the adaptive system.

In order to see that $\| \tilde{\Xi}_k \|$ is decreasing, we proceed as follows. Introduce,

$$\lambda_{ij,k+1} = A_f \lambda_{ij,k} - B_f E_{ij} C_s h_k, \tag{4.8}$$

$$h_{k+1} = A_s h_k + B_s \Theta_k z_k. \tag{4.9}$$

We have then, comparing (2.7) with (4.9) under the condition that $C_\Delta = 0$, that

$$z_{f,k} = \sum_{ij} \gamma_{ij,k} \Xi^*_{ij} + \sum_{ij} \lambda_{ij,k} \Xi^*_{ij}, \tag{4.10}$$

and also

$$\hat{z}_{f,k} = \sum_{ij} \gamma_{ij,k} \Xi_{ij,k} + \sum_{ij} \lambda_{ij,k} \Xi_{ij,k} + O(\mu). \tag{4.11}$$

Here $O(\mu)$ stands for a term f_k that can be over bounded as $|f_k| \leq K_1\mu$ for some $K_1 > 0$ (see also Appendix C). Moreover, because the filter (A_f, B_f, C_f) is designed such that it eliminates the spectrum of d_k, and because we assume that the driving signals w_k are orthogonal to d_k, it follows that, for constant Θ:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=m+1}^{N+m} \gamma_{ij,k}\lambda_{ij,k-m} = 0 \quad \text{for all } m = 0, 1, 2, \ldots \quad \text{and all } i, j.$$

The above expression embodies the most important design difficulty, on the one side d_k must be sufficiently rich to lead to the identification of Ξ^*, but we also need to filter it out of the Q loop via (A_f, B_f, C_f), complicating the controller design.

For the adaptive update equation, see (3.1), we find thus after substituting (4.10) and (4.11):

$$\Xi_{ij,k+1} = \Xi_{ij,k} - \mu\gamma_{ij,k} \sum_{\ell,t} \left(\gamma_{\ell t,k} + \lambda_{\ell t,k}\right)\left(\Xi^*_{\ell t} - \Xi_{\ell t,k}\right) + O(\mu^2)$$

$$\text{for all } i, j; k. \tag{4.12}$$

This equation (4.12) is in the standard form to apply averaging theory, see Appendix C. Using the results from Appendix C we find for the averaged equation

$$\operatorname{vec}\left(\Xi^{av}_{k+1}\right) = \operatorname{vec}\left(\Xi^{av}_{k}\right) - \mu\Gamma \operatorname{vec}\left(\Xi^{av}_{k} - \Xi^*\right) \tag{4.13}$$

where the matrix Γ contains as elements

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \gamma'_{ij,k}\gamma_{\ell t,k}$$

in appropriate order. For sufficiently rich d_k, the matrix Γ is positive definite, $\Gamma = \Gamma' > 0$. It follows that Ξ^{av}_k converges exponentially to Ξ^*, for sufficiently small μ and for sufficiently rich d_k. Theorem C.4.2 informs us that,

$$\lim_{\mu\to 0} \left\| \Xi_k - \Xi^{av}_k \right\| = 0.$$

Because d_k has a finite spectrum, we are able to obtain the stronger result $\left\| \Xi_k - \Xi^{av}_k \right\| = O(\mu)$, for all k.

Hence for sufficiently small μ, optimal control is achieved up to small errors of the order of the adaptation step size, without losing the stability property of the initial stabilizing controller. We summarize:

Theorem 4.4 (Ideal case, filtered excitation). *Consider the adaptive system* (2.7) *with* (3.1) *under the condition that the plant belongs to the model class* $C_\Delta = 0$*. Let Assumption 3.1 be satisfied. Let d have a finite spectrum, but sufficiently rich as to enforce unique identifiability of Ξ^* (d's spectrum contains as many distinct frequency lines as there are elements in Ξ^*). Let (A_f, B_f, C_f) be chosen as to null the spectrum of d* (2.5)*. Let w_k be uncorrelated with d_k. Then there exists a $\mu^* > 0$, such that for all $\mu \in (0, \mu^*)$ and for all initial conditions such that Ξ_0 leads to a stable closed loop, we have that:*

1. *The adaptive system state is bounded, more precisely there exists constants $C_1 > C_2,\ C_3 > 0$ and $0 < \lambda < 1$ such that*

$$\|X_k\| \le C_1\lambda^k \|X_0\| + C_2 \|d\| + C_3 \|w\|,$$
$$X_k = \left(v_k', \hat{v}_k', \hat{z}'_{f,k}, z_k', \hat{x}_k', x_k'\right)',$$

   and

$$\left\| \Xi_k - \Xi^* \right\| \le \left\| \Xi_0 - \Xi^* \right\| + O(\mu).$$

2. *Exponentially fast optimal control performance is achieved, in that there exists positive constants $C_4, C_5, C_6 > 0$ independent of μ, with $0 < 1 - \mu^* C_4 < 1$, such that:*

$$\left\| \Xi_k - \Xi^* \right\| \le C_6(1 - C_4\mu)^k \left\| \Xi_0 - \Xi^* \right\| + C_5\mu.$$

Remark.

The above result is applicable to Problems 1 and 2 of Section 7.2. We stress that it is important that $\square(\Xi)$ is Lipschitz continuous in Ξ, otherwise the averaging result is not valid. (See Theorem C.2.2). This Lipschitz continuity may be guaranteed via Assumption 3.1. Indeed, under Assumption 3.1, $\square$ defined via either pole-placement or LQ control can be constructed to be Lipschitz continuous along the estimates generated by the adaptive algorithm.

More importantly, the same result would also apply to the whole class of adaptive algorithms $(\Theta_k, \Lambda_k) = \square_k (\Xi_k)$, such that Hypothesis 4.3 is satisfied. For pole placement and LQ control, this has been verified, but a range of alternatives is conceivable. Most noteworthy is the earlier suggestion to have $\square(\Xi_k) = 0$ for all $k = 0, 1, \ldots, k_0$, that is, wait to update the controller until Ξ_{k_0} is such that controller design via LQ or pole placement does not destroy the robustness margin of the initial controller K. Also, $\square_k$ could reflect our desire to improve the closed-loop control performance as time progresses by, for example, increasing the bandwidth of the controlled loop.

Finally, we point out how $\square$ in the case of LQ control may be constructed. We consider the LQ index (see Problem 2 of Section 7.2):

$$J(\square) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \left(s_k' \square s_k + r_k' r_k\right); \qquad \square = \square' > 0 \tag{4.14}$$

To construct □ let us assume that the matrix pair (2.9) is stabilizable and the matrix pair (2.10) is detectable. Denote

$$\square = \begin{pmatrix} A_s & 0 \\ -B_f \Xi C_s & A_f \end{pmatrix}, \tag{4.15}$$

$$\square = \begin{pmatrix} B_s \\ 0 \end{pmatrix}, \tag{4.16}$$

$$\square = \begin{pmatrix} 0 & B_q C_f \end{pmatrix}. \tag{4.17}$$

Under the stated conditions, (□, □) stabilizable and (□, □) detectable, we can solve the following Riccati equations for unique positive definite □ and □:

$$\square = \square\square\square' + I - \left(\square\square\square'\right)\left(C\square\square' + I\right)^{-1}\square\square\square' \tag{4.18}$$

and

$$\square = \square'\square\square + \square - \left(\square'\square\square\right)\left(\square'\square\square + \square\right)^{-1}\square'\square\square. \tag{4.19}$$

Here I and □ are the weighting matrices from the LQ index (4.14).

Given □ and □ we construct

$$\square = \left(\square\square\square'\right)\left(C\square\square' + R\right)^{-1} \tag{4.20}$$

and

$$\square = \left(\square'\square\square + S\right)^{-1}\left(\square'\square\square\right), \tag{4.21}$$

which have, respectively, the property that □ − □□ and □ − □□ are stable matrices. The controller that solves the optimization of the index J of (4.14) can then be implemented in the standard way (see Chapter 2), with observer gain □ and feedback gain □. In particular, we have

$$\Lambda = \begin{pmatrix} 0 & 0 \\ -B_f \Xi C_s & 0 \end{pmatrix} - \square\square \tag{4.22}$$

and

$$\Theta = -\square. \tag{4.23}$$

Because □ is an affine function of Ξ it follows that □, □ and hence also □, □, and Λ, Θ depend on Ξ. Moreover it can be demonstrated that on any open subset of detectable matrix pairs $(\square, \square(\Xi))$ and on any open subset of stabilizable matrix pairs $(\square(\Xi), \square)$ this dependency on Ξ is analytic. (See Mareels and Polderman (1996) for a proof of the analyticity property.)

Main Points of Section

The behavior of the adaptive closed-loop system is studied in the situation that the parameterized class of models contains the particular plant, the so-called ideal situation. First attention is paid to the possible no-adaptation behaviors. The indirect adaptive algorithms introduced involving either filtered excitation or using classical adaptation, both enjoy the tuning property. The tuning property indicates that under conditions that the identification error is zero, the adaptively controlled closed loop appears to be controlled as if the true system is known. This establishes that the steady state behavior of the controlled loop is as good as we can hope for. Next we consider the transient behavior. We establish that the filtered excitation adaptive algorithm in conjunction with a sufficiently rich probing signal is capable of identifying the plant and hence achieves near optimal control performance. The result holds under the condition of sufficiently slow adaptation. The key idea in the filtered excitation algorithm is that as far as the probing signal is concerned the loop is essentially open. This achieves unbiased identification and allows one to exploit the robustness margin of the stabilizing controller without compromising the performance. Robust control and adaptation are truly complementary.

7.5 Adaptive Algorithm Analysis: Nonideal Case

The most important realization is that in our set up the tuning property remains valid in the presence of unmodeled dynamics for either the classical or the filtered excitation adaptive algorithm. This is a consequence of the fact that we start from the premise that our initial controller is stabilizing. It is in sharp contrast with the classical position in adaptive control, where arbitrarily small perturbations may lead to instability, destroying the tuning property. Moreover, if we restrict ourselves to the filtered excitation algorithm, we will deduce that in spite of modeling errors, we do identify the best possible model in our model class to represent S. This property does not hold for the classic adaptive algorithm due to the highly nonlinear interaction between identification and control. As a consequence the filtered excitation algorithm has a much larger robustness margin for inadequate modeling of the S parameter than the classic algorithm. The main result is therefore restricted to the filtered excitation algorithm.

Tuning Property

Although we argue that the tuning property is close to necessary for an adaptive algorithm, it is clear that in the nonideal case this property plays a less important role. This is because the condition $\hat{z}_{f,k} - z_{f,k} \equiv 0$ can hardly be expected to be satisfied along the solutions of the adaptive system. Following the same arguments as in the ideal case we find nevertheless that the tuning property also holds in the nonideal case. Indeed, the stability of the closed-loop adaptive system hinges on the stability of the system (neglecting the driving signals w_k):

$$\begin{pmatrix} v_{\Delta,k+1} \\ v_{k+1} \\ z_{f,k+1} \\ z_{k+1} \\ \hat{v}_{k+1} \\ \hat{z}_{f,k+1} \end{pmatrix} = \begin{pmatrix} A_\Delta & 0 & 0 & B_\Delta \Theta & 0 & 0 \\ 0 & A_s & 0 & B_s \Theta & 0 & 0 \\ -B_f C_\Delta & -B_f \Xi^* C_s & A_f & 0 & 0 & 0 \\ 0 & 0 & B_q C_f & A_q + \Lambda & 0 & 0 \\ 0 & 0 & 0 & B_s \Theta & A_s & 0 \\ 0 & 0 & 0 & 0 & -B_f \Xi C_s & A_f \end{pmatrix} \begin{pmatrix} v_{\Delta,k} \\ v_k \\ z_{f,k} \\ z_k \\ \hat{v}_k \\ \hat{z}_{f,k} \end{pmatrix} + \begin{pmatrix} B_\Delta \\ B_s \\ 0 \\ 0 \\ B_s \\ 0 \end{pmatrix} d_k. \tag{5.1}$$

Again using $\hat{z}_{f,k} \equiv z_{f,k}$, it follows that the stability of the loop depends only on the stability of the matrix

$$\begin{pmatrix} A_\Delta & 0 & 0 \\ 0 & A_s & 0 \\ -B_f C_\Delta & -B_f \Xi^* C_s & A_f \end{pmatrix}$$

and the design matrix

$$\begin{pmatrix} A_q + \Lambda & 0 & B_q C_f \\ B_s \Theta & A_s & 0 \\ 0 & -B_q \Xi C_s & A_f \end{pmatrix}.$$

The former is stable by assumption, the latter by construction. This establishes the tuning property.

Expression (5.1), however makes it very clear that due to the presence of $v_{\Delta,k}$, and certainly in the case of a sufficiently rich d_k, we can not expect to have $z_{f,k} \equiv \hat{z}_{f,k}$. In the absence of any probing signal d_k and with no external disturbances $w_k \equiv 0$, it is possible to have $z_{f,k} \equiv \hat{z}_{f,k}$.

Identification Behavior

Let us now focus on what model will be identified in closed loop via the filtered excitation adaptive algorithm. Again, we rely on slow adaptation and the stability Hypothesis 4.3. Whereas in the ideal case, it is clear that Hypothesis 4.3 is fulfilled under very reasonable assumptions, this is no longer guaranteed in the nonideal model case. We postpone a discussion of this crucial hypothesis until we have a clearer picture of what the possible stationary points for the identification algorithm are.

Clearly, as before we have (see (4.11))

$$\hat{z}_{f,k} = \sum_{ij} \gamma_{ij,k} \Xi_{ij,k} + \sum_{ij} \lambda_{ij,k} \Xi_{ij,k} + O(\mu).$$

However

$$z_{f,k} = \sum_{ij} \left(\gamma_{ij,k} \Xi^*_{ij} + \lambda_{ij,k} \Xi^*_{ij} \right) + \nu_{1,k} + \nu_{2,k},$$

where

$$\begin{aligned}
\nu_{1,k+1} &= A_f \nu_{1,k} - B_f C_\Delta v_{1,k}, \\
v_{1,k+1} &= A_\Delta v_{1,k} + B_\Delta d_k, \\
\nu_{2,k+1} &= A_f \nu_{2,k} - B_f C_\Delta v_{2,k}, \\
v_{2,k+1} &= A_\Delta v_{2,k} + B_\Delta \Theta_k z_k.
\end{aligned}$$

The adaptive update equation can thus be written as:

$$\begin{aligned}
&\Xi_{ij,k+1} \\
&= \Xi_{ij,k} - \mu \gamma'_{ij,k} \left(\sum_{t,\ell} \left(\gamma_{t\ell,k} + \lambda_{t\ell,k}\right) \left(\Xi_{t\ell,k} - \Xi^*_{t\ell}\right) + \left(\nu_{1,k} + \nu_{2,k}\right) \right) + O(\mu^2).
\end{aligned} \tag{5.2}$$

In the above expressions $\lambda_{t\ell,k}$ and $\nu_{2,k}$ are functions of Ξ_k, but as observed before, for fixed Ξ we clearly have

$$\begin{aligned}
\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \gamma_{ij,k} \nu'_{2,k}(\Xi) &= 0, \\
\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \gamma_{ij,k} \lambda'_{t\ell,k}(\Xi) &= 0,
\end{aligned}$$

because both ν_2 and $\lambda_{t\ell}$ are filtered versions of the signal z which does not contain the spectrum of d, as this is eliminated by the filter (A_f, B_f, C_f). Computing the averaged equation for (5.2), we have thus:

$$\operatorname{vec}\left(\Xi^{av}_{k+1}\right) = \operatorname{vec}\left(\Xi^{av}_k\right)_k - \mu \Gamma \operatorname{vec}\left(\Xi^{av}_k - \Xi^*\right) + \mu M,$$

where M consists of the elements

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \gamma'_{ij,k} \nu_{1,k},$$

in appropriate order. If follows that the averaged equation, under persistency of excitation such that $\Gamma = \Gamma' > 0$, has a unique equilibrium:

$$\operatorname{vec}\left(\Xi^{av}_\infty\right) = \operatorname{vec}\left(\Xi^*\right) + \Gamma^{-1} M.$$

Reinterpreting the above averages in the frequency domain, we see that the identification process is equivalent to finding the best Ξ parameter in an ℓ_2 approximation sense, that is

$$\Xi^{av}_\infty = \arg\min \left\| \left(S(z) - \Xi B(z)\right) d \right\|_2,$$

where

$$S(z) = C_\Delta \left(zI - A_\Delta\right)^{-1} B_\Delta + \Xi^* B\left(z\right).$$

This is clearly the best we can achieve in the present setting, but unfortunately it is not directly helpful in a control setting. We discuss this in the next subsection. Let us now summarize our main result thus far:

Theorem 5.1. *Consider the adaptive system* (2.7) *together with* (3.1). *Let Assumption 3.1 and Hypothesis 4.3 hold. Assume that the external probing signal d is sufficiently exciting in that $\|S\left(z\right) - \Xi B\left(z\right) d\|_2$ has a unique minimizer. Moreover the filter (A_f, B_f, C_f) nulls the signal d. Assume that the external signal w is stationary and has a spectrum which does not overlap with that of d in that:*

$$\lim_{k\to\infty} \frac{1}{N} \sum_{k=1}^{N} w_k d_k = 0.$$

Then for all initial conditions satisfying Hypothesis 4.3 in a compact domain there exists a $\mu^* > 0$ such that for all $\mu \in (0, \mu^*)$ the adaptively controlled loop is stable, in that all signals remain bounded. Moreover,

1. $$\left\| \Xi_k - \Xi_k^{av} \right\| = \delta(\mu),$$

2. $$\limsup_{k\to\infty} \left\| \Xi_k - \Xi_\infty^{av} \right\| = \delta(\mu),$$

for some order function $\delta(\mu)$ such that $\lim_{\mu\to 0} \delta(\mu) = 0$.

The main difficulty is of course Hypothesis 4.3 which we discuss now.

Identification for Control

As indicated earlier, the asymptotic performance of the adaptive algorithm is governed by:

$$\Xi_\infty = \arg\min \|\left(S(z) - \Xi B(z)\right) d\|_2 .$$

The corresponding closed-loop stability depends on

$$\left(\Theta_\infty, \Lambda_\infty\right) = \square\left(\Xi_\infty\right),$$

where by construction $\square$ ensures that the matrix

$$\begin{pmatrix} A_s & 0 & B_s\Theta_\infty \\ -B_f\,\Xi_\infty C_s & A_f & 0 \\ 0 & B_q C_f & A_q + \Lambda_\infty \end{pmatrix} \tag{5.3}$$

is a stable matrix. The closed-loop stability of the system is however determined by the stability properties of the matrix

$$\begin{pmatrix} A_\Delta & 0 & 0 & B_\Delta\Theta_\infty \\ 0 & A_s & 0 & B_s\Theta_\infty \\ -B_f C_\Delta & -B_f\Xi^* C_s & A_f & 0 \\ 0 & 0 & B_q C_f & A_q + \Lambda_\infty \end{pmatrix}. \tag{5.4}$$

Let us interpret these in terms of transfer functions. Introduce

$$\begin{aligned} S_\infty(z) &= \Xi_\infty B(z), \\ S(z) &= C_\Delta \left(zI - A_\Delta\right)^{-1} B_\Delta + \Xi B(z), \\ Q(z) &= \Theta_\infty \left(zI - A_q - \Lambda_\infty\right)^{-1} B_q C_f \left(zI - A_f\right)^{-1} B_f, \\ \Delta(z) &= S(z) - S_\infty(z). \end{aligned}$$

Thus (5.3) being a stable matrix states that $(S_\infty(z), Q(z))$ is a stable loop, and (5.4) stable expresses that $(S(z), Q(z))$ is a stabilizing pair. A sufficient condition for the latter is that:

$$\left\| \left(I - Q(z)S_\infty(z)\right)^{-1} \Delta(z) \right\|_\infty < 1. \tag{5.5}$$

Via the adaptive algorithm we have ensured that

$$\|\Delta(z)d\|_2^2 < 1, \tag{5.6}$$

which does go a long way in establishing (5.5) but is not quite enough. It is indeed possible that the minimization of (5.6) does not yield (5.5), and may even lead to instability in the closed-loop adaptive system. This indicates that the adaptive algorithm leads to unacceptable behavior. A finite-time averaging result (see Appendix C), allows us to conclude that the adaptive algorithm will indeed try to identify $S_\infty(z)$. This leads to a temporarily unstable closed loop, characterized by exploding signals. At this point averaging would no longer be valid, Hypothesis 4.3 being violated. But it does indicate that large signals in the loop are to be expected. Invariably the performance of such a controlled system is bad, even if the adaptive loop may recover from this explosive situation. Understanding what type of behavior ensues from this is nontrivial. For a discussion of the difficulties one may encounter we refer to Mareels and Polderman (1996, Chapter 9). Suffice it to say that chaotic dynamics and instability phenomena belong to the possibilities.

In order to have that the minimization of (5.6) leads to (5.5) being satisfied, we should have either

1. a sufficiently general model to ensure that C_Δ will be small, or

2. ensure that outside the frequency spectrum of d the controlled loop $(S_\infty(z), Q(z))$ has small gain.

The link between identification and control is obvious in the equations (5.5) and (5.6) and has been the focus of much research. See, for example Partanen (1995), Lee (1994), Gevers (1993).

Main Points of Section

In the nonideal case, the model class is insufficient to describe the mismatch between the plant and the nominal plant; the filtered excitation adaptive algorithm attempts to identify the best possible model in an ℓ_2 sense. Unfortunately, this may not be enough to guarantee stability let alone performance. Indeed despite the fact that the initial model and controller is stable it may be that the best possible model in the model class leads to instability. The interdependency of identification and control is clearly identified. The key design variables are the choice of model class, the probing signal and the control objective, as exhibited in equations (5.6) and (5.5).

Example. First we demonstrate the idea behind the filtered excitation adaptive algorithm, without using the (Q, S) framework. The example will deviate slightly from the theory developed in this chapter in order to illustrate the flexibility offered by the averaging analysis. The example is an abstraction of a problem encountered in the control of the profile of rolled steel products.

Consider the plant represented in Figure 5.1. The input u and output e are measurable. The control objective is to regulate e to zero as fast as possible. The signal w is an unknown constant. The plant output y is not measurable. The gain $g \in (0, \bar{g})$, is unknown but $\bar{g}$ is a known constant.

The proposed solution, in the light of the filtered excitation adaptive algorithm, is to use a probing signal $d_k = (-1)^k d$, where d is a small constant, leading to an acceptable error in the regulation objective. The controlled plant becomes as presented in Figure 5.2.

When $\hat{g} = g$, then we have dead beat response and e is regulated in three time steps. More precisely, with q^{-1} the unit delay operator

$$q^3 e = (q-1)(q-\tfrac{1}{2})(q+\tfrac{1}{3})w + g(q-\tfrac{1}{2})(q+\tfrac{1}{3})(-1)^k d.$$

The probing signal leads thus to a steady state error of $|gd/2|$ in magnitude. Of course, due to the integral in the plant, there is no steady state error for any constant w.

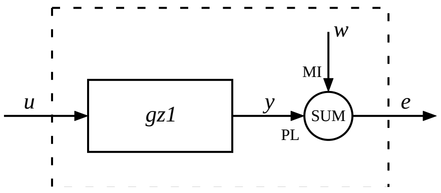
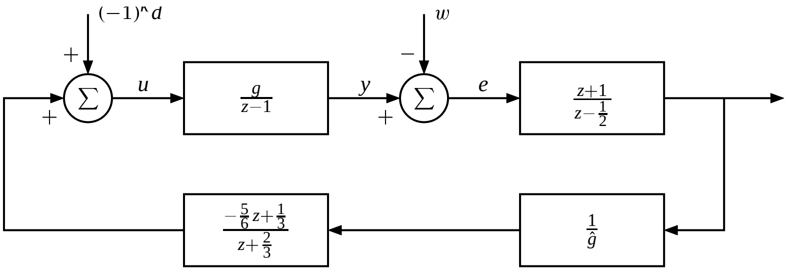


FIGURE 5.1. Plant



$(-1)^{\wedge} d$
w
$+$
$-$
u
y
e
$+$
$+$
$\frac{g}{z-1}$
$\frac{z+1}{z-\frac{1}{2}}$
$\frac{-\frac{5}{6}z+\frac{1}{3}}{z+\frac{2}{3}}$
$\frac{1}{\hat{g}}$

FIGURE 5.2. Controlled loop

It is easily verified that the above system is stable for all $g/\hat{g} \in (0, 2)$. It is thus advantageous to over estimate g. The filtered excitation adaptive algorithm can be implemented as follows:

$$\hat{g}_{k+1} = \hat{g}_k - \mu(-1)^k \left(e_k + \frac{\hat{g}_k d}{2}(-1)^k \right); \qquad \hat{g}_0 = \bar{g}.$$

The complete control loop is illustrated in Figure 5.3.

Indeed because of the filter we expect the steady state behavior of e_k due to the probing signal to be $-(gd/2)(-1)^k$, our estimate for this is $-(\hat{g}_k d/2)(-1)^k$, which leads to the above update law. Now provided $g/\hat{g}_k \in (0, 2)$ and for sufficiently small μ, we can look at the averaged update equation to see how the adaptive system is going to respond. According to our development this leads to

$$\hat{g}^{av}_{k+1} = \hat{g}^{av}_k - \mu \left(-\frac{gd}{2} + \frac{\hat{g}^{av}_k d}{2} \right), \qquad \hat{g}^{av}_0 = \hat{g}_0 = \bar{g},$$

or

$$\hat{g}^{av}_{k+1} = \hat{g}^{av}_k - \frac{\mu d}{2} \left(\hat{g}^{av}_k - g \right).$$

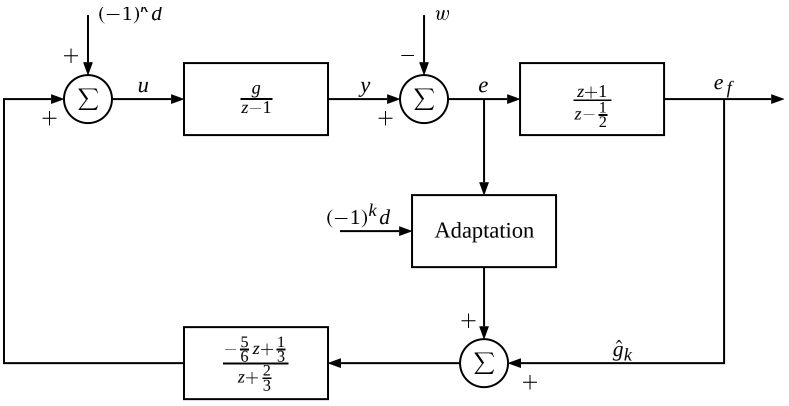


FIGURE 5.3. Adaptive control loop

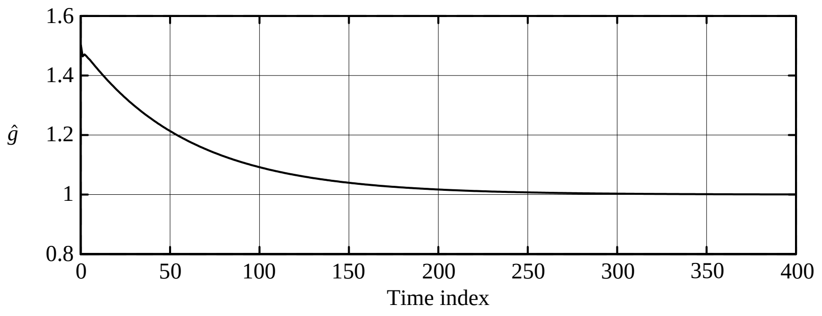


FIGURE 5.4. Response of $\hat{g}$

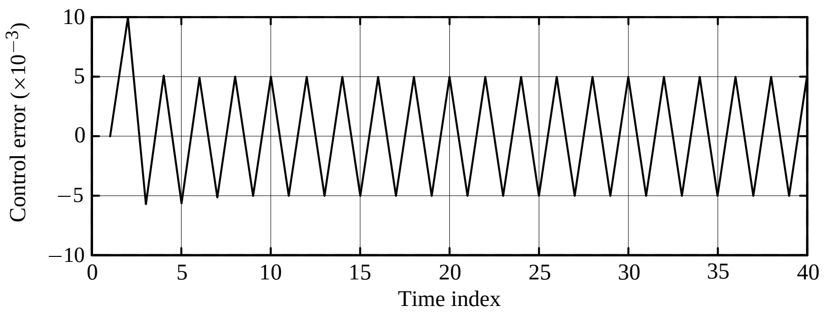


FIGURE 5.5. Response of e

Hence, as expected $\hat{g}_k^{av}$ converges monotonically to g whenever $0 < \mu d < 2$. Now from Theorem 4.4 we conclude that for all $g/\hat{g}_0 \in (0, 2)$

$$\left|\hat{g}_k - \hat{g}_k^{av}\right| = O(\mu) \quad \text{for all } k.$$

This leads to asymptotically near optimal performance for all $g \in (0, \bar{g})$, actually for all $g \in (0, 2\bar{g} - \varepsilon)$, where ε is any small constant $1 \gg \varepsilon > \mu > 0$.

A response of the algorithm is illustrated in Figures 5.4 and 5.5. Figure 5.4 illustrates the response of the $\hat{g}$ variable, while Figure 5.5 displays the response of the regulated variable. For the simulation we choose $\mu d = 0.2$, and all other initial conditions set to 1. Notice that the averaging approximation predicts the closed-loop behavior extremely well.

As can be seen from this example, stability of the plant to be controlled is not essential for the filtered excitation algorithm to be applied. The stability of the closed loop is, of course, important.

Simulation Results

In this subsection, we present simulations for the case where an explicit adaptive LQG algorithm is used to design an adaptive Q filter, denoted Q_k. The actual plant $G(S)$, and the nominal controller K used are designed based on the nominal plant G of the example in Section 5.2. The following LQ index penalizing r and s is used in the design of the adaptive LQG augmentation Q_k.

$$J_{LQ} = \lim_{k\to\infty} \sum_{i=1}^{k} (r_i^2 + s_i^2). \tag{5.7}$$

Table 5.1 shows a performance index comparison for the following various cases. The first is where the actual plant $G(S)$ is controlled by the LQG controller K for the nominal plant G with no adaptive augmentation. The second case is when the LQG controller for G is augmented with an indirect LQG adaptive-Q algorithm. A third order model S is assumed in this case, and the estimate of S 'converges' to

$$S = \frac{0.381z^{-1} - 0.092\,5z^{-2} - 0.358\,8z^{-3}}{1 - 0.423\,9z^{-1} - 0.439\,2z^{-2} - 0.018\,1z^{-3}}. \tag{5.8}$$

A marked improvement over that of the nonadaptive case is recorded. Note that the average is taken after the identification algorithm 'converges'. The third case is for the actual plant, $G(S)$, controlled by the corresponding LQG controller, designed based on knowledge of $G(S)$ rather than on that of a nominal model G. Clearly, the performance of the adaptive scheme (Case 2) is drastically better than for the nonadaptive case, and approaches that of the optimal scheme (Case 3), confirming the performance enhancement ability of the technique.

Case	$\frac{1}{k}\sum_{i=1}^{k}(y_i^2 + 0.005u_i^2)$
1. Actual plant $G(S)$ with LQG controller for nominal plant G.	0.423 0
2. Actual plant $G(S)$ with nominal LQG controller and adaptive Q_k.	0.186 0
3. Actual plant $G(S)$ with optimal LQG controller for $G(S)$.	0.175 6

TABLE 5.1. Comparison of performance

In a second simulation run, we work with a plant G which is not stabilized by the nominal controller K. Again S is identified on line using a third order model and there is employed an adaptive LQG algorithm, as in the run above, to design a Q_k to augment K. Figure 5.6 shows the plant output and input. In this instance, the adaptive augmentation, Q_k together with K stabilizes the plant. The results show that the technique not only enhances performance but also can achieve robustness enhancement.

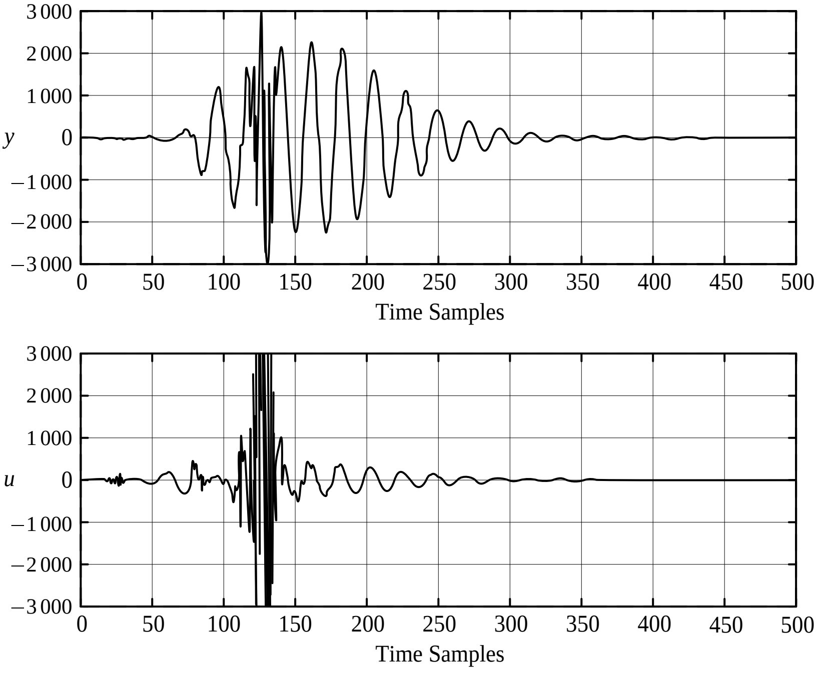


FIGURE 5.6. Plant output y and plant input u

Main Points of Section

The first example serves to illustrate the powerful nature of the averaging techniques. Clearly the analysis can be used for design purposes. The filtered excitation algorithm provides a suitable way of injecting an external signal such as to identify the plant characteristics in a particular frequency range, without compromising the control performance too much. The trade off between desired control objective and the identification requirements can easily be analyzed in the frequency domain. The second example clearly demonstrates the strength of the (Q, S) framework.

7.6 Notes and References

Indirect adaptive control has a played an important role in the development of control theory. It has been treated extensively in books such as Goodwin and Sin (1984), Sastry and Bodson (1989), Narendra and Annaswamy (1989), Mareels and Polderman (1996). Our premise here is of course the availability of a stabilizing, not necessarily well performing controller. This leads to a significant departure of the usual approach to the design of adaptive systems. It imposes a lot of structure on the adaptive problem which can be exploited to advantage. The developments of these ideas can be traced to Tay (1989) and Wang (1991). In order to achieve identification in closed loop without compromising the control too much, we introduced the filtered excitation adaptive scheme. This has not been treated before in the literature and much of the presentation here is new. The averaging ideas which are crucial to the analysis have been developed in, e.g. Mareels and Polderman (1996) and Solo and Kong (1995). Averaging has a long history in adaptive control and signal processing analysis, see for example Anderson et al. (1986).

Problems

1. Reconsider the example in the text in the (Q, S) framework. Let $G = 1/(z-1)$ with $N = 1/z$ and $M = (z-1)/z$. Let $K = 1$ with $U = V = 1$. Show that $S = (z-1)(g-1)/[z(z+(g-1))]$. Use as parameterized function class $B(z, \Xi) = ((z-1)/z)(\Xi/(z+\Xi))$ with $\Xi \in (-1, 1)$. Use the same probing signal as in the text $d(k) = (-1)^k d$. Consider the plug in controller to take the form $Q(z) = q_1(z+1)(z+q_2)/(z^2+q_3z+q_4)$. Achieve dead beat control for the (Q, S) loop, that is, compute $\square$. Show that $\square$ is Lipschitz continuous on the domain $\Xi \in (0, 2)$. Show that for correctly tuned Ξ this strategy achieves dead beat control for the complete closed loop. Now implement the filtered excitation algorithm.

   Remark. Observe that this control strategy is more complicated than the direct implementation discussed in the text. The advantage of the (Q, S) framework is that it leads to a more robust control loop with respect to other disturbances.

2. Using the example in Section 7.5, explore the effect of using other probing signals of the form $d(k) = d\cos(\omega k)$ with appropriate filter in the Q-loop. Show that as ω becomes smaller it becomes increasingly difficult to achieve good regulation behavior. Why is this?

3. The adaptive control algorithm, like all indirect adaptive control algorithms has no global stability property. In particular, as indicated, the filtered excitation algorithm may fail when the Hypothesis 4.3 fails. This may be illustrated using the example in Section 7.5 by taking initial conditions as, for example, $\hat{g}_0 = 0,\ y = 10$. The stability hypothesis fails and limit cycle behavior is observed. What happens when $\hat{g}_0 \gg \bar{g}$?

CHAPTER **8**

Adaptive-Q Application to Nonlinear Systems

8.1 Introduction

For nonlinear plants there has evolved both open-loop and closed-loop optimal control theory. Optimal feedback control for very general plants and indices is very new and appealing, see Elliott et al. (1994), but requires infinite dimensional controllers designed using off-line infinite dimensional calculations. Optimal open-loop nonlinear control methods are considered very elegant in theory, but lack robustness in practice, see Sage and White (1977). Can adaptive-Q methods somehow bridge the gap between these elegant theories and result in practical feedback controllers that have most of the benefits of both approaches without the disadvantages of each? We proceed as in the work of Imae, Irlicht, Obinata and Moore (1992) and Irlicht and Moore (1991).

In the optimal open-loop control approach to nonlinear control, a nonlinear mathematical model of the process is first formulated based on the fundamental laws in operation or via identification techniques. Next, a performance index is derived which reflects the various cost factors associated with the implementation of any control signal. Then, off-line calculations lead to an optimal control law u^* via one of the various methods of optimal control, see for example Teo, Goh and Wong (1991). In theory then, applying such a control law to the physical process should result in optimal performance. However, the process is rarely modeled accurately, and frequently is subject to stochastic disturbances. Consequently, the application of the "optimal" control signal u^* results in poor performance, in that the process output y differs from y^*, the output of the idealized process model.

A standard approach to enhance open-loop optimal control performance, see for example Anderson and Moore (1989), is to work with the difference between the ideal optimal process output trajectory y^* and the actual process output y, denoted δy, and the difference δu, between the optimal control u^* for the nominal model and any actual control signal u applied. For nominal plants and performance indices with suitably smooth nonlinearities, a linearization of the process allows an approximate linear time-varying dynamic model for relating δy to δu. With this model, and an associated quadratic index also derived from the Taylor series expansion function, optimal linear quadratic feedback regulator theory can be applied to calculate δu in terms of δy which is measurable, so as to regulate δy to zero, or equivalently to force the actual plant to track closely the optimal trajectory for the nominal plant. Robust regulator designs based on optimal theory, perhaps via H_∞ or LQG/LTR, could be expected to lead to performance improvement over a wider range of perturbations on the nominal plant model.

Even with the application of linearization and feedback regulation to enhance optimal control strategies, there can still be problems with external disturbances and modeling errors. The linearization itself may be a poor approximation when there are large perturbations from the optimal trajectory.

In this chapter, it is proposed to apply the adaptive-Q techniques developed for linear systems so as to achieve high performance in the face of nonlinearities and uncertainties, that is to assist in regulation of the actual plant so that it behaves as closely as possible to the nominal (idealized) model under optimal control. Some analysis results are presented giving stability properties of the optimal/adaptive scheme, and certain relevant nonlinear coprime factorization results are summarized. Simulation results demonstrate the effectiveness of the various control strategies, and the possibility of further performance enhancement based on functional learning is developed.

8.2 Adaptive-Q Method for Nonlinear Control

In this section, we first introduce a nonlinear plant model and associated nonlinear performance index. Next, we perform a linearization, then apply feedback regulation to the linearized model to achieve a robust controller. Finally we apply the adaptive-Q algorithms to this robust controller. There is a mild generalization for dependence of the linear blocks (operators) on the desired trajectory. Since these operators are inherently time varying, the notion of time-varying coprime factorizations is developed.

Signal Model, Optimal Index and Linearization

Consider some nonlinear plant, denoted $\bar{G}$, and a generalized nonlinear nominal plant model G, an approximation for $\bar{G}$:

$$G : x_{k+1} = f(x_k, u_k), \qquad y_k = h(x_k, u_k), \tag{2.1}$$

with $f(\cdot,\cdot)$ and $h(\cdot,\cdot) \in C^1$, the class of continuously differentiable functions. Consider also some performance index over the time interval $[0, T]$

$$I\left(x_0, u_{[0,T]}\right) = \frac{1}{T}\sum_{k=0}^{T} \ell\left(x_k, u_k\right), \tag{2.2}$$

Assume that (2.2) can be minimized subject to (2.1), and that the associated optimal control is given by u^*, the optimal state trajectory is x^*, and the optimal output trajectory is y^* so that in an operator notation $y^* = Gu^*$, where G is initial condition dependent. For details on such calculations see Teo et al. (1991).

Consider now a linearized version of the above plant model driven by u^* and with states x^*, denoted ΔG^*:

$$\begin{aligned} \Delta G^* : \delta x_{k+1} &= A\delta x_k + B\delta u_k; \qquad \delta x_0 = 0, \\ \delta y_k &= C\delta x_k + D\delta u_k. \end{aligned} \tag{2.3}$$

In obvious notation,

$$(A, B, C, D) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial u}, \frac{\partial h}{\partial x}, \frac{\partial h}{\partial u}\right)\Bigg|_{(x=x^*, u=u^*)}$$

are time-varying matrices since x^* and u^* are time dependent. We take the liberty here to use the operator notation

$$\delta y_k = \Delta G^* \delta u_k. \tag{2.4}$$

The following shorthand notation is a natural extension of the block notation of earlier chapters for time-invariant systems to time-varying systems. The asterisk highlights the optimal state and control dependence,

$$\Delta G^* : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]^* . \tag{2.5}$$

Let us denote $\Delta\bar{G}$ as the operator of the system with input $\Delta u = u - u^*$ and output $\Delta y = y - y^*$. Of course, the 'linearization' can only be a good one and the following design approach effective if the actual plant is not too different in behavior from that of the model G.

Let us associate with the linearized model a quadratic performance index penalizing departures Δy and Δu away from the optimal trajectory:

$$\Delta I^* = \frac{1}{T}\sum_{k=0}^{T} e_k' e_k, \tag{2.6}$$

where,

$$\begin{gathered} e = L^* \begin{bmatrix} \Delta y \\ \Delta u \end{bmatrix}, \qquad L^* L^{*\prime} = \begin{bmatrix} Q_c^* & S_c^* \\ S_c^{*\prime} & R_c^* \end{bmatrix}, \\ Q_c^* = Q_c^{*\prime} \geq 0, \qquad Q_c^* - S_c^* (R_c^*)^{-1} S_c^* \geq 0, \qquad R_c^* = R_c^{*\prime} > 0. \end{gathered} \tag{2.7}$$

Here e is interpreted as a disturbance response which we seek to minimize in an rms sense. Of course, ΔI^* and thus L^*, can be generated from a Taylor series expansion for I about $I^* = (1/T)\sum_{k=0}^{T} \ell(x_k^*, u_k^*)$ up to the second order term. In fact the first two terms are zero due to optimality and the second term can be selected as ΔI^* with $L^* L^{*\prime}$ the Hessian matrix of I^*. Other selections for ΔI^* could be simpler and even more appropriate.

As already noted, we assume that u^*, x^*, y^* are known *a priori* from an open-loop off-line design such as found in books on nonlinear optimal control, see for example Teo et al. (1991). However when u^* is applied to an actual plant G, which includes unmodeled disturbances and/or dynamics, there are departures from the optimal trajectories. With departures $\Delta y = y - y^*$ measured on-line, a standard approach is to apply control adjustments $\Delta u = u - u^*$ to the optimal control by means of output feedback control to minimize (2.6). Thus for the augmented plant arrangement, denoted P_A, and depicted in Figure 2.1, let us consider a linear feedback regulator. We base such a design on the linearized situation depicted in Figure 2.2 where the linearized nominal plant, denoted P, is given from

$$P = \begin{bmatrix} * & P_{12} \\ * & P_{22} \end{bmatrix}, \qquad P_{12} = L \begin{bmatrix} \Delta G^* \\ I \end{bmatrix}, \qquad P_{22} = \Delta G^*. \tag{2.8}$$

The star terms P_{11}, P_{21} are not of interest for the subsequent analysis. Of course in Figure 2.2 the outputs e and Δy are not identical to those of Figure 2.1, but they approximate these.

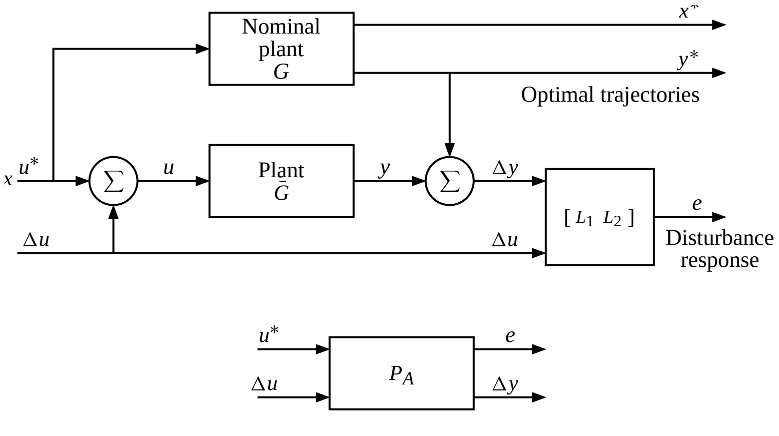


FIGURE 2.1. The augmented plant arrangement

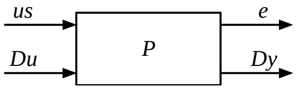


FIGURE 2.2. The linearized augmented plant

Feedback Regulator for Linearized Model

Let the regulator of the linearized model P above, based on the nominal model ΔG^* (equation (2.3)), be given by

$$\begin{aligned} K^* : \delta\hat{x}_{k+1} &= A\delta\hat{x}_k + B\delta u_k - Hr_k, &\qquad \delta\hat{x}_0 &= 0, \\ r_k &= \delta y_k - C\delta\hat{x}_k - D\delta u_k, &\qquad \delta u_k &= F\delta\hat{x}_k = K^*\delta y_k. \end{aligned} \tag{2.9}$$

Here r is the estimator residual, $\delta\hat{x}$ is the estimate of δx and H and F are time-varying matrices formed, perhaps via standard LQG/LTR theory of Chapter 4, see also Anderson and Moore (1989), so that under uniform stability of A, B and uniform detectability of A, C the following systems are exponentially stable:

$$\xi_{k+1} = (A + BF)\xi_k, \qquad \zeta_{k+1} = (A + HC)\zeta_k. \tag{2.10}$$

Actually, the important aspect of the LQG design for our purposes is that under the relevant uniform stabilizability and uniform detectability assumptions, the (time-varying) gains H, F exist, and are given from the solution of two Riccati equations with no finite escape time. Moreover, for the limiting case when the time horizon T becomes infinite, the controller K^* stabilizes ΔG^*.

It is well known that the LQG controller (2.9) for the linearized plants (2.3), although optimal for the nominal linear time-varying plant for the assumed noise environment, may be far from optimal in other than the nominal noise environments, or in the presence of structured or unstructured perturbations on (2.3). Stability may be lost even for small variations from the nominal plant.

Methods to enhance LQG regulator robustness exist, such as modifying Q_c, S_c, R_c (usually $S_c \equiv 0$) selections, or assumed noise environments, as when loop recovery is used. Such techniques could well serve to strengthen the robustness properties of the optimal/adaptive schemes studied subsequently.

In order to proceed, we here merely assume the existence of a controller (2.9) stabilizing ΔG^*, although our objective is to achieve a controller which also stabilizes $\Delta\bar{G}$, and achieves a low value of the index ΔI^* when applied to $\Delta\bar{G}$.

Coprime Factorizations

Now it turns out that most of the coprime factorization results of Chapter 2 developed in a time-invariant linear systems context have a natural generalization to time-varying systems. The essential requirement for these to hold is linearity, not time invariance. Thus many of the equations of Chapter 2 still hold with appropriate interpretation of the notation. Notions of system stability, system inverse, series and parallel connection of systems all carry over to this time-varying system context in a natural way. We proceed on this basis, and leave the reader to check such details by working through a problem at the end of the chapter. Let it suffice here to state that the developments proceed most naturally in the state space framework developed in Sections 2.4 and 2.5.

Here, it is convenient to introduce normalized x^*-dependent and u^*-dependent coprime factorizations for ΔG^* and K^*, such that

$$\Delta G^* = NM^{-1} = \tilde{M}^{-1}\tilde{N}, \tag{2.11}$$

$$K^* = UV^{-1} = \tilde{V}^{-1}\tilde{U}, \tag{2.12}$$

satisfy the double Bezout identity,

$$\begin{aligned}\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix}\begin{bmatrix} M & U \\ N & V \end{bmatrix} &= \begin{bmatrix} M & U \\ N & V \end{bmatrix}\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}.\end{aligned} \tag{2.13}$$

Here the factors N, M, N, V, $\tilde{M}$, $\tilde{N}$, $\tilde{U}$, $\tilde{V}$ are stable and causal operators. Since they are x^*-, u^*-dependent, and thus time-varying system linear operators, they are natural generalizations of the linear time-invariant operators (transfer functions) of earlier chapters. Here the product notation is that of the concatenation of systems (i.e. of linear system operators).

Now using the notation of (2.5), suitable factorizations are readily verified under (2.10) as, see also Moore and Tay (1989b),

$$\begin{aligned}\begin{bmatrix} M & U \\ N & V \end{bmatrix} &= \left[\begin{array}{c|cc} A+BF & B & -H \\ \hline F & I & 0 \\ C+DF & -D & I \end{array}\right]^*, \\ \begin{bmatrix} \tilde{V} & -\tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix} &= \left[\begin{array}{c|cc} A+HC & -(B+HD) & H \\ \hline F & I & 0 \\ C & -D & I \end{array}\right]^*.\end{aligned} \tag{2.14}$$

The Class of all Stabilizing Controllers

The theory of the class of stabilizing linear controllers for linear plants, spelled out in Chapter 2 for the time-variant case, is readily generalized to cope with time-varying systems. Again, the strength of the results depends on linearity, not the time-invariance. The details are left for the reader to verify in a problem at the end of the chapter, (see also Imae et al. (1992), Moore and Tay (1989b), Tay and Moore (1990)). Thus, the class of all linear, causal stabilizing controllers for ΔG^* (the linearized plant model) under (2.10) can be generated, not surprisingly, as depicted in Figure 2.3 using a J_k subsystem defined below, and a so-called Q parameterization. Here the blocks ΔG, H, A, B, C, F and Q are time-varying linear system operators. Referring also to Figure 2.4, the subsystem J_K is readily extracted.

$$\begin{aligned} J_K : \delta\hat{x}_{k+1} &= (A+BF)\delta\hat{x}_k + Bs_k - Hr_k, \\ \delta u_k = F\delta\hat{x}_k + s_k, &\qquad r_k = \delta y_k - C\delta\hat{x}_k - D\delta u_k, \end{aligned} \tag{2.15}$$

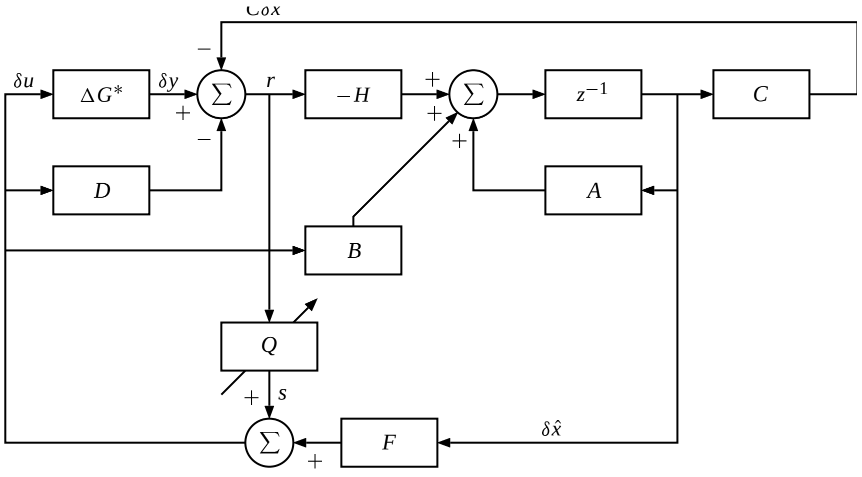


FIGURE 2.3. Class of all stabilizing controllers—the linear time-varying case

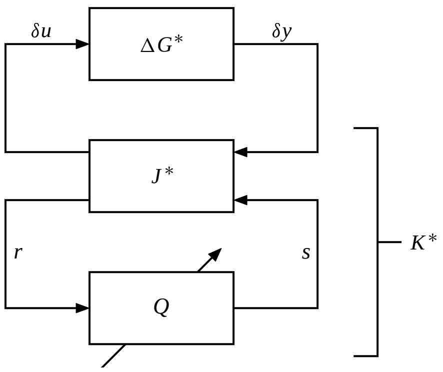


FIGURE 2.4. Class of all stabilizing time-varying linear controllers

or equivalently,

$$J_K = \begin{bmatrix} K & \tilde{V}^{-1} \\ V^{-1} & -V^{-1}N \end{bmatrix}. \tag{2.16}$$

In the Figure 2.4, Q is arbitrary within the class of all linear, time varying, causal bounded-input, bounded-output (BIBO) stable operators. Thus:

$$\begin{aligned} K^*(Q) &= U(Q)V^{-1}(Q) = \tilde{V}^{-1}(Q)\tilde{U}(Q), \\ U(Q) &= U + MQ, \qquad V(Q) = V + NQ, \\ \tilde{U}(Q) &= \tilde{U} + Q\tilde{M}, \qquad \tilde{V}(Q) = \tilde{V} + Q\tilde{N}, \end{aligned} \tag{2.17}$$

or equivalently, after some manipulations involving (2.12) and (2.13),

$$K^*(Q) = K + \tilde{V}^{-1}Q(I + V^{-1}NQ)^{-1}V^{-1}. \tag{2.18}$$

Simple manipulations also give an alternative expression for r, as

$$r = \tilde{M}\delta y - \tilde{N}\delta u. \tag{2.19}$$

It is known that the closed-loop transfer functions (operators) of Figure 2.4 are affine in Q, which facilitates either off-line or on-line optimization of such Q dependent transfer operators. We proceed with a class of on-line optimizations.

Adaptive-Q Control

Our proposal is to implement a controller $K^*(Q)$ for some adaptive Q-scheme applied to $\Delta\bar{G}$. The intention is for Q to be chosen to ensure that $K^*(Q)$ stabilizes the feedback loop and thereby the original plant G, and moreover, achieves good performance in terms of the index ΔI^* of (2.6). Thus consider the arrangement of Figure 2.5 where the block P is characterized by $\Delta\bar{G}$ and L

A refinement on this proposal is to consider a two-degree-of-freedom controller scheme. This is depicted in Figure 2.6. As discussed in Chapter 2, it can be derived from a one-degree-of-freedom controller arrangement for an augmented plant □ $= \left[\begin{smallmatrix} 0 \\ G' \end{smallmatrix}\right]$, reorganized as a two-degree-of-freedom arrangement for G. The objective is to select □ $= [\, Q_f \;\; Q \,]$ causal, bounded-input, bounded-output operators on line so that the response e is minimized in an ℓ_2 sense, see also the work of Tay and Moore (1990).

In order to present a least squares algorithm for selection of Q, generalizing the schemes of Chapter 6 to the time-varying case as in the schemes of Moore and Tay (1989b), some preprocessing of the signals e, δu, δy is required.

Prefiltering

Using operator notation, we define filtered variables

$$\xi = \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} P_{12}M & u^* \\ P_{12}M & r \end{bmatrix}, \qquad \zeta = e - P_{12}Ms. \tag{2.20}$$

Least Squares □ Selection

To fix the ideas, and to simplify notation, we assume r and s to be scalar signals in the subsequent developments. Let us define a (possibly time-varying)

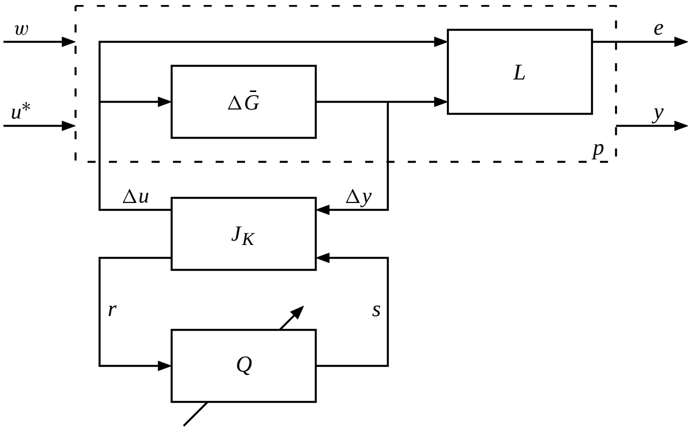


FIGURE 2.5. Adaptive Q for disturbance response minimization

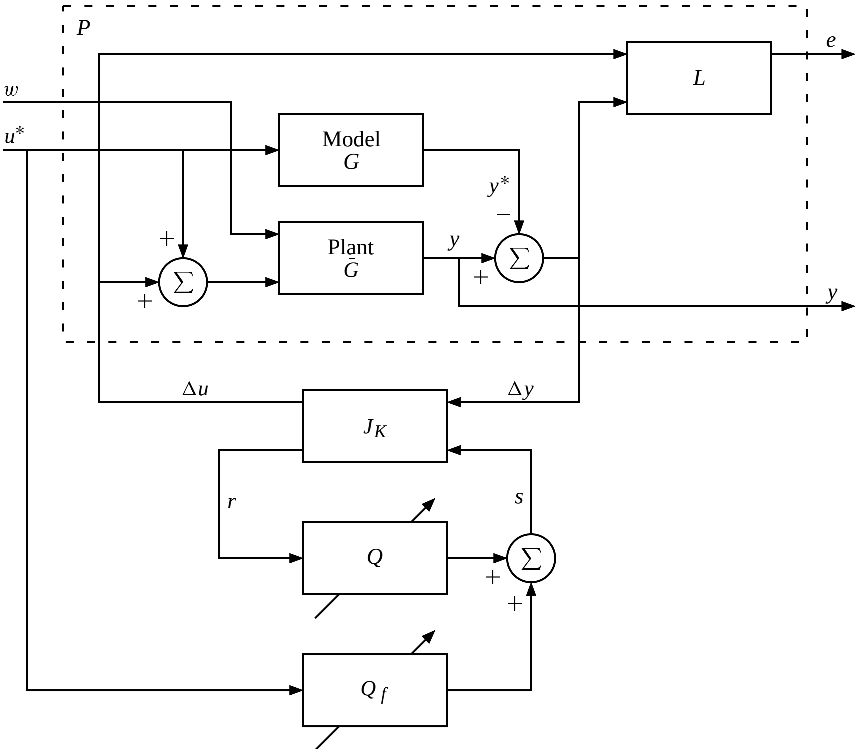


FIGURE 2.6. Two degree-of-freedom adaptive-Q scheme

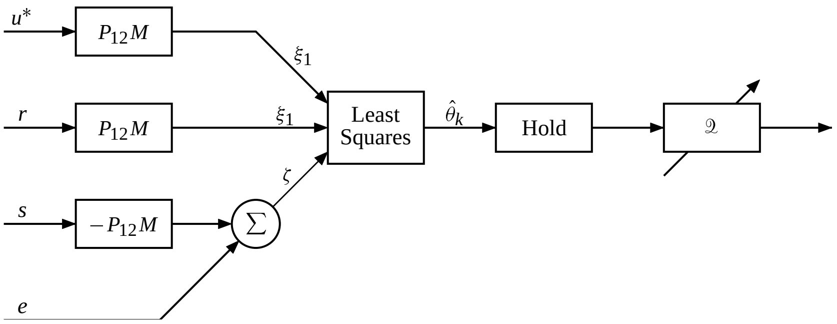


FIGURE 2.7. The least squares adaptive-Q arrangement

single-input, single-output, discrete-time version of $\square$ in terms of a unit delay operator q^{-1},

$$
\begin{aligned}
Q_f(q^{-1}) &= \frac{\gamma + \gamma_1 q^{-1} + \cdots + \gamma_p q^{-p}}{1 + \alpha_1 q^{-1} + \cdots + \alpha_n q^{-n}}, \\
Q(q^{-1}) &= \frac{\beta + \beta_1 q^{-1} + \cdots + \beta_m q^{-m}}{1 + \alpha_1 q^{-1} + \cdots + \alpha_n q^{-n}}, \\
\square(q^{-1}) &= \begin{bmatrix} Q_f(q^{-1}) & Q(q^{-1}) \end{bmatrix}, \\
\theta' &= \begin{bmatrix} \alpha_1 \dots \alpha_n & \beta_1 \dots \beta_m & \gamma \dots \gamma_p \end{bmatrix},
\end{aligned}
\tag{2.21}
$$

with (possibly time-varying) parameters $\alpha_i, \beta_i, \gamma_i$. The following state (regression) vector in discrete time is

$$
\phi_k' = \begin{bmatrix} -s_{k-1} & \dots & -s_{k-n} & r_k & \dots & r_{k-m} & \omega_k & \dots & \omega_{k-p} \end{bmatrix}. \tag{2.22}
$$

The dimensions n, m, p are set from an implementation convenience/performance trade-off. In the adaptive-Q case, the parameters are time-varying resulting from least squares calculations given below. We assume a unit delay in calculations. Thus θ is replaced by $\hat{\theta}_{k-1}$ and the filter with operator $\square_k = [\, Q_{fk} \; Q_k \,]$ is implemented with parameters (time-varying in general) as

$$
s_k = \hat{\theta}_{k-1}' \phi_k, \qquad \hat{\theta}_k' = \begin{bmatrix} \hat{\alpha}_{1k} \dots \hat{\alpha}_{nk} & \hat{\beta}_{0k} \dots \hat{\beta}_{mk} & \hat{\gamma}_{0k} \dots \hat{\gamma}_{pk} \end{bmatrix}. \tag{2.23}
$$

We seek selections of $\hat{\theta}_k$ so that the adaptive controller minimizes the ℓ_2 norm of the response e_k. With suitable initializing we have the adaptive-Q arrangement of

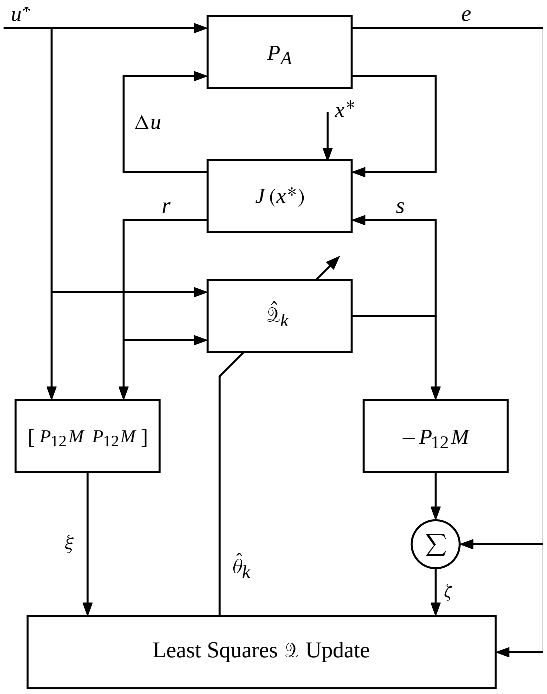


FIGURE 2.8. Two degree-of-freedom adaptive-Q scheme

Figure 2.6 with equations

$$\begin{aligned}
\hat{\theta}_k &= \hat{\theta}_{k-1} + \hat{P}_k \hat{\phi}_k \hat{e}_{k/k-1}, \\
\hat{e}_{k/k-1} &= \zeta_k - \hat{\phi}_k' \hat{\theta}_{k-1}, \\
e_{k/k} &= \zeta_k - \hat{\phi}_k' \hat{\theta}_k, \\
\hat{P}_k &= \left(\sum_{i=1}^{k} \hat{\phi}_i \hat{\phi}_i' \right)^{-1} = \hat{P}_{k-1} - \hat{P}_{k-1}\hat{\phi}_k (I + \hat{\phi}_k' \hat{P}_{k-1} \hat{\phi}_k)^{-1} \hat{\phi}_k \hat{P}_{k-1}, \\
\hat{\phi}_k' &= \big[\; (\hat{e}_{k-1/k-1} - \zeta_{k-1}) \quad \dots \quad (\hat{e}_{k-n/k-n} - \zeta_{k-n}) \\
&\qquad -\xi_{2,k} \quad \dots \quad -\xi_{2,k-m} \quad -\xi_{1,k} \quad \dots \quad -\xi_{1,k-m} \; \big].
\end{aligned} \tag{2.24}$$

The complete adaptive-Q scheme is a combination of Figures 2.6 and 2.7 with key equations (2.14) and (2.24), see also Figure 2.8. A number of remarks concerning the algorithm are now in order.

The algorithms (2.24) should be modified to ensure that $\hat{\theta}_k$ is projected into a restricted domain, such as $\|\Box_k\| < \epsilon$, for some norm and some fixed ϵ. Such projections can be guided by the theory discussed in the next section.

To achieve convergence of $\hat{\theta}_k$, then $\hat{P}_k$ must approach zero, or equivalently, $\hat{\phi}_k$ must be persistently exciting in some sense. However, parameter convergence is not strictly necessary to achieve performance enhancement. With more general

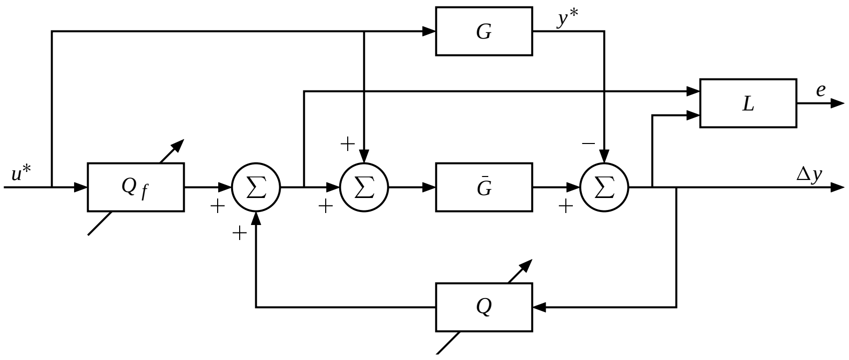


FIGURE 2.9. Model reference adaptive control special case

algorithms which involve resetting or forgetting, then care must be taken to avoid ill-conditioning of $\hat{P}_k$, as can occur when there is instability.

It turns out that appropriate scaling can be crucial to achieve the best possible performance enhancement. Scaling gains can be included to scale r and/or e with no effect on the supporting theory, other than when defining projection domains as above. Likewise, the "scaling" can be generalized to stable dynamic filters for r and/or e with no effect on the supporting theory. In this way frequency shaped designs can be effected.

The scheme described above can be specialized to the cases when Q_f, Q are finite impulse response filters by setting $n = 0$. The □, so defined, are stable for all bounded $\hat{\theta}_k$. Also, either Q_f or Q can be set to zero to simplify the processing, although possibly at the expense of performance.

In the case that Q_f is a moving average and Q is zero, then our scheme becomes very simple, being a moving average filter Q_f in series with the closed-loop system $(\Delta\bar{G}, K)$. In this case then, if Q_f is stable, guaranteed when the gains $\hat{\theta}_k$ are bounded, and $(\Delta\bar{G}, K)$ is stable, then there is obvious stability of the adaptive scheme.

When the linearized plant model ΔG^* is stable, and one selects trivial values $F, H = 0$ so that $K = 0$, then the arrangement of Figure 2.6 simplifies to a familiar model-reference adaptive control arrangement depicted in Figure 2.9.

In the case that Q_f is set to zero there is no adaptive feedforward control action.

The operators ΔG^*, J_K are in fact functions of the optimal trajectories x^*. It makes sense then to have the operator Q also as a function of x^*. Then the adaptive-Q approach generalizes naturally to a learning-Q approach as studied in a later section.

Main Points of Section

In the case of "smooth" nonlinear systems, linearizations yield trajectory dependent time-varying models. Straightforward generalizations of the adaptive-Q methods of Chapter 6 to the time-varying case allow application of the ideas to enhance the performance and robustness of "optimal" nonlinear controllers.

8.3 Stability Properties

In this section we focus on stability results as a basis to achieve convergence results for our system. We first analyze a parameterization of the *nonlinear* plant $\Delta\bar{G}$ with input Δu and output Δy in terms of the coprime factorizations of the linearized version ΔG^*, and stabilizing *linear* controller K^*, and establish that this parameterization covers the class of well-posed closed-loop systems under study. Next, stability of the scheme is studied in terms of such parameterizations and then expected convergence properties are noted based on this characterization and known convergence theories in the linear time invariant case.

Nonlinear System Fractional Maps[*]

Let us consider the right and left coprime factorizations for the nominal linearized plant and controller, paralleling the approach used in Chapter 2 for linear systems, but here with time varying and at times nonlinear systems. The operators are expressed as functions of the desired optimal trajectory x^*, and optimal control u^*, but since x^*, u^* are time dependent, then for any specific $x^*(\cdot)$, $u^*(\cdot)$ the operators are merely linear time-varying operators, and can be treated as such. We denote $\Delta\bar{G}$ as the (nonlinear) system with input Δu and output Δy, and ΔG^* is a linearization of the nominal plant G. Also, a unity gain feedback loop with open loop operator W_{ol} is said to be well-posed when $(I + W_{ol})^{-1}$ exists. Recall that for a nonlinear operator S, then, in general $S(A + B) \neq SA + SB$, or equivalently superposition does not hold, and care must be taken in the composition of nonlinear operators.

Theorem 3.1 (Right fractional map forms). *Consider that the time-varying linear feedback system pair $(\Delta G^*, K^*)$ is well posed and stabilizing with left and right coprime factorizations for ΔG^*, K^*, as in* (2.11) *and* (2.12)*, and the double Bezout* (2.13) *holds. Then any* nonlinear *plant with $\Delta\bar{G}$ such that $(\Delta\bar{G}, K^*)$ is a well-posed closed-loop system can be expressed in terms of a* (nonlinear) *operator S in right fractional map forms:*

$$\Delta\bar{G}(S) = N(S)M^{-1}(S) \tag{3.1}$$

$$= \Delta\bar{G} + \tilde{M}^{-1}S(I + M^{-1}US)^{-1}M^{-1}, \tag{3.2}$$

where

$$N(S) = (N + VS), \qquad M(S) = (M + US). \tag{3.3}$$

[*]The remainder of this section can be omitted on first reading, or at least the proof details which perhaps require technical skill, albeit paralleling developments for the linear case.

Also, closed-loop system (nonlinear) operators are given from

$$\begin{bmatrix} I & -K^* \\ -\Delta\bar{G} & I \end{bmatrix}^{-1} = \begin{bmatrix} I & -K^* \\ -\Delta G^* & I \end{bmatrix}^{-1} + \begin{bmatrix} U & M \\ V & N \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix}. \tag{3.4}$$

Moreover, the maps (3.1), (3.2) *have the block diagram representations of Figure 3.1(a) and (b) where*

$$J_G = \begin{bmatrix} -M^{-1}U & M^{-1} \\ \tilde{M}^{-1} & \Delta G^* \end{bmatrix}. \tag{3.5}$$

The solutions of (3.1), (3.2) *are unique , given from the right fractional maps in terms of* $\Delta\bar{G}$*, or* $(\Delta G^* - \Delta\bar{G})$ *as the (nonlinear) operator*

$$\begin{aligned} S &= (-\tilde{N} + \tilde{M}\Delta\bar{G})(\tilde{V} - \tilde{U}\Delta\bar{G})^{-1} && (3.6) \\ &= \tilde{M}(\Delta\bar{G} - \Delta G^*)M[I - \tilde{U}(\Delta\bar{G} - \Delta G^*)M]^{-1}, && (3.7) \end{aligned}$$

or in terms of the closed-loop system operators as

$$S = \begin{bmatrix} -\tilde{N} & \tilde{M} \end{bmatrix} \left[\begin{bmatrix} I & -K^* \\ -\Delta\bar{G} & I \end{bmatrix}^{-1} - \begin{bmatrix} I & -K^* \\ -\Delta G^* & I \end{bmatrix}^{-1} \right] \begin{bmatrix} M \\ N \end{bmatrix}. \tag{3.8}$$

Moreover, $(N(S), M(S))$ *are coprime and obey a Bezout identity*

$$\tilde{V}M(S) - \tilde{U}N(S) = I. \tag{3.9}$$

Proof. Care must be taken in our proof to permit only operations valid for nonlinear operators when these are involved. Now simple manipulations allow (3.6) to be reorganized under the well-posedness assumption as

$$\begin{bmatrix} I \\ S \end{bmatrix} = \begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} I \\ \Delta\bar{G} \end{bmatrix} (I - K^*\Delta\bar{G})^{-1}\tilde{V}^{-1},$$

and via the Bezout identity, as

$$\begin{bmatrix} M(S) \\ N(S) \end{bmatrix} = \begin{bmatrix} M + US \\ N + VS \end{bmatrix} = \begin{bmatrix} M & U \\ N & V \end{bmatrix} \begin{bmatrix} I \\ S \end{bmatrix} = \begin{bmatrix} I \\ \Delta\bar{G} \end{bmatrix} (I - K^*\Delta\bar{G})^{-1}\tilde{V}^{-1}. \tag{3.10}$$

Thus under (3.6) then $M^{-1}(S)$ exists and, (3.1) holds as follows

$$N(S)M^{-1}(S) = \Delta\bar{G}(I - K^*\Delta\bar{G})^{-1}\tilde{V}^{-1}\left[(I - K^*\Delta\bar{G})^{-1}\tilde{V}^{-1}\right]^{-1} = \Delta\bar{G}.$$

To prove the equivalence of (3.1) and (3.2), simple manipulations give

$$\begin{aligned}\Delta\bar{G} &= \Delta G^* + (N+VS)(I+M^{-1}US)^{-1}M^{-1} - NM^{-1}\\ &= \Delta G^* + (V - NM^{-1}U)S(I+M^{-1}US)^{-1}M^{-1}\\ &= \Delta G^* + (V - \tilde{M}^{-1}\tilde{N}U)S(I+M^{-1}US)^{-1}M^{-1}\\ &= \Delta G^* + \tilde{M}^{-1}(\tilde{M}V - \tilde{N}U)S(I+M^{-1}US)^{-1}M^{-1}\\ &= \Delta G^* + \tilde{M}^{-1}S(I+M^{-1}US)^{-1}M^{-1},\end{aligned}$$

so that under (2.13), we have that (3.2) holds. Likewise (3.6) is equivalent to (3.7) as follows

$$\begin{aligned}S &= \tilde{M}(\Delta\bar{G} - \Delta G^*)(\tilde{V} - \tilde{U}\Delta\bar{G})^{-1}\\ &= \tilde{M}(\Delta\bar{G} - \Delta G^*)M(\tilde{V}M - \tilde{U}\Delta\bar{G}M)^{-1}\\ &= \tilde{M}(\Delta\bar{G} - \Delta G^*)M(I + \tilde{U}NM^{-1}M - \tilde{U}\Delta\bar{G}M)^{-1}\\ &= \tilde{M}(\Delta\bar{G} - \Delta G^*)M[I - \tilde{U}(\Delta\bar{G} - \Delta G^*)M]^{-1}.\end{aligned}$$

To see that the operator of (3.1) is equivalent to that depicted in Figure 3.1(a), observe from Figure 3.1(a) that $\ell = M^{-1}(e_1 - US\ell)$, or equivalently, $\ell = (M+US)^{-1}e$. Also, $(e_2 - w_2) = (N+VS)\ell = (N+VS)(M+US)^{-1}e_1$, which is equivalent to (3.1).

Now suppose there is some other $(S + \Delta S)$ which also satisfies (3.1). Then

$$\begin{aligned}\begin{bmatrix} I \\ \Delta\bar{G}\end{bmatrix} &= \begin{bmatrix} M & U \\ N & V\end{bmatrix}\begin{bmatrix} I \\ S\end{bmatrix}(M+US)^{-1}\\ &= \begin{bmatrix} M & U \\ N & V\end{bmatrix}\begin{bmatrix} I \\ S+\Delta S\end{bmatrix}(M+US+U\Delta S)^{-1}.\end{aligned}$$

Then, using (2.13),

$$\begin{aligned}\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M}\end{bmatrix}\begin{bmatrix} I \\ \Delta\bar{G}\end{bmatrix} &= \begin{bmatrix} I \\ S\end{bmatrix}(M+US)^{-1}\\ &= \begin{bmatrix} I \\ S+\Delta S\end{bmatrix}(M+US+U\Delta S)^{-1}.\end{aligned} \tag{3.11}$$

Premultiplication by $[\,I\ 0\,]$ gives $M + US = M + US + U\Delta S$, and premultiplication by $[\,0\ I\,]$ gives in turn that $\Delta S = 0$.

To verify (3.8), first observe that

$$\begin{bmatrix} I & -K^* \\ -\Delta\bar{G} & I\end{bmatrix} = \begin{bmatrix} M & -U \\ -N & V\end{bmatrix}\begin{bmatrix} I & 0 \\ -S & I\end{bmatrix}\begin{bmatrix} M+US & 0 \\ 0 & V\end{bmatrix}^{-1}. \tag{3.12}$$

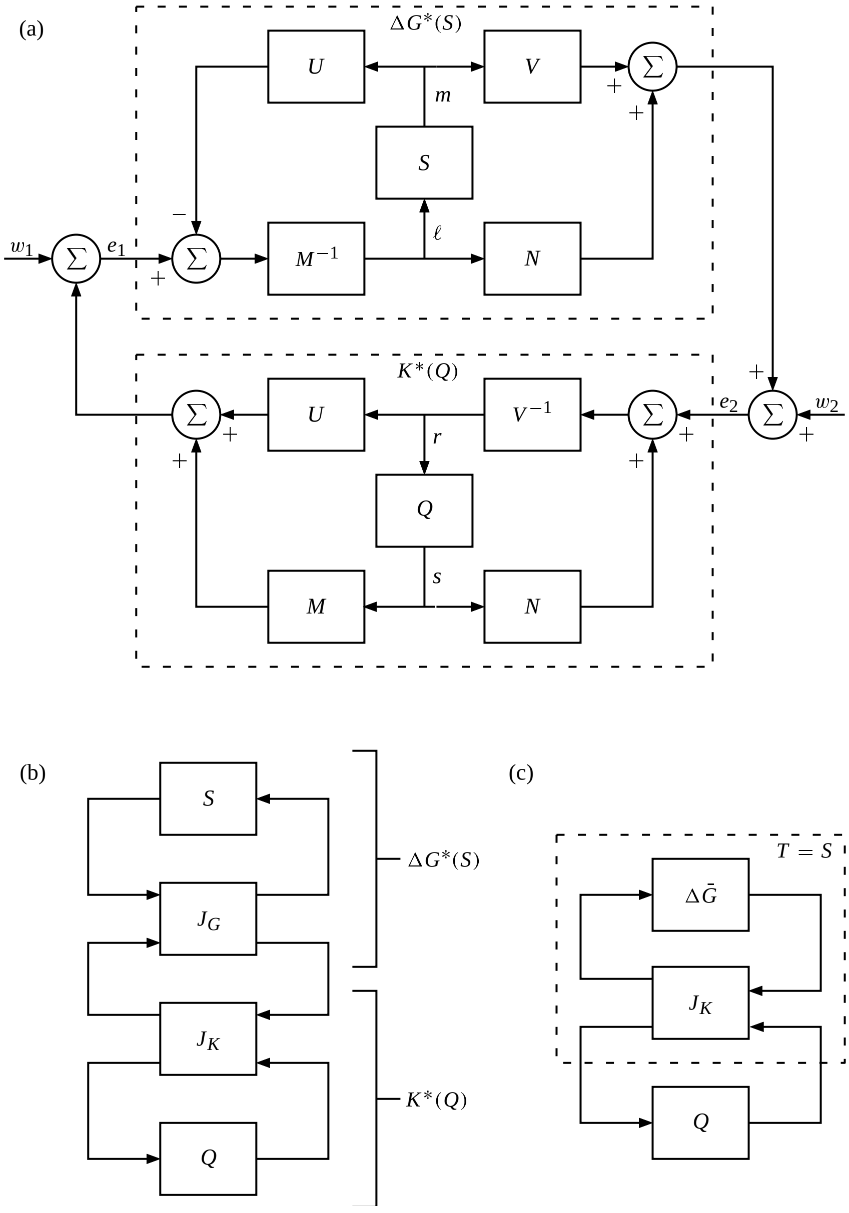


FIGURE 3.1. The feedback system $(\Delta G^*(S), K^*(Q))$

Thus

$$\begin{bmatrix} I & -K^* \\ -\Delta\bar{G} & I \end{bmatrix}^{-1} - \begin{bmatrix} I & -K^* \\ -\Delta G^* & I \end{bmatrix}^{-1}$$
$$= \left[\begin{bmatrix} M+US & 0 \\ 0 & V \end{bmatrix}\begin{bmatrix} I & 0 \\ -S & I \end{bmatrix}^{-1} - \begin{bmatrix} M & 0 \\ 0 & V \end{bmatrix}\right]\begin{bmatrix} M & -U \\ -N & V \end{bmatrix}^{-1}$$
$$= \begin{bmatrix} M & U \\ N & V \end{bmatrix}\begin{bmatrix} 0 & 0 \\ S & 0 \end{bmatrix}\begin{bmatrix} M & -U \\ -N & V \end{bmatrix}^{-1},$$

and applying the double Bezout (2.13) gives

$$\begin{bmatrix} \tilde{V} & -\tilde{U} \\ -\tilde{N} & \tilde{M} \end{bmatrix}\left[\begin{bmatrix} I & -K^* \\ -\Delta\bar{G} & I \end{bmatrix}^{-1} \begin{bmatrix} I & -K^* \\ -\Delta G^* & I \end{bmatrix}^{-1}\right]\begin{bmatrix} M & -U \\ -N & V \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ S & 0 \end{bmatrix},$$

or equivalently (3.4) holds, and (3.8). (This result is generalized in Theorem 3.2.)

Simple manipulations from Figure 3.1(b) give the operator representation of the G block to be $J_{21}S(1 - J_{11}S)^{-1}J_{12} + J_{22}$, and substitution of (3.5) gives $\Delta\bar{G}$ by (3.2).

To establish coprimeness of $N(S)$, $M(S)$ observe that under the double bezout (2.13)

$$\tilde{V}M(S) - \tilde{U}N(S) = \tilde{V}M - \tilde{U}N + (\tilde{V}U - \tilde{U}V)S = I.$$

Since I is unimodular, then from Paice and Moore (1990b, Lemma 2.1), $N(S)M(S)^{-1}$ is a right coprime factorization. □

A number of remarks are in order. When $\Delta\bar{G}$ is linear and time invariant, the above results specialize to the results in Chapter 2, although the details of the theorem proof appears quite different so as to avoid using superposition when nonlinear operators $\Delta\bar{G}$, S are involved.

The fact that $\tilde{M}, \tilde{N}, M, N, \tilde{U}, \tilde{V}, U, V$ are linear has allowed derivations to take place without differential boundedness or other such assumptions as in a full nonlinear theory as developed in Paice and Moore (1990a), Paice and Moore (1990b) using left coprime factorizations.

Dual results apply for fractional mappings of $K^*(Q)$, as in (3.13) and (3.14) along with duals of the other results. Thus $K^*(Q)$ can be expressed as a linear controller K^* augmented with a nonlinear Q. Also, by duality, Figure 3.1(a) depicts a block diagram arrangement for

$$K^*(Q) = U(Q)V^{-1}(Q); \qquad U(Q) = (U + MQ), \qquad V(Q) = (V + NQ), \tag{3.13}$$

where

$$Q = (-\tilde{U} + \tilde{V} K^*(Q))(\tilde{M} - \tilde{N} K^*(Q))^{-1}. \tag{3.14}$$

Stabilization Results

We define a system (G, K^*) with G possibly nonlinear to be *internally stable* if for all bounded inputs, the outputs are bounded.

Theorem 3.2. *Consider the well-posed feedback system* $(\Delta\bar{G}, K^*)$ *under the conditions of Theorem 3.1, with* $\Delta\bar{G}$ *and* K^* *parameterized by* Q, S *as in* (3.1), (3.13) *and as depicted in Figure 3.1(a) and (b). Then the pair* $(\Delta G^*(S), K^*(Q))$ *is well posed and internally stable if and only if the feedback system* (Q, S) *depicted in Figure 3.2 is well posed and internally stable. Moreover, referring to Figure 3.1(c), the* J_K, $\Delta\bar{G}$ *block with input/output operator* T *satisfies*

$$T = S. \tag{3.15}$$

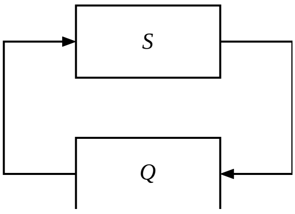


FIGURE 3.2. The feedback system (Q, S)

Proof. Observe that from (3.1),(3.13)

$$\begin{bmatrix} I & -K^*(Q) \\ -\Delta G^*(S) & I \end{bmatrix} = \begin{bmatrix} M & -U \\ -N & V \end{bmatrix} \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix} \begin{bmatrix} M + US & 0 \\ 0 & V + NQ \end{bmatrix}^{-1}. \tag{3.16}$$

Clearly, under the double Bezout identity (2.13), or equivalently under $(\Delta G^*, K^*)$ well posed and internally stable,

$$\begin{bmatrix} I & -K^*(Q) \\ -\Delta G^*(S) & I \end{bmatrix}^{-1} \text{ exists} \quad \Longleftrightarrow \quad \begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} \text{ exists.}$$

Equivalently, $(\Delta G^*(S), K^*(Q))$ is well posed if and only if (Q, S) is well posed. Thus under well-posedness assumptions, taking inverses in, and exploiting

(3.16), simple manipulations yield

$$\begin{aligned}
&\begin{bmatrix} I & -K^*(Q) \\ -\Delta G^*(S) & I \end{bmatrix}^{-1} \\
&\quad = \left[\begin{bmatrix} M & 0 \\ 0 & V \end{bmatrix} + \begin{bmatrix} U & 0 \\ 0 & N \end{bmatrix}\begin{bmatrix} S & 0 \\ 0 & Q \end{bmatrix}\right]\begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1}\begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix} \\
&\quad = \begin{bmatrix} I & -K^* \\ -\Delta G^* & I \end{bmatrix}^{-1} \\
&\qquad + \begin{bmatrix} U & M \\ V & N \end{bmatrix}\begin{bmatrix} S & 0 \\ 0 & Q \end{bmatrix}\begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1}\begin{bmatrix} \tilde{V} & \tilde{U} \\ \tilde{N} & \tilde{M} \end{bmatrix}.
\end{aligned} \tag{3.17}$$

Now internal stability of $(\Delta G^*, K^*)$, (Q, S), and stability of $N, \tilde{N}$, etcetera leads to internal stability of the right hand side and thus of $(\Delta G^*(S), K^*(Q))$ as claimed. Moreover from (3.17), (2.13)

$$\begin{aligned}
&\begin{bmatrix} S & 0 \\ 0 & Q \end{bmatrix}\begin{bmatrix} I & -Q \\ -S & I \end{bmatrix}^{-1} \\
&\quad = \begin{bmatrix} -\tilde{N} & \tilde{M} \\ \tilde{V} & -\tilde{U} \end{bmatrix}\left[\begin{bmatrix} I & -K^*(Q) \\ -\Delta G^*(S) & I \end{bmatrix}^{-1} - \begin{bmatrix} I & -K^* \\ -\Delta \bar{G} & I \end{bmatrix}^{-1}\right] \\
&\qquad \times \begin{bmatrix} M & -U \\ -N & V \end{bmatrix}.
\end{aligned}$$

Thus well-posedness and internal stability of $(\Delta G^*(S), K^*(Q))$ and $(\Delta G^*, K^*)$ gives well-posedness and internal stability of (Q, S) to complete the first part of the proof.

Now with J_K defined as in (2.16), the operator T in Figure 3.1(c) can be represented as

$$\begin{aligned}
T &= V^{-1}\Delta\bar{G}(I - \tilde{V}^{-1}\tilde{U}\Delta\bar{G})^{-1}\tilde{V}^{-1} - \tilde{N}\tilde{V}^{-1} \\
&= V^{-1}\Delta\bar{G}(\tilde{V} - \tilde{U}\Delta\bar{G})^{-1} - \tilde{N}\tilde{V}^{-1} \\
&= [V^{-1}\Delta\bar{G} - \tilde{N} + \tilde{N}\tilde{V}^{-1}\tilde{U}\Delta\bar{G}](\tilde{V} - \tilde{U}\Delta\bar{G})^{-1} \\
&= \tilde{M}[\tilde{M}^{-1}(V^{-1} + \tilde{N}\tilde{V}^{-1}\tilde{U})\Delta\bar{G} - \Delta\bar{G}](\tilde{V} - \tilde{U}\Delta\bar{G})^{-1} \\
&= \tilde{M}[\Delta\bar{G} - \Delta G](\tilde{V} - \tilde{U}\Delta\bar{G})^{-1} \\
&= S.
\end{aligned} \tag{3.18}$$

□

A number of remarks are in order. This proof does not use superposition associated with operators Q, S, but does in regard to M, N, etc. The results following

Theorem 3.1 also apply for Theorem 3.2. Thus the proof approach differs (of necessity) from the proof approach for the linear Q, S case based on work with the left factorizations , since when working with left factorizations, superposition is used associated with the operators Q, S.

If $\|S\| < \epsilon$ then by the small gain theorem for closed feedback loops, if $\|Q\| < 1/\epsilon$ then Q stabilizes the loop. From this, and Theorem 3.2 with $(\Delta\bar{G} - \Delta G^*)$ suitably small in norm, there exists some Q which will guarantee stability.

In the case where $\Delta\bar{G} = \Delta G^*$, we trivially have $S = 0$, and any Q selection based on identification of S will be trivially $Q = 0$. This contrasts the awkwardness of one alternative design approach which would seek to identify the closed-loop system as a basis for a controller augmentation design.

Observations on examples in the linear $\Delta\bar{G}$ case have shown that if K^* is robust for G, then S can be approximated by a low order system, thus making any Q selection more straightforward than might be otherwise expected.

Averaging analysis could lead to robust convergence results as in the case of linear plants, but such is clearly beyond the scope of this work.

Main Points of Section

The closed-loop stability theory that is the foundation of the adaptive-Q schemes in the time-invariant linear model case is generalized in a natural way to the time-varying linear system context, and at least partially to the nonlinear setting.

Example. Here we demonstrate the efficacy of the adaptive-Q approach through simulation studies. Consider an optimal control problem based on the Van der Pol equation. This equation is considered interesting because it is a simple yet fundamentally nonlinear equation which has a rich oscillatory behavior, not in general sinusoidal.

$$\dot{x}_1 = (1 - x_2^2)x_1 - x_2 + u; \qquad \dot{x}_2 = x_1, \qquad y = x_1, \tag{3.19}$$

with $x_1(0) = 0,\ x_2(0) = 1$ and the performance index defined by

$$I = \frac{1}{2}\int_0^5 (x_1^2 + x_2^2 + u^2)dt. \tag{3.20}$$

A second-order algorithm of Imae and Hakomori (1987), using 400 integration steps, was adopted for the numerical solution of the open-loop optimal control signal u^*. An arbitrary initial nominal control $u \equiv 0$, $t \in [0, 5]$, was chosen. The value of the performance index was reduced to the optimal one in 4 iterations in updating $u(\cdot)$ over the range $[0, 5]$.

Four situations have been studied in simulations. For each case we add a stochastic or a deterministic disturbance which disturbs the optimal input signal. Also, in some of the simulations we apply a plant with unmodeled dynamics. The objective is to regulate perturbations from the optimal by means of the index $\Delta I = \int_0^5 (\delta x_1^2 + \delta x_2^2 + \delta u^2)dt$ which is expressed in terms of perturbations δx, δu. For each of the disturbances added, and for the unmodeled dynamics case, we compare five controller strategies, and demonstrate the robustness and performance properties of the adaptive-Q methodology.

1. **Open-loop design.**
   Here we adopt the optimal control signal u^* as an input signal of the nonlinear system with added disturbance. Figure 3.3 shows that the open-loop design is quite sensitive to such disturbances in that x_1, x_2 differ significantly from x_1^*, x_2^*.

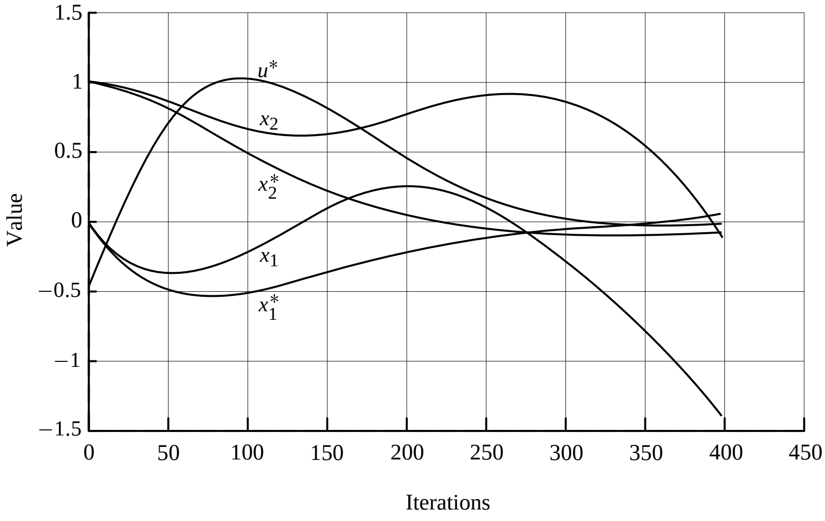


FIGURE 3.3. Open Loop Trajectories

2. **LQG design.**
   In order to construct feedback controllers, we adopt the standard LQG theory based on a discretized model obtained by fast sampling. This model is then linearized about the optimal trajectories and the performance index (3.20). Of course, the input signals $u^* + \delta u$ are no longer ‘optimal’ for the nominal plant. The LQG controller’s design yields better performance than the open-loop case in that the errors $x_1 - x_1^*$, $x_2 - x_2^*$ are mildly smaller than in the previous figure for the open-loop cases, see Table 3.1. It is well known, however, that the LQG controller, although optimal for the nominal plant model under the assumed noise environment, may lose performance and perhaps its stability even for small variations from the nominal plant model.

3. **LQG/LTR design.**
   In order to enhance the robustness properties of LQG controllers, we adopt well known loop transfer recovery (LTR) techniques Doyle and Stein (1979). Thus the system noise covariance Q_f^* in a state estimator design is parameterized by a scaler $q > 0$, and a loop recovery property is achieved

as q becomes large. In our scheme the state estimator 'design system and measurement noise covariances', $Q_f^*(q)$ and R_f^*, are given by

$$Q_f^*(q) = I + q^2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix}; \qquad R_f^* = I, \qquad q = 50.$$

There is a more dramatic reduction of errors $x_1 - x_1^*$, $x_2 - x_2^*$ over that for the LQG design of the previous case as indicated in Table 3.1. Of course, the LQG/LTR design is identical to that for the case when $q = 0$. Also, simulations not reported here show that the LQG/LTR design performs virtually identically to an LQ design where states δx are assumed available for feedback.

4. **Adaptive-Q design.**

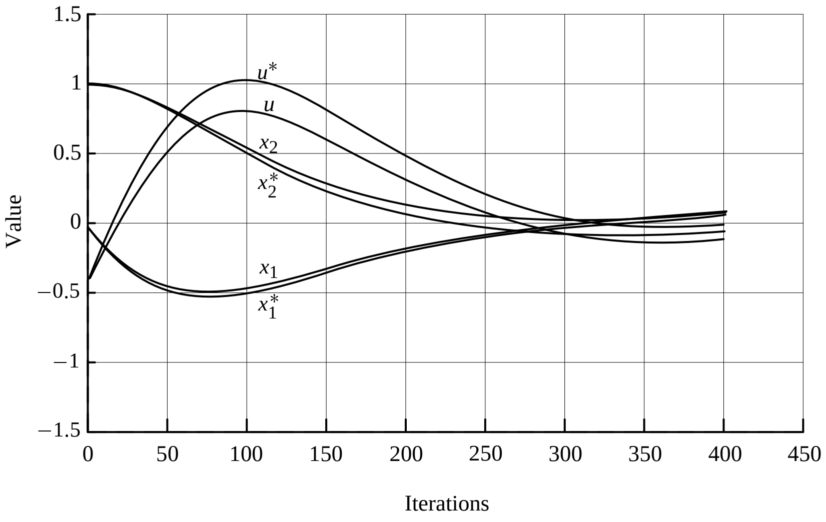


FIGURE 3.4. LQG/LTR/Adaptive-Q Trajectories

Disturbance	$d = 0.2$	$d \in U(0.1, 0.3)$	$d \in U(0, 1)$
Open loop	3.071	3.056	6.259
LQG	0.749	0.744	3.256
LQG/LTR	0.230	0.228	1.348
LQG/Ad-Q	0.348	0.345	1.993
LQG/LTR/Ad-Q	0.160	0.160	1.001

TABLE 3.1. ΔI for Trajectory 1, $x(0) = [\,0\ 1\,]$

The adaptive-Q, two-degree-of-freedom controller design for optimal control problem is studied, with the LQG or LQG/LTR controller K and the adaptive $\square = [\, Q_f \;\; Q \,]$ using least square techniques. Third-order FIR models are chosen for the forward $Q_f(z)$ and the backward $Q(z)$. Simulations, summarized in Table 3.1, show that adaptive-Q controller design strengthens the robustness/performance properties of both the LQG and LQG/LTR design without the need for any high gains in the controller. See also Figures 3.3 and 3.4. The intention in this first design example has not been to demonstrate that an adaptive-Q approach works dramatically better than all others, although one example is shown where such is the case. Rather, we have sought to stress that the adaptive-Q method is perhaps best used only after a careful robust fixed controller design, and then only to achieve fine tuning. Actually, for the design study here, the robust LQG/LTR design performed better than the LQG adaptive-Q design. The values of ΔI for all five cases are summarized in Table 3.1 for a deterministic disturbance $d = 0.2$, and then two stochastic disturbances, in the first instance with d uniformly distributed between 0.1 and 0.3, and in the second with d uniformly distributed between 0 and 1.

To demonstrate the robustness of the adaptive-Q control strategy, the simulations were repeated with unmodeled dynamics in the actual plant, see Table 3.2. The state equations of the actual plant in this case are

$$\begin{aligned}
\dot{x}_1 &= (1 - x_2^2)x_1 - x_2 + x_3 + u,\\
\dot{x}_2 &= x_1,\\
\ddot{x}_3 &= -\dot{x}_3 - 4x_3 + u,\\
y &= x_1,
\end{aligned}$$

with initial state vector $[\, 0 \;\; 1 \;\; 0 \,]$.

The simulations in Tables 3.1 and 3.2 are repeated for different initial conditions, and thus a different optimal trajectory. The results are included in Tables 3.3 and 3.4. This trajectory also has the same unmodeled dynamics added to demonstrate robustness.

Disturbance	$d = 0.2$
Open loop	5.907 7
LQG	1.943 8
LQG/LTR	0.662 3
LQG/Ad-Q	0.984
LQG/LTR/Ad-Q	0.425 1

TABLE 3.2. ΔI for Trajectory 1 with unmodeled dynamics, $x(0) = [\, 0 \;\; 1 \;\; 0 \,]$

Disturbance	$d = 0.2$	$d \in U(0.1, 0.3)$	$d \in U(0, 1)$
Open loop	3.664 6	3.647 8	7.450 2
LQG	0.752 4	0.747 6	3.381 7
LQG/LTR	0.216 5	0.214 8	1.276 6
LQG/Ad-Q	0.344 7	0.341 5	1.973 4
LQG/LTR/Ad-Q	0.150 5	0.149 5	0.927 8

TABLE 3.3. ΔI for Trajectory 2, $x(0) = [\,1\ 0.5\,]$

Disturbance	$d = 0.2$
Open loop	4.730 1
LQG	1.280 5
LQG/LTR	0.516 2
LQG/Ad-Q	0.631 3
LQG/LTR/Ad-Q	0.298 1

TABLE 3.4. ΔI for Trajectory 2 with unmodeled dynamics, $x(0) = [\,1\ 0.5\ 0\,]$

In our simulation for the adaptive-Q controller, two passes are needed for "warming up" of the controller. Subsequently, the coefficients in Q_f and Q, in the notation of (2.21), "converge" to slowly varying values in the vicinity of $\gamma = 0.0976$, $\gamma_1 = -0.0002$, $\gamma_2 = -0.1016$, $\beta = -11.18$, $\beta_1 = -9.247$, $\beta_2 = -7.891$, with $\alpha_i \equiv 0$.

The prefilters $P_{12}M$ used in our study are as follows:

$$\dot{x}_{ps} = (A + BF)x_{ps} + Bu^*, \qquad \xi_1 = \begin{pmatrix} F \\ I \end{pmatrix} x_{ps} + \begin{pmatrix} I \\ 0 \end{pmatrix} u^*,$$

with input u^* and output ξ_1. Likewise for the prefilters driven by δr and s.

Our simulations not reported here show significant improvements when scaling adjustments are made to r and e. Also, other simulations not reported here show that there is insignificant benefit with increasing the dimensions $p = 3,\ m = 3,\ n = 0$ in Q, although the cost of reducing p or m is significant.

The message from this simple example is clear. Open-loop optimal control of nonlinear systems is nonrobust. Linearization and application of linear optimal feedback methods improves the situation. Working with robust linear methods is even better, and adding an adaptive-Q loop enhances performance and robustness even further.

8.4 Learning-Q Schemes

For robust nonlinear optimal control, it makes sense to explore the adaptive-Q enhancement approach modified such that Q is state (or state estimate) dependent. Such an approach is termed here learning-Q control.

In generalizing the least squares based adaptive-Q filter coefficients update scheme, applied for the nonlinear control tasks of the previous sections, the essence of our task is to replace a parameter least squares algorithm by one involving *functional learning*. To optimize the Q-filter when its parameters are state dependent, the key strategy we propose here is to apply functional learning algorithms as in Perkins, Mareels and Moore (1992). The least squares based functional learning of Perkins et al. (1992) suggests bisigmoid sum representations of the parameters over the state space, or for practical reasons, over only the significant components of the state space. The parameters of this representation are tuned on line by a least squares scheme. Bisigmoids, such as Gaussians or truncated Gaussians, B-splines, or radial basis functions have the dual roles of interpolating between parameter estimates at grid points in the state space, and of spreading learning on either side of a trajectory in the state space.

In adaptive control, the controller for the plant adapts to optimize some performance specification. Should the plant be time-varying, then the adaptive controller tracks in some sense an optimal controller. There is built into the controller a *forgetting factor* so that distant past experiences are totally forgotten. If the plant dynamics are nonlinear being a function of a slowly changing variable , such as the slow states of the plant, then it makes sense to remember the past in such a way that the controller can recall appropriately from past experience and give better performance than it would with built-in forgetting. To facilitate such control action, enhanced with memory, functional learning algorithms appear attractive.

Functional learning here refers to a method by which the values of a function $y = f(x)$ can be estimated at all points in the input variable space Γ_x from data pairs (x_i, y_i), or noisy measurements of these. Given an estimate $f(\cdot)$ at time k, denoted $\hat{f}$, then with a new measurement x_k, a prediction of y_k is $\hat{y}_k = \hat{f}(x_k)$. The error $(y - \hat{y}_k)$ can then be used to update $\hat{f}(\cdot)$ for time $k+1$ based on assumed smoothness properties of $f(\cdot)$. The function is here represented as a sum of simply parameterized functions which could be basis functions such as polynomials or Gaussians. We define the error between a representation of a given function $\hat{f}(\cdot)$ and the actual function $f(\cdot)$ in terms of some error norm. Thus, here, the learning involves adapting the parameters of the basis functions to achieve lower error norms. We look to approximate arbitrary continuous functions within a class of such.

A key representation theorem for our approach is in Cybenko (1989). This theorem tells us that sums of sigmoids, or more general bisigmoids such as Gaussians or truncated Gaussians, suitably parameterized, are dense and can represent functionals over finite domains with arbitrary precision.

Function Representation

Given a function, $f(\cdot)$, an approximation to that function could be represented by a superposition of a finite number of the simply parameterized functions $f_i(\cdot)$, such as sigmoids or bisigmoids, each centered at different points γ_i ,within the input variable Γ_x space. The representation must be chosen with regard to required accuracy, convergence properties and computability. For example, to approximate a two input variable scalar function with a bounded first derivative by a grid of simply parameterized functions being piecewise constant functions on a grid, the following result is easily established.

Lemma 4.1. *Suppose there is given a two input variable scalar function $f(x, y)$ with a first derivative bounded by C, and a square region $R = \{(x, y) : |x|, |y| < r\}$ over which it is to be approximated. Furthermore, suppose there is an approximation to the function by a piecewise constant function on a rectangular $N \times N$ grid covering the region R. Then the ℓ_2 error bound between $f(x, y)$ and the approximation $\hat{f}(x, y)$ is*

$$b = O(\frac{C^2}{N^2}). \tag{4.1}$$

Proof.

$$b = \sum_{N^2} \int_{y_0}^{y_0+r/N} \int_{x_0}^{x_0+r/N} (\hat{f}(x, y) - f(x, y))^2 dx\, dy. \tag{4.2}$$

The worst case approximation to $f(x, y)$ of a level function will be when $f(x, y)$ is at its maximum gradient within a grid square. Also, the worst error will be the same for each grid square. To calculate a worst case error then, consider the "worst case" functions $f(x, y) = f_0 \pm Cx \pm Cy$. In this case,

$$b = N^2 \int_{y_0}^{y_0+r/N} \int_{x_0}^{x_0+r/N} (\hat{f}(x, y) - f(x_0, y_0) \pm Cx \pm Cy)^2 dx\, dy. \tag{4.3}$$

Now, the region of that square is $\{x, y : x_0 < x < x_0+r/N, y_0 < y < y_0+r/N\}$. Set $\hat{f} = f(x_0, y_0)$. Then a substitution of variables gives

$$b = \frac{7}{6} N^2 C^2 (\frac{R}{N})^4 = O(\frac{C^2}{N^2}). \tag{4.4}$$

□

In selecting the simply parameterized functions $f_i(\cdot)$ for learning in the control environment of interest, we take into account the need for 'fast' learning, reasonable interpolation and extrapolation approximation properties, and the ability to spread learning in the Γ_x space. For 'fast' learning, we require here that the measurements are linear in the parameters of $f_i(\cdot)$ so that the least squares techniques can apply. This contrasts the case of neural networks where backward propagation gradient algorithms are inevitably ‘slow’ in convergence.

For reasonable interpolation capabilities, any of a number of selections such as polynomials, splines, sigmoids, or bisigmoids can be used, but to avoid poor extrapolation outside the domains of most excitation of x_k in Γ_x, we select bisigmoids. Likewise, only the bisigmoids allow learning along the trajectories in Γ_x to be spread acceptably to neighborhoods of such trajectories. This approach is taken in Perkins et al. (1992) for an open loop identification task, which we now adapt for our control task. First let us review the least squares learning taken in Perkins et al. (1992), and related results.

Least Squares Learning

Consider the signal model usually derived from an ARMAX representation,

$$y_k = \Phi_k' \Theta(x_k) + \omega_k, \tag{4.5}$$

where y_k are the measurements, Φ_k is a known regression vector of the model inputs and outputs, and $\Theta(\cdot)$ are the unknown functionals with input variables x_k, representing the perhaps nonlinear ’parameters’ of an ARMAX model. Here ω_k is taken to the zero mean white noise.

Let us investigate finite representations estimating $\Theta(x)$ of the form

$$\hat{\Theta}(x) = \sum_{i=1}^{n} K_I'(x, \gamma_i)\hat{\theta}(\gamma_i) = K_I'(x)\hat{\Theta}(\Gamma_I), \tag{4.6}$$

with $\hat{\Theta}'(\Gamma_I) = [\hat{\theta}'(\gamma_1)\ldots\hat{\theta}'(\gamma_n)]$, and $K_I'(x) = [K_I'(x,\gamma_1)\ldots K_I'(x,\gamma_n)]$. Here $\theta(\gamma_i)$ are the parameters and $K_I(x,\gamma_i)$ the interpolation function representation (4.6). For simplicity we work with $K_I(x,\gamma_i)$ which acts as both a scalar interpolating function and learning spread function between the points $x \in \Gamma_x$ and $x \in \Gamma_I$ with a preselected set of points $\Gamma_I = \{\gamma_1, \gamma_2, \ldots, \gamma_n\}$ in Γ_x. In our simulations, we use Gaussians or truncated Gaussians for the vectors indicated above. Now (4.5) can be represented as

$$y_k = \Phi(x_k)'\Theta(\Gamma_I) + \hat{\omega}_k, \tag{4.7}$$

where

$$\Phi(x_k) = K_I(x_k)\Phi_k, \qquad \Theta(x_k) = K_I(x_k)\Theta(\Gamma_I), \tag{4.8}$$

and $\hat{\omega}_k$ approximates ω_k.

Consider an error measure for the representation:

$$d_2^{(r)}(\hat{\Theta}) = \frac{1}{r}\left[\sum_{k=1}^{r} \left\| \Theta(x_k) - \hat{\Theta}(x_k) \right\|^2\right]^{1/2}. \tag{4.9}$$

As shown in Perkins et al. (1992), under strong persistence of excitation conditions on x_k minimization of this index is equivalent to minimization of the d_2 index :

$$d_2(\hat{\Theta}) = \left[\int_{\Gamma_x} \left\| \Theta(x) - \hat{\Theta}(x) \right\|^2 dx \right]^{1/2}. \tag{4.10}$$

A key result associated with this latter minimization task is as follows:

Theorem 4.2. *The minimization task has a unique critical point, denoted $\hat{\Theta}^*$if and only if the elements of $K_I(x)$ are allowable, in that*

$$\infty > \left[\int_{\Gamma_x} K_I(x) K_I'(x) dx \right] > 0. \tag{4.11}$$

This optimal $\hat{\Theta}$ is given from

$$\hat{\Theta}^* = \left(\int_{\Gamma_x} K_I(x) K_I'(x) dx \right)^{-1} \int_{\Gamma_x} y(x) K_I'(x) dx. \tag{4.12}$$

Moreover, when $\Theta(x)$ is reconstructible with respect to the class of functions $\hat{\Theta}(x)$ of (4.6), *then $\Theta(x)$ is uniquely parameterized as in* (4.6) *with $\Theta = \hat{\Theta}^*$ given in* (4.12).

Proof. The proof of this can be found in Perkins et al. (1992). $\square$

In order to minimize $d_2^{(r)}$ of (4.9) for $r = 1, 2, \ldots,$ given a sequence $\{x_k, y_k\}$, standard least squares derivations applied to (4.9) and (4.10) lead to a recursive estimate of $\Theta(x_k)$, denoted $\hat{\Theta}_k(x_k)$, as

$$\hat{\Theta}_k(x_k) = K_I'(x_k) \hat{\Theta}_k(\Gamma_I) \tag{4.13}$$

$$\hat{\Theta}_k(\Gamma_I) = \hat{\Theta}_{k-1}(\Gamma_I) + P_k(\Gamma_I) \Phi_I(x_k) [y_k - \Phi_I'(x_k) \hat{\Theta}_{k-1}(\Gamma_I)], \tag{4.14}$$

where

$$P_k^{-1}(\Gamma_I) = P_{k-1}^{-1}(\Gamma_I) + \Phi_I(x_k) \Phi_I'(x_k), \qquad \Phi(x_k) = K_I(x_k) \Phi_k, \tag{4.15}$$

with suitable initial conditions $\hat{\Theta}$, P. Under appropriate conditions (Perkins et al. (1992)), P_k^{-1} approaches a diagonal matrix. With truncated K_I we have that P_k^{-1} is block diagonal with only one block updated at each iteration and only one corresponding segment of $\hat{\Theta}_k$ updated. In selecting a truncation, there is clearly a trade off between function estimation accuracy and computational effort.

Least Squares (Scalar variable case)

Now with the definitions of (2.20), and as shown in Tay and Moore (1991) for the case of scalar variables $\delta u_k, \delta y_k, r_k, b_k$ (for simplicity), with $s_k = Q_f u_k^* + Q r_k$, following the derivations leading to (2.24) we see that ζ_k is linear in Θ, as

$$\xi_k = \Phi_k' \Theta + e_k, \tag{4.16}$$

$$\Phi_k' = \Big[(\hat{e}_{k-1/k-1} - \zeta_{k-1}) \quad \dots \quad (\hat{e}_{k-n/k-n} - \zeta_{k-n}) \quad -\xi_k \quad \dots \quad -\xi_{k-m} \Big]. \tag{4.17}$$

These equations allow a least squares recursive update for Θ, denoted $\hat{\Theta}_k$, to minimize the index $\Sigma_{i=1}^{k} \left\| e_{i|\Theta} \right\|^2$ as spelled out earlier for the adaptive-Q scheme. Here $e_{i|\Theta}$ denotes $e_{i|\hat{\Theta}_1 \dots \hat{\Theta}_{k-1}\Theta}$ in obvious notation. In fact the details are a special case of those now derived for the proposed learning-Q scheme of Figure 4.1. Note that the adaptive-Q scheme is that of Figure 4.1 specialized to the case when $\hat{\Theta}_k(x^*)$ is independent of x^*, so that $\hat{\square}_k(x_k^*)$ is replaced by $\hat{\square}_k$.

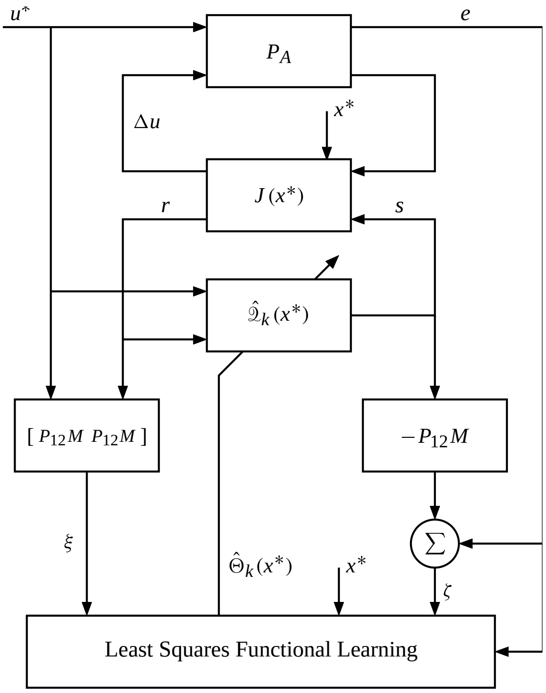


FIGURE 4.1. Two degree-of-freedom learning-Q scheme

Learning-Q Scheme based on x^*

We build upon the earlier work of Chapter 6 and this chapter by extending the adaptive-Q scheme to what could be viewed as an adaptive-$Q(x^*)$ scheme, but which we call a learning-Q scheme. The $\square$ filter with which we work has coefficients which are functions of the state space. Denoting $\hat{\square}_k(x)$, $\hat{\alpha}_{ik}(x)$, $\hat{\beta}_{ik}(x)$, the key idea of the learning-Q algorithm is to update estimates $\hat{□}_k(\cdot)$, for all x in some domain Γ_x in which x_k^* lies. The □ filter is implemented as $\hat{□}_k(x_k^*)$. The arrangement is depicted in Figure 4.1. The least squares functional learning block yields estimates $\hat{\Theta}(x_k^*)$ for implementation of the filter $\hat{□}_k(x_k^*)$, being driven from ζ_k, ξ_k, e_k and x_k^*. The parameter estimates $\hat{\Theta}_k(x_k^*)$ are derived via the approach above. Thus , corresponding to (4.7) , we have the formulation (4.16) , and corresponding to (4.13)–(4.15) we have

$$\hat{\Theta}_k(x_k^*) = K_I'(x_k^*)\hat{\Theta}_k(\Gamma_I), \tag{4.18}$$

$$\hat{\Theta}_k(\Gamma_I) = \hat{\Theta}_{k-1}(\Gamma_I) + P_k(\Gamma_I)\Phi_I(x_k^*)\left[\xi_k(x^*) - \Phi_k'\hat{\Theta}(\Gamma_I)\right], \tag{4.19}$$

$$P_k^{-1} = P_{k-1}^{-1}(\Gamma_I) + \Phi_I(x_k^*)\Phi_I'(x_k^*). \tag{4.20}$$

The □ filter can be implemented as $\hat{□}_k(\hat{x}_k)$, where $\hat{x}_k = x_k^* + \delta\hat{x}_k$, as an alternative to implementing $\hat{□}_k(x_k^*)$. This suggests also that the nominal plant ΔG, controller K and indeed J be functions of $\hat{x}_t$, rather than x_t^*. For this case then, any $\Delta\bar{G}(\hat{x}_t)$, $K(\hat{x}_t)$, $J(\hat{x}_t)$ are inevitably nonlinear and a nonlinear factorization theory is required. Care should be taken because there is in essence an extra feedback loop in the system which can cause instability, so this approach is not recommended.

Of course when δx is small, one expects that x_t^* could be just as good an estimate of x_t as $\hat{x}_t$. In this case, there would be an advantage in working in a full nonlinear context. However, we would expect δx to be small only when the plant is nearly linear, and to avoid dealing with a full nonlinear context is really to avoid tackling systems that are in essence nonlinear.

Learning-Q Simulation Results

The Signal Model and Performance Index

Consider again the specific nominal plant model (Van der Pol equation):

$$G_0 : \dot{x}_1 = (1 - x_2^2)x_1 - x_2 + u + d; \qquad \dot{x}_2 = x_1, \qquad y = x_1, \tag{4.21}$$

with scalar input u, scalar output y, and state vector $x = [\, x_1 \; x_2 \,]'$. Consider also a regulator performance index defined by

$$I(x_0, u) = \frac{1}{2}\int_0^5 (x_1^2 + x_2^2 + u^2)dt. \tag{4.22}$$

Of course for such a simple example, one could use a trivial linearization, by taking $u = \hat{x}_1\hat{x}_2^2 + u_1$ and then optimizing $u_1 = K\hat{x}$ via LQG control, thereby achieving an attractive nonlinear controller. Also, for more general nonlinear systems one could similarly exploit the linearization approach of Isidori (1989). However, here we wish to illustrate the design approach of the previous sections and use this plant as an example. Here, for each initial condition investigated, an optimal trajectory is calculated. Next, the closed-loop feedback controller schemes of the previous section are studied in the presence of constant and stochastic disturbances added to the plant input, and some of the simulations include unmodeled dynamics. The actual plant with unmodeled dynamics is

$$\begin{aligned}\dot{x}_1 &= (1 - x_2)x_1 - x_2 + x_3 + u + d,\\ \dot{x}_2 &= x_1,\\ \ddot{x}_3 &= -\dot{x}_3 - 4x_3 + u,\\ y &= x_1,\end{aligned} \tag{4.23}$$

where x_3 is the state of the unmodeled dynamics and d is the disturbance.

The major objective of the learning-Q method is to learn from one trajectory, or set of trajectories information which will enhance the control performance for a new trajectory.

For the simulations, functions $\Theta(x) = \Theta(x_1, x_2)$ are represented as the sum of equal covariance Gaussians centered on a sparse two-dimensional grid in Γ_{x_1,x_2} space. In the learning-Q scheme, the weighting of the Gaussians is updated to best fit the given data via least squares functional learning. The tailing off of the Gaussians will ensure that trajectory information effectively spreads only to the neighboring grid points with more weighting on the near neighbors.

Selection of Algorithm Parameters

The algorithm requires a priori selections of the interpolating functions and their parameters including location on grid points in state space. It then learns the relative weightings.

Selection of Grid Points

The state space of interest is selected to include the space spanned by the optimal trajectories. The Gaussians are fixed initially at a 4×4 grid $of \gamma_i$ over a unit "box" region covering well the trajectory region to avoid distortions due to edge effects. A scaling method facilitates quick changes of the apparent denseness and compactness of the grid. Optimal placing of the grid points for one particular trajectory is not usually optimal for other trajectories, and the chosen grid points can be seen in Figure 4.2.

The Spread of the Interpolating Function

The interpolating function is of the form $We^{-n^2 d_i^2 k}$, where n is the number of Gaussians in each dimension, d_i is $\|x - \gamma_i\|^2$, W is the initial weighting, and k is a constant used to tune the learning (in our simulations $W = 10^{-10},\ k = 4$). The "optimal" spread of the Gaussian is a function of the shape of the function being learned, and the denseness of the Gaussians in the grid.

Trajectory Selection

The data must be "persistently" spanning the grid space in order to learn all the Gaussian weights. As the estimates are functions of stochastic outputs, greater

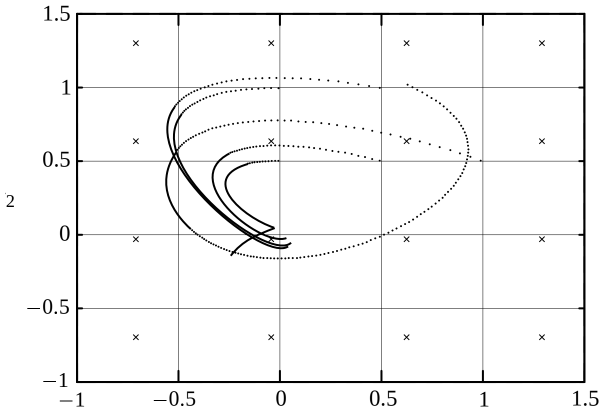


FIGURE 4.2. Five optimal regulation trajectories in Γ_{x_1,x_2} space

excitation of a mode allows for a more accurate estimate of the weighting of that mode. Five initial conditions have been chosen to illustrate performance enhancement due to the learning-Q approach. The initial conditions of x are: $(0.5, 1)$, $(0.5, 0.5)$, $(1, 0.5)$, $(0, 0.5)$, $(0, 1)$. The optimal regulation state trajectories calculated from these initial conditions are shown in Figure 4.2, to indicate the extent to which the state space Γ_x is covered in the learning process.

Results

For the trajectories and algorithm parameters of the previous subsection, robustness and performance properties are examined.

Persistence of Excitation

In order to test the learning, two simulations were compared. In the first, denoted *Global Learning*, Trajectories 1 through 5 were executed and then repeated, each time enhancing the learning with the knowledge previously learned. In the second, denoted *Local Learning*, Trajectory 5 was repeated 10 times, to achieve enhanced learning. In the example of Table 4.1 below, the disturbance was $d = 0.2$.

Run Number	Global Learning	Local Learning
5	0.337 3	0.342 2
10	0.329 6	0.335 5

TABLE 4.1. Error index for global and local learning

As can be seen in Table 4.1, the global learning actually gives marginally better results than the specialized local learning by virtue of its satisfying persistence of excitation requirements. These are typical of other simulations not reported here.

Deterministic Disturbances

There are three types of disturbances simulated. First a zero disturbance is used, and since the plant then follows the optimal trajectory, as expected the values of δx and δu are zero. The other disturbances used are: a constant $\omega = 0.2$, stochastic disturbances uniformly distributed, and disturbances where the disturbance is a function of position in state space.

Since the error index is a function of the total level of input disturbances, the constant disturbance is the one used to compare various parameters and methods, with the stochastic and functional disturbances being then used to test the selected parameters under more realistic conditions.

Stochastic Disturbance

The simulation is run for each trajectory with $d = \mathrm{RAND}(-0.5, 0.5)$, that is the disturbance is uniformly distributed with an upper bound of 0.5, and a lower bound of −0.5. In the case of 'global learning', where Trajectories 1 through 5 were run, then repeated for all of the trajectories, the algorithm gives an improvement in the error index, except Trajectory 5. Details are summarized in Table 4.2.

Trajectory	Run 1	Run 2	Improvement
1	0.015 7	0.008 6	45%
2	0.016 0	0.007 0	56%
3	0.063 7	0.021 3	66%
4	0.070 8	0.012 3	83%
5	0.011 3	0.019 9	−5%

TABLE 4.2. Improvement after learning

Unmodeled Dynamics

The simulations were run with the disturbances as before, as well as with the inclusion of the unmodeled dynamics as in (4.23). The system achieved good control, with for example the error indices of Trajectories 1 through 5, then repeated in a stochastic disturbance case being 2.8, 0.9, 1.5, 1.1, 1.6 and the second run giving 1.7, 1.0, 1.5, 1.1, 1.7.

Nearest Neighbor Approximation and Grid Size

The standard algorithm requires $O(n^2)$ iterations for calculation of the values of an $n \times n$ grid. An approximation can be made where only the weights of the closest Gaussians are updated at each step. This approximation significantly speeds the algorithm, but loses accuracy. As is shown in Table 4.3, for the case with a stochastic disturbance $d = \text{RAND}(-0.5, 0.5)$ as above, and with no unmodeled dynamics, however going to a finer grid of 5×5 Gaussians, with the nearest neighbor approximation/truncation improves upon the 4×4 full calculation/untruncated case.

Average	4×4 truncated	4×4	5×5 truncated
Run 1	0.397 0	0.384 4	0.382 8
Run 2	0.345 8	0.345 1	0.330 4

TABLE 4.3. Comparison of grid sizes and approximations

As expected, the finer grid improves in comparison to the others during the second run, as it is better able to fit to the information given. The Θ_f surfaces generated in this simulation for the 2×2, 3×3, 4×4 and 5×5 cases for the constant disturbance $d = 0.2$ are shown in Figure 4.3. The error index averaged over the second run for these cases are respectively, 0.320 4, 0.325 2, 0.346 5, 0.323 0. The 4×4 case gave the worst result, possibly due to the artifact on the surface not displayed by the others. These results show that a finer grid spacing may not always improve the control.

Comparison with Adaptive case

Let us compare the results of our learning controller with those of the adaptive-Q controller . In the first instance, when running a trajectory for the first time, the adaptive algorithm gives better results than the learning algorithm. Allowing the learning algorithm previous experience on other trajectories however, lets it in most cases "beat" the adaptive one. For instance, in the nonzero mean stochastic disturbance case $\{\omega \in 0.2 \pm 0.1\}$, with no unmodeled dynamics, for Trajectory 1, the adaptive case gave 0.490 7, with the learning case giving 0.367 1 after learning on some trajectories.

However, the adaptive case can also be enhanced to give better performance than the learning scheme by exploiting previous information from a learning-Q scheme. Simulations are performed comparing the extended adaptive to the learning scheme with a variety of disturbances and with/without unmodeled dynamics. The constant, stochastic, and nonconstant deterministic disturbances are $d = 0.2$, $d \epsilon 0.2 \pm 0.05$, $d = x_1^2 + x_2^2$. The results are summarized in Tables 4.4 and 4.5.

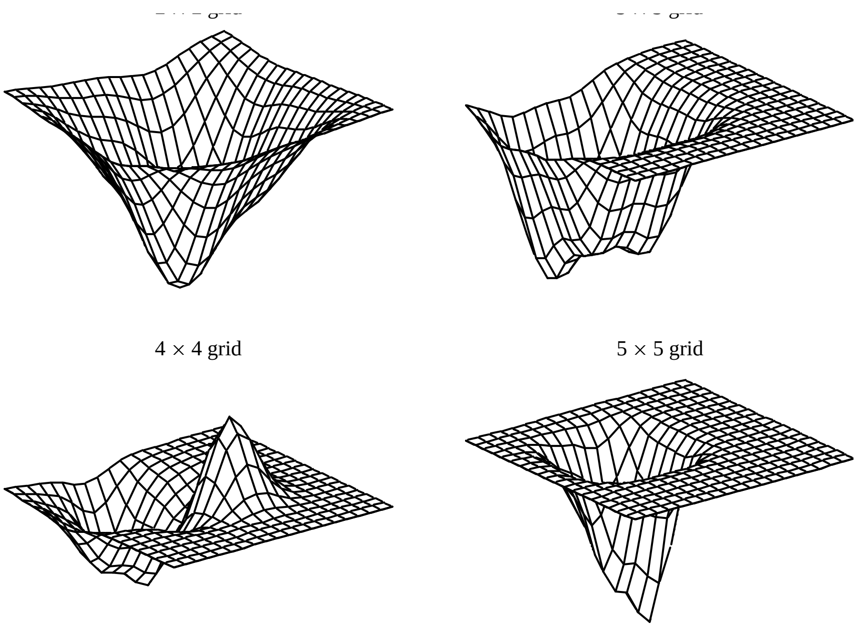


FIGURE 4.3. Comparison of error surfaces learned for various grid cases

Disturbance	Learn	Adapt	Improvement
constant	0.365 1	0.288 5	21%
stochastic	0.368 4	0.283 9	28%
deterministic	1.884 6	1.124 8	40%

TABLE 4.4. Error index averages without unmodeled dynamics

Disturbance	Learn	Adapt	Improvement
constant	1.504 4	0.989 1	34%
stochastic	1.492 5	0.982 5	34%
deterministic	2.460 8	2.029 3	18%

TABLE 4.5. Error index averages with unmodeled dynamics

Main Points of Section

Learning-Q schemes are really adaptive-Q schemes with Q being a parameterized nonlinear function of the state. Adapting its parameters on line achieves functional learning. In our example, the benefit of a learning-Q scheme over an adaptive-Q scheme would be worth the computational cost only in performance critical applications.

8.5 Notes and References

The application of the direct adaptive-Q techniques to nonlinear systems was first studied in Imae et al. (1992) The generalization to learning-Q schemes was first studied in Irlicht and Moore (1991). At the heart of this work is some form of functional learning. The application of parameter update techniques to functional learning theory is of course the meat of *neural network* theory. In these, the parameterizations are fundamentally nonlinear. Here in contrast and for simplicity our focus has been on nonlinear functions which are *linear* in their parameters, such as in a Gaussian sum representation, as explored by us in Perkins et al. (1992). Thus here our learning-Q filter, based on Irlicht and Moore (1991), uses such representations.

This work on adaptive-Q and learning-Q techniques provided motivation for us to understand further the theory of nonlinear fractional maps and nonlinear versions of the Youla-Kučera parameterizations, see Paice, Moore and Horowitz (1992), Perkins et al. (1992) and Moore and Irlicht (1992).

One key insight to cope with nonlinear systems is that in the nonlinear case where the operators are initial-condition dependent, the mismatch between those of estimators/controllers and those of the plant can be viewed as unmodeled dynamics, namely as a nonlinear operator S.

Another observation which helps explain much of the scope of the nonlinear theory is that the class of all stabilizing controllers for nonlinear plants is more easily characterized in terms of left coprime factorizations, yet these factorizations are much harder to obtain than right coprime factorizations.

CHAPTER 9

Real-time Implementation

9.1 Introduction

We have until this point, developed the theory and algorithms to achieve high performance control systems. How then do we put the theory into practice? In this chapter, we will examine real-time implementation and issues in the industrial context. In particular we will discuss using discrete-time methods in a continuous-time setting, the hardware requirement for various applications, the software development environment and issues such as sensor saturation and finite word length effects. Of course, just as control algorithms and theory develop, so does implementations technology. Here we focus on the principles behind current trends, and illustrate with examples which are contemporary at the time of writing.

We begin with the hardware requirements. The hardware for a modern control system is microprocessor-based. Analog-based control systems are still used in some speed critical applications, and where reliability is already established and is critical. However the cost in implementing complex algorithms using analog circuits with present day technology is not competitive with the new digital technology available. With modern fast microprocessors and *digital signal processors* (DSPs), there is the possibility of connecting many processors to work in a parallel fashion, and it is now possible to implement fairly complex algorithms for sampling intervals of microseconds. The speed consideration is therefore becoming less important and digital implementation dominates.

We begin the chapter with a discussion on using discrete-time methods for continuous-time plants. The key theorem is the *Nyquist Sampling Theorem*.

Our next task is the design of a stand-alone microprocessor-based control system. Design and implementation of such a system can be time consuming in practice. This is because there is very little scope for incremental testing and verification in the initial stage of the design. The engineer has to get both the hardware and software working at the same time in order to verify that the design is successful. Often when the system fails to work, the engineer is left wondering whether the bug is in the hardware or the software. In response to this difficulty, many microprocessor manufacturers are now marketing emulator sets that essentially provide a software development and debugging environment. However such sets tend to be very expensive and are available for popular microprocessor series only. Also, there may be advantages in choosing certain specific processors to exploit their on-chip facilities for a particular application. In such cases, we may be left developing the controller system without the aid of emulator sets. We will outline in this chapter a systematic procedure that will greatly simplify the task of developing a stand-alone system without the aid of an emulator set. Design examples are also provided.

To cope with the rather complex algorithms introduced in this book, there is a need to look into high performance hardware platforms. We will describe in this chapter a two-processor design involving a number crunching oriented DSP and an input/output oriented *microcontroller*. We will highlight the modular approach adopted to facilitate easy development of both the hardware and the software.

It is well understood that the cheapest systems are ones which use the least number of components and at the same time use components that are commonly available. Take for example, a personal computer mother board which implements a feature-rich complex computing environment. This can be purchased for a relatively low price because the personal computer mother boards are produced by the millions and great effort is made to reduce component cost. Also with highly automated assembly lines, possible only with high volume production, the per unit manufacturing cost of such a board is very low. A comparable customized system with production volume of only a few thousands would be many times more costly.

The plunging cost of mass-produced electronic boards gives rise to some new thoughts on designing the hardware of a control system. The trend is the design and manufacture of modules that are feature-rich so that they can be mixed and matched into systems for use in many applications. This allows the production volume of each module to be high and therefore drives the per unit cost down. An immediate implication is therefore to design a universal input/output interface card that has many features to cater to all kinds of controller needs. The specific control system is then assembled from ready-made personal computer mother boards and the input/output interface card. We will describe the design of one such card in this chapter.

Development of the software for control algorithms is a parallel area for attention alongside hardware development. For simple controllers, this is not an issue. However, the high performance controllers studied in this text are complex enough for software aspects to be important. In this chapter we will look at the various platforms for which such software can be developed easily. The intention is to aid the designer to take steps towards realization, or at least to be able to talk to the implementation specialist.

9.2 Algorithms for Continuous-time Plant

For the majority of controller designs introduced in this book, plant models which are discrete-time, linear, time-invariant models are used, and consequently, the resulting controllers are discrete-time, linear, and time-invariant controllers. The discrete-time setting is desirable since we believe that in virtually all future controller realizations, digital computers and discrete-time calculations will be used. The discrete-time model of (2.2.3) can be directly derived from the underlying process in some plants, or indirectly by an identification scheme. Of course, for most plants, the underlying process is analog in nature, and indeed corresponding results to those spelled out in this book can be derived for continuous-time models, with little difficulty. How then do we deal with continuous-time plants when our approach is via discrete-time theory and algorithms?

For continuous-time plants and digital computer controller implementation, there are two approaches. In the first approach, a continuous-time controller design is based on the continuous-time plant model. The continuous-time controller is then replaced by an approximated discrete-time controller for implementation on a digital computer with appropriate analog-to-digital, and digital-to-analog converters. However in general, a relatively fast sampling rate has to be used in practice in order to achieve a good discrete time approximation. For recent work in this direction, see Blackmore (1995).

An alternative approach is to first derive a discrete-time plant model from the continuous-time plant with its analog-to-digital and digital-to-analog converters, *anti-aliasing* filters, and post sampling filters attached, and then perform a discrete-time controller design based on the discrete-time model of the plant, see Figure 2.1. This approach can lead to a possible reduction in the sampling rate compared to the former approach, see Åstrom and Wittenmark (1984). We will adopt this approach in the treatment of continuous-time plants.

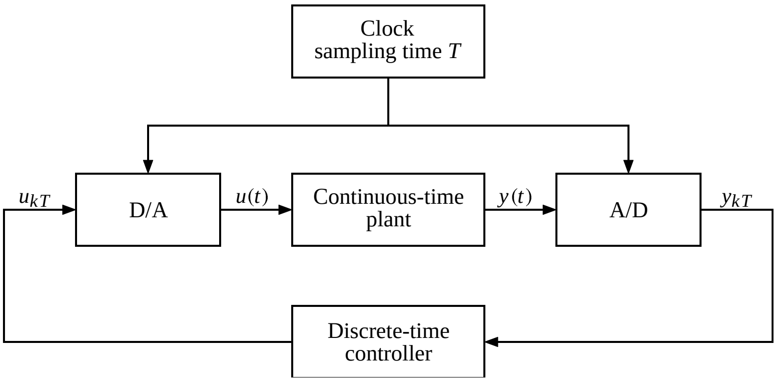


FIGURE 2.1. Implementation of a discrete-time controller for a continuous-time plant

Let's consider the following linear, time-invariant continuous-time plant model.

$$\begin{aligned} \dot{x}(t) &= A_c x(t) + B_c u(t), \\ y(t) &= C_c x(t) + D_c u(t). \end{aligned} \tag{2.1}$$

The pairs (A_c, B_c) and (C_c, A_c) are assumed stabilizable and detectable, respectively.

This continuous-time plant model (2.1) can be discretized in time by an appropriate sampling process. For the moment, assume that the same sampling interval is used for both the inputs and outputs of the plant and *zero order holds* are inserted at each input to the plant to achieve digital-to-analog conversion. This arrangement is depicted in Figure 2.1.

For the continuous-time plant (2.1) embedded in the scheme of Figure 2.1, with sampling time T, the discrete-time model (2.2.8) then has state space matrices as follows.

$$A = e^{A_c T}, \qquad B = \int_0^T e^{A_c s} ds B_c, \qquad C = C_c, \qquad D = D_c. \tag{2.2}$$

With this discrete-time model, discrete-time controller design techniques can be applied and the control scheme implemented as in Figure 2.1. In this arrangement, the control signal derived from the controller accesses the plant through the digital-to-analog converters.

It is also possible to design discrete-time controllers directly for continuous-time plants. These methods are based on a lifting idea that exploits the periodic nature of the sampled data controller when applied to a continuous-time system, Feuer and Goodwin (1996). Alternatively one can work with fast sampling model representations of the plant while performing the design at lower sampling periods, Keller and Anderson (1992), Madievski, Anderson and Gevers (1993), Blackmore (1995), Gevers and Li (1993), Williamson (1991). We do not explore these approaches here.

9.3 Hardware Platform

In this section we explore a microcontroller-based solution, a dual processor solution for fast processes, and a personal computer based solution for achieving controller hardware platforms.

A Microcontroller-based Solution

The block diagram of a typical real-time controller system is shown in Figure 3.1. The system can be divided into three subsystems. They are the computing section, the input/output interface subsystem and the host computer interface subsystem.

The main subsystem of the controller is for computing. It consists of the target microprocessor, nonvolatile memory such as *Erasable Programmable Read-Only*

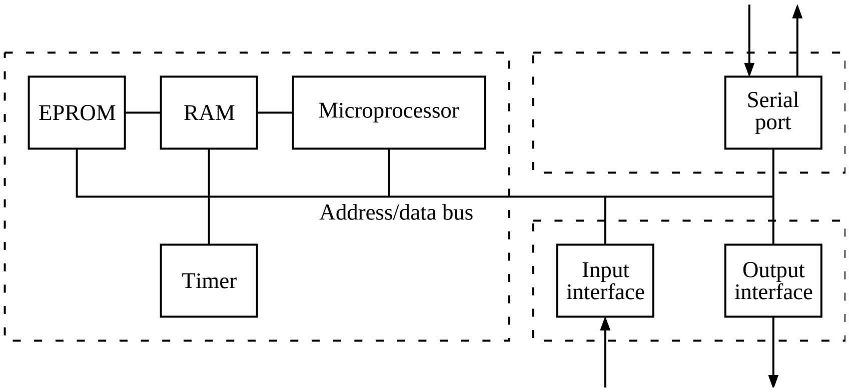


FIGURE 3.1. The internals of a stand-alone controller system

Memory (EPROM), volatile memory such as *Random Access Memory* (RAM) and a timer circuit. This subsystem is responsible for executing the software codes of the control algorithm as well as controlling all the other devices connected to the system. The software codes that implement the control algorithm are stored in the EPROM. The size of the EPROM will depend on the complexity of the algorithm implemented. The RAM is required to store all temporary variables in the execution of the control algorithms. For the implementation of a simple controller scheme such as the ubiquitous *proportional plus integral plus differential* (PID) controller, a RAM size of about 20 bytes is sufficient. For more complex algorithms, the size increases correspondingly. The timer circuit is included to allow implementation of the sampling time interval.

The input/output subsystem consists of interface circuits to allow the microprocessor to communicate with sensors and actuators of the plant. The exact circuit will depend on the sensors and actuators used. In the case of industrial processes, where the sensor instruments usually deliver sensor readings in the form of analog signals in the range of 4 mA to 20 mA, the input interface circuits then consist of analog-to-digital converters and current-to-voltage converters. In the case of robotic applications, where the sensor instruments are optical encoders that deliver quadrature phase pulses, the input interface circuits consist of *up/down counters*.

As for actuators, industrial processes usually have actuators that accept analog signals in the range of 4 mA to 20 mA. The output interface circuit are then digital-to-analog converters together with voltage-to-current converters. On the other hand, in servo-mechanism type processes where DC motors are used, the interface circuits are required to generate *pulse-width-modulated* (PWM) signals.

The third subsystem is the host computer interface section. This subsystem implements a communication link to a host computer. As discussed later, this communication link which can be serial or parallel will allow us to implement a program development environment as well as providing a *data logging* facility.

Component Selection

For low volume production, the percentage of the development cost attributed to each unit is high. It is therefore important that a suitable platform be chosen such that the development cost is minimal. On the other hand, for high volume production, the percentage of the development cost attributed to each unit is very low. It is then advisable to use cheap components at the expense of greater development effort and cost. In such cases, multiple components are often integrated into a single chip, and firmware programs are hand optimized to achieve short code length and high speed of operation. Balancing the various trade-offs for a good design is an art as well as a science.

The key component in the hardware design is the microprocessor. There are at least three considerations; namely the computational requirement, the complexity of the software to be developed and the type of peripherals required to interface to the sensors and actuators of the plant.

To begin with, the microprocessor selected must be powerful enough to complete all the control algorithm computations required within the sampling interval. It might be thought that a good gauge of how much computational effort is required can be obtained by taking time needed to execute programs of similar complexity on a general purpose computer of known power, and to take into account differences in the average efficiency of compiled code on the general purpose computer and that of the targeted processor.

The second consideration is the complexity of the program to be developed. If the program to be developed is very complex, then it is advisable to select a commonly used microprocessor. This is because commonly used microprocessors usually have many supporting tools, either from the manufacturer or other third parties. These tools greatly simplify the task of developing the program. Be aware that a highly optimized compiler can produce object code that runs an order of magnitude faster on the same hardware platform than that of a nonoptimized

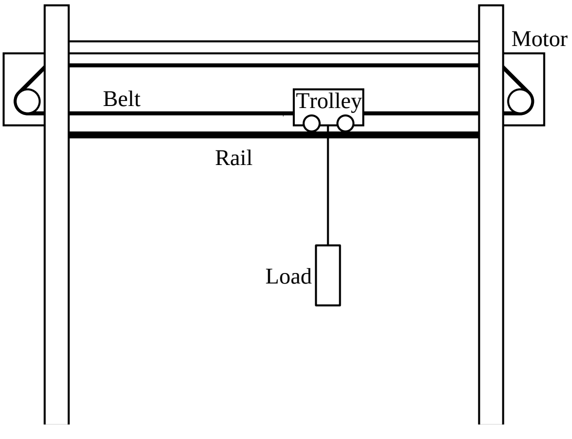


FIGURE 3.2. Schematic of overhead crane

compiler.

The third consideration is the type of peripherals required for the microprocessor. There is a class of integrated circuits, commonly known as microcontrollers or single chip microcomputers, which not only contain the basic microprocessor but also many other peripherals integrated onto the chip. Using one of these chips will reduce the chip count on the controller board. It will also reduce the possibility of hardware bugs as well as increase the reliability of the final board. These three considerations should be carefully weighed in selecting the microprocessor for the controller board.

The amount of EPROM and RAM to provide for the controller system will very much depend on the program it is executing. Checking the memory needed when a similar program is compiled on a general purpose computer will generally give a good gauge of the memory required by the target system, qualified by possibly different compiler efficiencies. Typically, controller systems do not have huge memory requirements. Otherwise there may be a need to look into using the cheaper dynamic RAM in controller systems instead of static RAM. Of course dynamic RAM is more complex to interface to the microprocessor, and is slower than static RAM.

The input/output interfaces required will very much depend on the application at hand. The interface requirement may tie in closely with the selection of the microprocessor for the system. If there is a microprocessor that meets the application requirements, and in addition, has all the peripheral interfaces needed integrated on chip, then it is only natural to select that particular processor for the application. Otherwise it may be necessary to use separate interface components, or seek a solution using two or more processors.

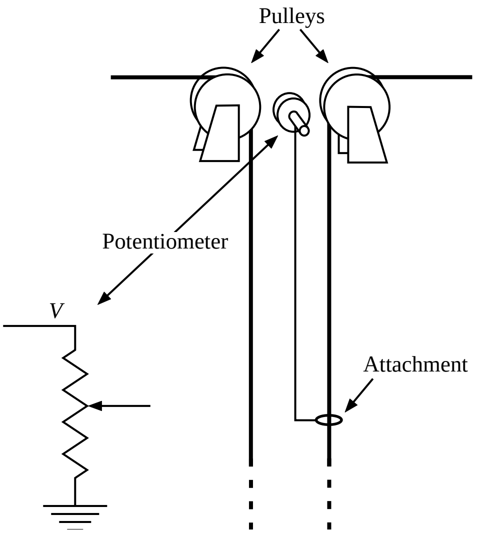


FIGURE 3.3. Measurement of swing angle

Controller for an Overhead Crane

Figure 3.2 shows the schematic of a laboratory size overhead crane. The crane is designed to pick up a load from one location and move it to another location. The bigger cousins of such cranes are used at many harbors to move containers from ships onto trucks and vice versa. The crane has a movable platform running on a pair of parallel suspended tracks. The load in turn is suspended from the platform. The height of the load from the ground is controllable by a motorized hoist which is mounted on the structural frame. The platform's position is controlled by another motor driving a geared belt attached to the platform. The DC motors have quadrature phase optical encoders attached to their shafts, which allow precise calculation of relative positions. The swing of the load is also measured using a potentiometer as depicted in Figure 3.3.

From a control point of view, there are three sensors and two actuators. The three sensors are the two optical encoders that produce two sets of quadrature phase signals and a potentiometer setup that produces a varying voltage output. The two actuators are the two DC motors with their speed and torque controlled by *pulse-width modulated* (PWM) input signals.

Based on the sensor and actuator requirements, the control system must have a target microprocessor, two up/down counters for the two sets of quadrature phase signals, an analog-to-digital converter for the output of the potentiometer and two bidirectional PWM driving channels.

In view of the requirements, it turns out to be possible with current technology

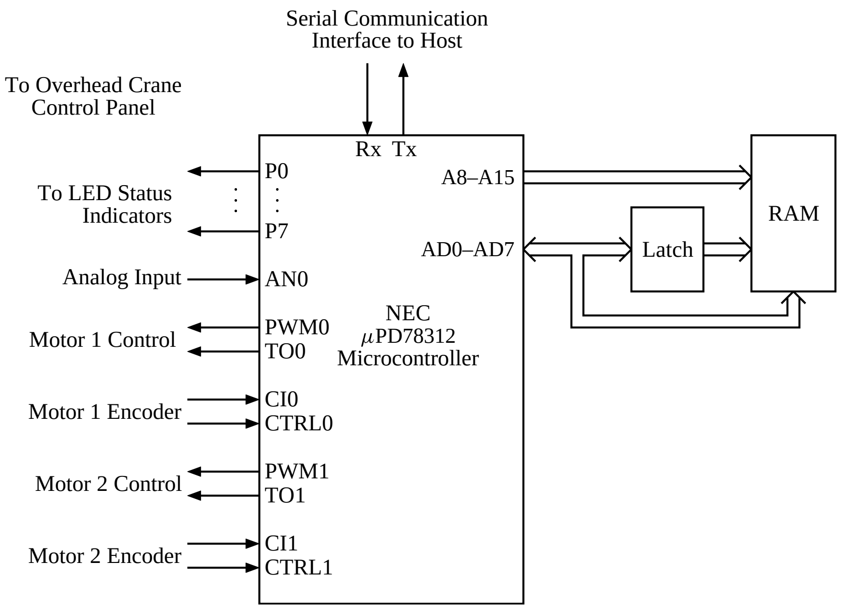


FIGURE 3.4. Design of controller for overhead crane

to select a microcontroller with all the required peripherals and a built-in serial port on-chip as the target processor for the controller board. The hardware design is shown in Figure 3.4.

Note that the number of components used in this application is minimal. The processor has a built-in 8 KB of EPROM and 256 bytes of RAM. These are sufficient for implementing simple control algorithms. However for program development purposes as well as possibly implementation of more complex algorithm, a 32 KB RAM chip is included.

Controller for a Heat Exchanger

Figure 3.5 shows the schematic diagram of a scaled down version of a heat exchanger. The control objective is to maintain the level and temperature of the tank at some preset references. The two valves can be electronically opened or closed

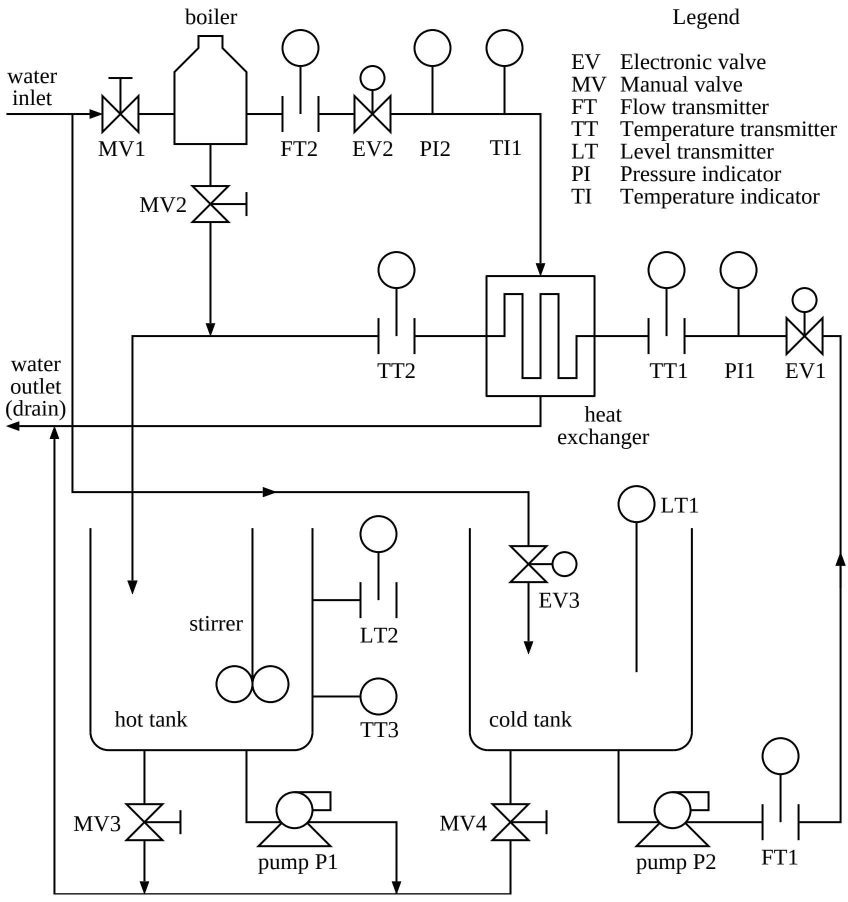


FIGURE 3.5. Schematic of heat exchanger

to control the amount of steam and water through the heat exchanger.

For this plant, there are two sensors, namely the temperature and level sensors and two electronic valves acting as the actuators. (The pressure indicators are not used in this control loop.) All the sensors and actuators are of industrial grade, and deliver and receive 4 mA to 20 mA signals, respectively. To interface to these sensors and actuators, two analog-to-digital and two digital-to-analog converters are required.

In view of the requirements, we select a microcontroller target processor with two channels consisting of analog-to-digital and digital-to-analog converters. The hardware design of the controller system for the heat exchange is shown in Figure 3.6.

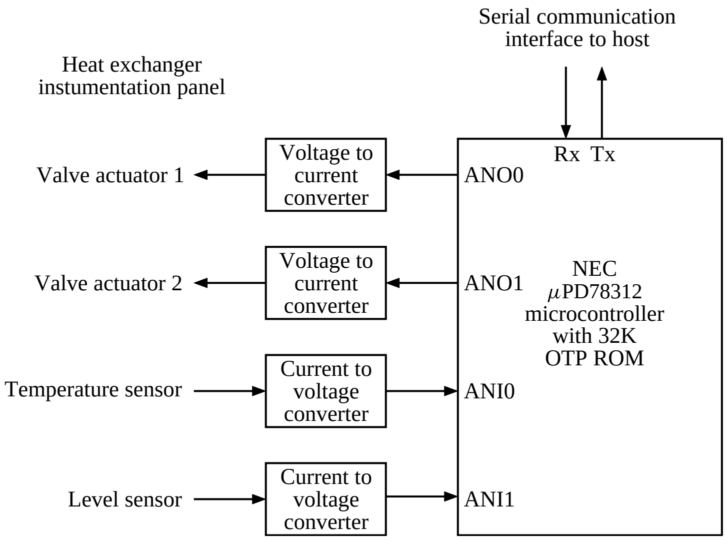

Serial communication interface to host
Heat exchanger instumentation panel
Valve actuator 1
Voltage to current converter
Valve actuator 2
Voltage to current converter
Temperature sensor
Current to voltage converter
Level sensor
Current to voltage converter
Rx Tx
ANO0
ANO1
ANI0
ANI1
NEC μPD78312 microcontroller with 32K OTP ROM

FIGURE 3.6. Design of controller for heat exchanger

Software Development Environment

For commonly used microcontrollers, processor emulators are provided by the manufacturer or some third party vendors. A processor emulator emulates all the functionalities of the specific microcontroller in real-time. It usually consists of an electronic board connected to a personal computer or workstation. The electronic board has a plug to be inserted into the microcontroller socket of the target controller hardware. This plug has the same pin-outs as the specific microcontroller. The emulator then provides a software development and debugging environment for the stand-alone system using the resources of the PC or workstation.

Processor emulators are ideal for software development. However they are either very expensive, or only made available to developers of high volume products. Low volume controller based designers often have difficulty getting such an emulator. In the absence of an emulator, what then is the procedure for development of the control software? Traditionally, it is done as follows.

The software that implements the control algorithm is coded on a workstation using a native editor of the workstation. It is then compiled using a cross-compiler into binary object code for the target stand-alone system. The EPROM of the target system is then unplugged from the stand-alone board and “erased”. This involves subjecting the EPROM to about 15 minutes of ultra-violet radiation, but recent CMOS RAM technology with a switch can simplify this process. Once the EPROM is “erased”, the binary object code can be programmed into the EPROM using an EPROM programmer. Once this is done, the EPROM can be plugged back onto the target board. Power to the board can then be applied to verify the software code. If there are bugs in the program, the whole cycle will have to be repeated.

Even for a simple program, it is not unusual for the procedure to be repeated a number of times before all the bugs are removed. If the program to be implemented is complex, as would be the case for the algorithms proposed in this book, the development using this approach could be very inefficient. Note also that the success of this procedure is on condition that the hardware is already debugged and verified to be working properly. In the event that this is not the case, the procedure could be quite frustrating for the inexperienced, and even for the experienced.

In this section, we present a procedure that will greatly simplify the development procedure. Consider the setup of Figure 3.7. The host computer is any workstation or personal computer where a cross-compiler for the target microprocessor of the stand-alone controller system exists. The host computer is linked to the controller system through either a serial link or a parallel link.

Programs for the target microprocessor can be developed and cross-compiled to an object code on the host computer. Once this is done, the program can be sent or downloaded to the target system through the communication link. This is easily accomplished using a serial port driver program available with most general purpose host computers. To complete the loop, a program residing on the target computer has to be written to accept the object code when it is downloaded through the communication link, load it into memory and execute it when the entire file is received.

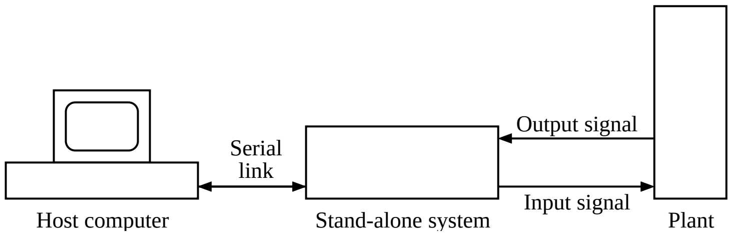


FIGURE 3.7. Setup for software development environment

The procedure is conceptually simple. The question is: How easily can this be achieved in practice? The key difficulty in the procedure described appears to be the writing of the program which resides on the target system referred to as the bootstrap loader. This program has to be developed in the traditional way as described above, that is through the programming of the EPROM. Clearly, if this program is more complex than the controller algorithm that we are going to develop, then there is no advantage in adopting this procedure. Let us examine the complexity of writing the bootstrap loader.

The format of the object code which the cross-compiler produces varies. Beside the program data in binary format, the object code also contains information to load the program data. This is the address information that tells a loader where in the memory space the program is to be loaded. The format of the object code should be given in the manual of the cross-compiler. If the object code is sent down to the target microprocessor through the communication link, then it is the

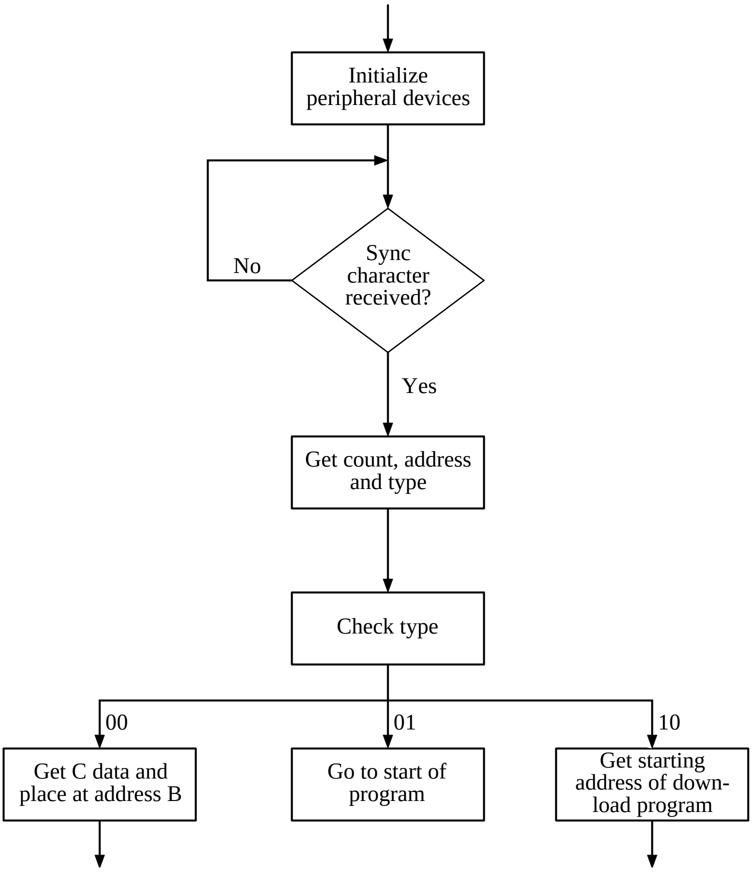

Initialize peripheral devices
No
Sync character received?
Yes
Get count, address and type
Check type
00
01
10
Get C data and place at address B
Go to start of program
Get starting address of down-load program

FIGURE 3.8. Flowchart for bootstrap loader

task of the bootstrap loader to interpret these codes; essentially to extract the address information and load the program data at the appropriate memory locations according to the address information extracted.

Currently popular microprocessors are those from Intel[*] or Motorola[†]. For the microprocessors from these two manufacturers, the cross-compilers usually compile to the Intel Hex format and the Motorola S format, respectively. A bootstrap loader is but a few lines of code.

A C-program that implements a bootstrap loader to receive a downloaded object code in the Intel Hex format can be written to implement the flowchart of Figure 3.8. In the event that the cross-compiler does not output the object code in the Intel Hex format, it is probably easier to write a converter program on the host computer to convert the foreign format to the Intel Hex format. In this case, all the programming support resources are available to the user.

With this setup, the environment of the stand-alone system is virtually extended to include that of the host computer. All the resources of the host computer such as the keyboard, monitor, disk drives, editor, compiler, etc. are now accessible to the stand-alone system and can be used in the development of its software. Practically, the stand-alone system is loaded with an *operating system*, and a *Serial Port Operating System*, (SPOS) as compared to merely the *Disk Operating System* (DOS) of a PC. Moreover, once the program is developed and verified to be working correctly, it can then be recompiled and programmed into the EPROM so that the eventual controller system can operate as a stand-alone unit with the serial link removed.

Software Debugging Environment

In the last section, a serial link and a bootstrap loader is used to virtually extend the hardware and software capability of the stand-alone system. The logical next step is to provide a debugging facility on the stand-alone system for the development of its software. Of course the necessity of providing this facility will depend on the complexity of the software to be developed. In the case where the software is relatively simple, it may not be cost effective to invest time in the provision of the debugging facility. In the other case where the software to be developed is complex, the investment will save considerable time in the subsequent debugging of the software.

There are at least three levels of debugging to be provided for effective development of a program. The first is to provide a programmer with a single stepping facility. This facility allows the programmer to execute one instruction at a time. After each instruction, the programmer is allowed to view the values of variables to ascertain the correctness of the program. This facility is normally used to debug short segments of a program which contain complex logical flows. It is usually not used to debug the entire program, with possibly thousands of lines, as it is too time consuming.

---

[*]Intel® is a registered trademark of the Intel Corporation.

[†]Motorola® is a registered trademark of the Motorola Corporation.

The second allows the programmer to set a breakpoint somewhere in the program. In this case, the program is run from the beginning until the breakpoint. After the breakpoint, control is transferred to a monitor routine which allows the programmer to view the values of the variables in the program and then engage the stepping facility. This facility is used to debug larger segments of a program which are suspected to contain logical errors.

The third level is to provide a facility for the programmer to monitor the values of variables in the program as the program is executed. On a general purpose computer, this level is equivalent to printing the variables on the monitor screen as the program is executed or to a file for later reference. This level allows the programmer a general view into the program, being especially useful when the exact error in the program cannot be isolated.

Single Stepping

Single stepping cannot be provided entirely through software means. The microprocessor must either have hardware features that support this operation, otherwise certain external hardware logic has to be included. To provide a single stepping facility, the microprocessor or external hardware logic should have the capability to generate an exception or interrupt after executing each instruction. Many processors provide such a facility and the facility is usually activated by setting a bit in the status register of the microprocessor. In the Intel 80x86 series of microprocessor, this bit is known as the trap flag whereas in the Motorola 68K series microprocessor, it is known as the trace flag.

Figure 3.9 shows the mechanism that implements the single-stepping facility in microprocessors. To invoke single stepping, the user sets the relevant bit at the point where the microprocessor is required to go into the single stepping mode. Once this bit is set, the microprocessor will on completing the current instruction generate an internal interrupt. This interrupt will cause the microprocessor to do an indirect branch to an interrupt service routine through a fixed memory location in the interrupt vector table.

The user has to write the interrupt service routine to perform the tasks desired. The task is usually to display or store the values of the microprocessor's internal register as well as certain relevant memory locations so that the user can ascertain their validity. The starting address of the service routine is then inserted at the predetermined fixed memory location in the interrupt vector table.

As part of the indirect interrupt call, the return address and the program status word are saved on the stack and the single-stepping bit in the status register is cleared. Once the single-stepping bit is cleared, the microprocessor will no longer generate an interrupt after completing each instruction. The interrupt service routine is therefore executed without interrupt after every instruction.

At the end of the interrupt service routine is an "interrupt return" instruction. When this instruction is executed, the previously saved return address and program status word is restored into the microprocessor. Control is thus returned to the main program, where the next instruction will then be executed. Of course,

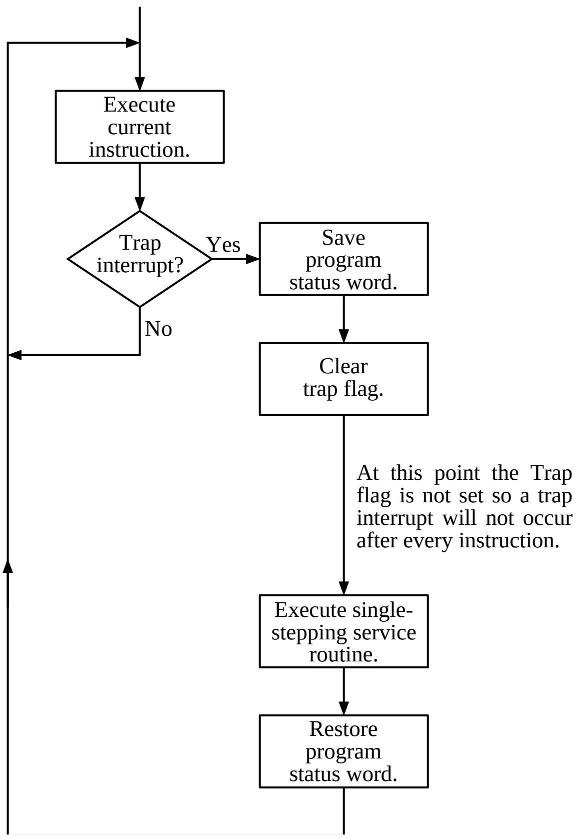

Execute current instruction.
Trap interrupt?
Yes
No
Save program status word.
Clear trap flag.
At this point the Trap flag is not set so a trap interrupt will not occur after every instruction.
Execute single-stepping service routine.
Restore program status word.

FIGURE 3.9. Mechanism of single-stepping

when the program status word is restored, the single-stepping bit which is part of the program status word is again set. The microprocessor will therefore generate an interrupt after executing the next instruction in the main program. The whole procedure is thus repeated until an instruction to clear the single-stepping bit is encountered in the main program.

From the above explanation, a single-stepping facility is achieved by writing an interrupt service routine and inserting the start address of the routine at a predetermined fixed address location. The facility can then be invoked by set a relevant bit in the program status word.

Breakpoints

Technically, breakpoints can be generated entirely using software means. The easier way is to insert calls to a breakpoint service routine at the places where breakpoints in the main routine are desired. Once the execution of the main program reaches the place where the breakpoint routine is inserted, control will be transferred to the breakpoint service routine. In this routine, users can examine the various internal values of the microprocessor. Once this is completed, control can be returned to the main program and execution of the main program can continued to the next breakpoint.

In fact breakpoints and single-stepping can be alternatively invoked in the same main program. This allows a localized as well as global debugging of the main program.

Continuous Monitoring

To monitor certain variables, calls to a subroutine which puts the variables concerned into temporary memory locations are inserted at the relevant places in the main program. The saved values of the variables can then be transmitted to the host PC via the serial link after execution of the main program is completed. Alternatively, the values of the variables can be saved into a first-in, first-out queue. The serial port can then be placed in an interrupt mode, which will transmit in real-time any saved variables in the queue to the host PC. The implementation of the software queue is depicted in Figure 3.10.

The subroutine `putchar()` is called whenever certain variables are to be saved. This subroutine saves these variable in the first-in, first-out queue implemented in RAM. The serial port is connected in the interrupt mode. Thus whenever the transmit buffer in the serial port is empty, it will generate an interrupt. An interrupt service routine will then be called to retrieve a value from the queue and write it into the serial transmit buffer for onward transmission to the the host

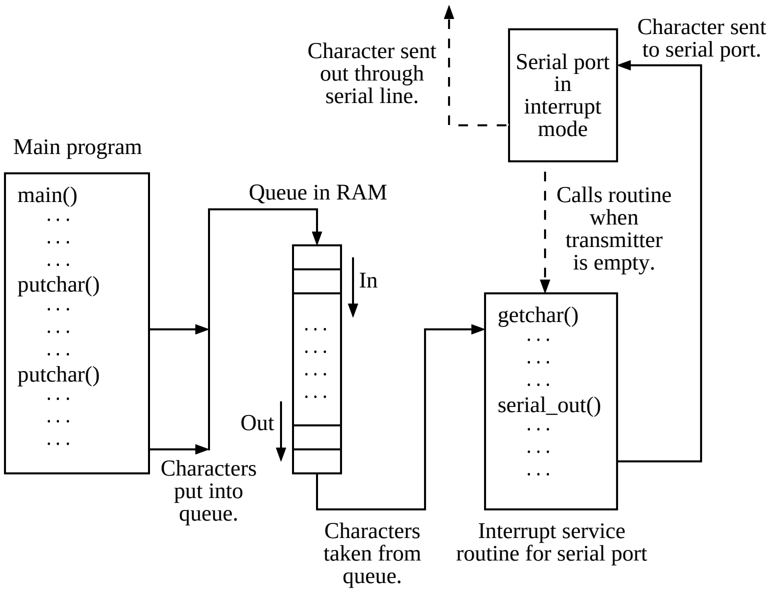

Character sent out through serial line.
Serial port in interrupt mode
Character sent to serial port.
Main program
main()
putchar()
putchar()
Queue in RAM
In
Out
Calls routine when transmitter is empty.
getchar()
serial_out()
Characters put into queue.
Characters taken from queue.
Interrupt service routine for serial port

FIGURE 3.10. Implementation of a software queue for the serial port

PC. This additional task for the microprocessor does not normally take up too much resources from the microprocessor. Of course, in real-time data processing where the main task of the microprocessor takes up all the processing time of the microprocessor, then this procedure is not feasible. However in many cases, the microprocessor is selected to have an additional 10% to 15% processing power than the requirement. This additional processing power is to allow for unforeseen additional tasks within the main task. This additional processing power if not utilized can be used to implement the debugging facilities.

A Dual-Processor Solution for Fast Processes

For relatively fast processes and relatively complex control algorithms, such as those encountered in the aerospace industry, the sampling interval and computations required tends to be beyond the limits of simple processor technology. There is motivation to move to more powerful processors. However beside requiring a powerful processor, the controller system will also require all the interfaces to the sensors and actuators of the plant to have rapid data flows.

Powerful processors optimized for number crunching such as DSPs are the processors needed in such applications. It is ideal if the DSPs also come with many peripheral functions integrated on-chip. However currently this is not the case. Firstly, the demand for such chips is not high enough for integrated circuit (IC) manufacturers to invest in the design and manufacturing. Secondly the die size for such highly integrated chips would be large, and consequently the production yields of the chips low and therefore the costs are higher.

To provide a hardware platform with a powerful number crunching capability as well as numerous interface functions, it seems appropriate to exploit the strength of both the DSPs and microcontrollers in what we call a dual chip design. We present one such design which we call the Fast Universal Controller (FUC). The high computation speed of the FUC is derived from the DSP and it can go up to tens or hundreds of MFLOPS (millions of floating point operations per seconds), depending on the particular DSP selected. It is universal in that it contains numerous input/output interfaces to satisfy most controller needs.

The design of the FUC is shown in Figure 3.11. It can either function as a stand-alone unit or maintain a serial communication link to a host computer. Internally, the FUC consists of two distinct processing modules which perform different tasks. The first module (DSP module) contains a DSP and associated resident RAM. This module is assigned to perform number crunching in the overall strategy of the FUC. Typically, it is programmed to execute all tasks of the FUC unit except servicing of external devices. The DSP does not maintain any hardware link to external devices except to the second module through a DMA (direct memory access) channel.

The heart of the second module (microcontroller module) is a microcontroller which provides interaction with all external processes or devices through its built-

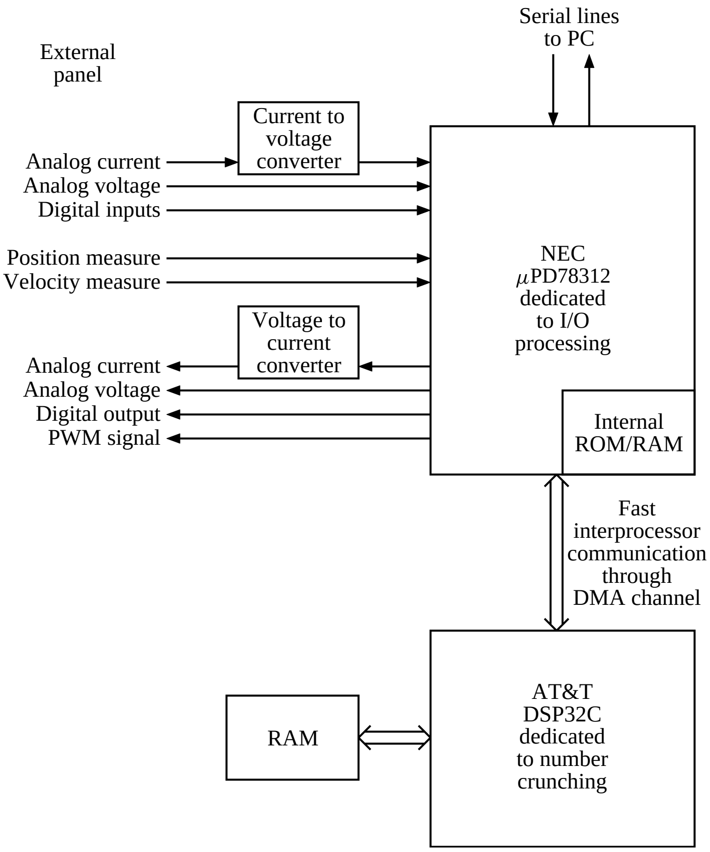

Serial lines to PC
External panel
Current to voltage converter
Analog current
Analog voltage
Digital inputs
Position measure
Velocity measure
Voltage to current converter
Analog current
Analog voltage
Digital output
PWM signal
NEC μPD78312 dedicated to I/O processing
Internal ROM/RAM
Fast interprocessor communication through DMA channel
RAM
AT&T DSP32C dedicated to number crunching

FIGURE 3.11. Design of a fast universal controller

in I/O interfaces as well as other specialized chips such as analog-to-digital converters (ADC), digital-to-analog converter (DAC) and programmable digital I/O chip. Any information to be passed to the DSP section is done through the DMA channel.

DSP Module

This module consists of the DSP, in our application an AT&T[‡] DSP32C, 128 K 32-bit words of RAM and the necessary logic circuits to interface the RAM to the DSP. There is no nonvolatile memories or interfaces to external devices. The only link to the external world is a DMA channel to the NEC[§] microcontroller. The DMA controller is a built-in feature of the DSP. This DMA controller is accessible

[‡]AT&T® is a registered trademark of the AT&T Corporation—formerly the American Telephone and Telegraph Company.

[§]NEC® is a registered trademark of the NEC Corporation—formerly the Nippon Electric Company, Limited.

by the microcontroller through a 16-bit control/data bus. Through this control/data bus, the microcontroller is able to control the read/write of any memory location within the DSP memory address space. This bus also allows the microcontroller to do a soft reset of the DSP.

The program to be executed on the DSP is loaded onto the DSP's memory by the microcontroller through the DMA channel. Once the entire program is loaded into the DSP's memory, the microcontroller will initiate a software reset of the DSP. This causes the DSP to branch to the first executable instruction of the downloaded program and commences execution of the application.

In real-time execution of the application program, the DSP will monitor sensor readings and send control signals to actuators. Since there is no hardware link between the DSP and the external devices, interaction is done using a software approach as follows. Within the memories of the DSP, certain fixed memory locations are reserved as mailboxes for each of the hardware ports. That is, there is a memory location that serves as a mailbox for each of the peripherals; analog-to-digital converters (ADC), digital-to-analog converters (DAC), pulse width modulator (PWM) ports, etc. For input devices such as the ADC, the microcontroller is responsible for obtaining the readings from the hardware device and then transmitting this information to its corresponding mailbox (in the DSP memory address space) using the DMA channel. The DSP then picks up the data from the mailbox (memory location). Similarly, for any data that the DSP needs to write to an output device (such as a DAC), it places the data into the corresponding mailbox for the output device. The microcontroller next retrieves the data from the mailbox using the DMA channel and then appropriately transmits the data to the actual hardware device. In this way, the DSP interfaces to external devices are implemented by simply accessing memory locations. Furthermore the hardware devices have "perceived" hardware characteristics for the DSP that are exactly the same as those of RAM.

Microcontroller Module

This module consists of the microcontroller and all the peripheral chips that are needed to service all the various hardware processes. The microcontroller is not involved in the execution of the control algorithm. However it has to perform a number of supporting tasks.

The first task is to ensure that application programs for both the DSP and itself are properly loaded. In the stand-alone mode of the FUC unit, the program for the DSP is burned into the EPROM of the module. Once the unit is powered up, a bootstrap loader is executed to transmit the DSP program through the DMA channel to the proper location within the DSP memories, reset the DSP and branch to the beginning of the NEC microcontroller program. In the mode where the programs are to be downloaded from the host computer, the bootstrap monitors the serial link for the downloaded program. Once the program is downloaded from the host computer, the bootstrap loader receives the programs and loads them into the appropriate memory locations of the DSP or microcontroller accordingly.

The second task is to maintain synchronous timing for the entire FUC unit. The built-in timer of the microcontroller is tasked to maintain the precise time required to implement sampling intervals.

The third task is to service all the peripheral devices. For input devices, the microcontroller will have to ensure that proper timing and procedures corresponding to each device be adhered to so that a reliable reading is obtained. The obtained reading is then transmitted to the device's mailbox within the DSP module at the appropriate instant. For output devices, the microcontroller will obtain the data value from the appropriate mailbox and then drive the corresponding hardware output device. All device characteristics are to be taken care of at the microcontroller's end.

Finally should there be a need to send any data to the host computer for data logging, the microcontroller will appropriately condition the data and then send them via the serial line to the host computer.

As as concluding remark to our discussion of a dual processor solution, it is interesting to note the modular approach adopted in the design. This approach allows an easy development of the control algorithms. There is no need for the control engineer to worry about servicing the peripheral devices. This is an advantage since having to service peripheral devices in real-time always makes programming complicated because different devices have different operating requirements. As an example of this increased complication, an ADC requires some finite time interval from the command to start converting to delivery of the converted data. This finite time interval may take many processor working cycles. In this case, it may not be feasible for the processor to just wait for the ADC to complete the conversion, because of other commitments. It is then necessary to set up an interrupt routine so that when the ADC is ready to deliver the data, the processor is interrupted to read in the data. In our case the control engineer can assume that all these operating requirements are taken care of by the microcontroller module and the data will be available in the mailbox for the peripheral when and as it is required. There is no need for background and foreground differentiation of various parts of the program, which then allows straightforward programming of the control algorithm. The program on the microcontroller will be servicing all the peripherals and takes care of all the operating requirements. This program may be a little more complicated but since there is one task less, the overall design is somewhat simplified.

A Personal Computer-based Solution

In this section we describe the design of a universal input/output (UIO) board that plugs into the I/O bus of a personal computer (PC). The board is similar to the microcontroller module of the FUC except that it does not have a DMA link to the DSP module. Instead it has a parallel interface to the I/O bus of the PC mother board. This board plugs directly into the expansion slot of the PC mother board and the peripherals on board are then accessible by the microprocessor residing on the PC mother board. The design of the UIO is shown in Figure 3.12.

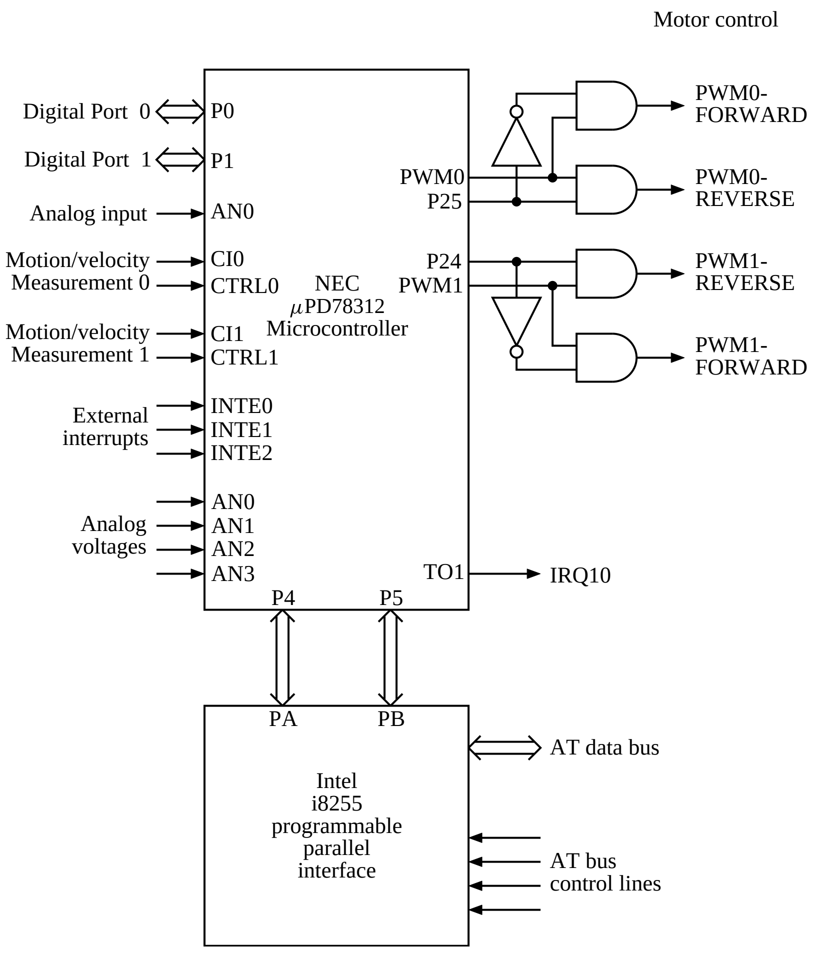


FIGURE 3.12. Design of universal input/output card

The program for the control algorithm is now developed and executed on the PC. All the resources of the PC are available to the programmer for development of the software. There is no need to write a bootstrap loader. There is also the advantage of having a cheap and well written optimized compiler for the development. Although we call this solution a personal computer based solution, there is really no need for the PC once the program for the control algorithm is developed. We can transform it into a stand-alone system. We only need to purchase a PC mother board at low cost. The object code of the program can be programmed into a EPROM and this replaces the EPROM that comes with the PC mother board. The UIO will also have to be plugged into the PC mother board. Once done, power can be applied and the three components consisting of the PC mother board, UIO and the control software (on EPROM) are transformed into a stand-alone controller system. What is further required is a properly designed casing.

As mentioned before, this approach is ideal for low volume production of controller systems. Because of the high volume turnover for PC mother boards, the price is relatively low. Also because of the high integration used in the design of the mother boards, they tend to be reliable. It also has the advantage of allowing the control engineer to develop the software on the PC. A limitation for computationally intensive controllers is the limited bandwidth of a PC bus. However, there are PC/DSP arrangements which are available now which only use the PC for development and not for controller implementation, and in this way overcome the bandwidth limitations of the PC.

Main Points of Section

In this section, we have described the hardware platform to implement control algorithms. A methodology to simplify the development of a microprocessor based controller system is presented. A dual-processor design approach for plants which require more computational resources and a personal computer based approach are also described.

9.4 Software Platform

The software for the control algorithm can be developed on a number of platforms. The platform will depend on the complexity of the algorithm as well as the degree of optimality of the code desired. To obtain an object module that is short as well as executes in the shortest possible time, it is necessary in some cases to program in assembly language and do hand optimization of the code. This however is very time consuming and difficult unless the problem is relatively simple. Of course for the development of products that will be mass produced, this may be a worthwhile investment. However for most situations this is to be avoided.

The other alternative is to code in the C language, which is frequently the software platform of choice. The object code produced is not as optimized as that produced by a hand optimized assembly language source, but it is a good compromise between development efficiency and code optimality.

Control via Coding in MATLAB[¶]

With more powerful computers and more complex problems to be solved, the associated software is correspondingly more complex. Just as the trend in hardware is towards higher integration, so there is a trend for software to move in this direction. There is now a greater dependency on prewritten software modules. In the

[¶]MATLAB® is a registered trademark of the MathWorks, Inc.

past, it would takes an experienced programmer many days or weeks to write a graphical based user interface, but today this can be accomplished in a few minutes by any programmer using one of the many application packages. Software libraries are not new, however the functions offered by today's software library are no longer basic. They now come in more integrated forms and each function accomplishes a more complex task. The question that arises is: How does this trend affect the coding of control algorithms?

MATLAB was originally a public domain software package developed to provide easy access to matrix and linear algebra manipulation tools developed by the *LINPACK* and *EISPACK* projects. Since then it has been further developed by the MathWorks into a commercial version. Rewritten public domain versions are also available and users are advised to check the *Usenet* archive sites for announcement of the latest release. MATLAB comes with a programming language command set as well as many built-in functions that are commonly used in design and simulation of control systems. The command language allows users to write their own functions or script files, referred to as m-files. These m-files which are in plain text format can be called from within the MATLAB environment to perform the functions they are written to perform. The m-file functions have been grouped together to form toolboxes for application to tasks such as system identification, signal processing, optimization, and control law design. There are many such toolboxes in both the commercial and public domain.

The thousands of m-file functions implement all kinds of control system design and simulation functions. In fact many algorithms are already available as m-files. With the aid of these m-files, often there is no necessity to code algorithms from scratch. The main program to perform the task defined is merely a series of calls to each of these m-file functions. For illustration, rather than for application by the reader, the design and simulation of an LQG controller is given in Figure 4.1. Note that `dlqr` and `dlqe` are m-file functions that compute the controller and estimator gains of the LQG controller respectively.

The question we now ask is: How can the above design and simulation program be used to control a real plant? Consider the personal computer based solution described in the last section. We mentioned that the program for the control algorithm is to reside on the PC and the input and output linkage to the plant is done through the universal input/output (UIO) card. In order for us to use the above MATLAB program to control the plant, we need to do three things. First, we need to install MATLAB on the PC. Second, we need some routines which allow us to access the UIO peripherals from within the MATLAB environment. Third, we need to modify the above program so that instead of simulating the plant dynamics to get the plant output, we can directly call a routine to read output measurement data from the actual plant. Similarly the controller output will have to be sent to the plant through the UIO.

Two C-language programs ADC.c and DAC.c can be written. ADC.c, when called, accesses the analog-to-digital converter of the UIO and returns the digital values of the analog signal. DAC.c on the other hand accepts the parameter passed to it by the MATLAB program and sends it to the digital-to-analog converter of

```
% State space definition of nominal plant
A = [0.7  0 ; 0  0.8];
B = [0.5 ; 0.8];
C = [1  1.5];
D = [0];

% Definition of design parameter for LQG controller
Q_control = C'*C;
R_control = 1;
Q_estimator = 1;
R_estimator = 1;

% Design of LQG controller
K_control = dlqr(A,B,Q_control,R_control);
K_estimator = A*dlqe(A,B,C,Q_estimator,R_estimator);

% Initialization of variables
xk = [0;0];
uk = 0;
xhat = [0;0];

while ( 1 ),
    yk = C*xk + D*uk + rand(1,1);
    xk = A*xk + B*uk + B*rand(1,1);
    uk = K_control*xhat;
    xhat= A*xhat + B*uk + K_estimator*(yk - C*xhat);
end
```

FIGURE 4.1. Program to design and simulate LQG control

the UIO for conversion. The two programs can be linked with MATLAB which then allows them to be called from within MATLAB.

With the routines just described, the simulation program in Figure 4.1, modified as illustrated in Figure 4.2 (again not necessarily for use by the reader), becomes a real-time control program.

There are three places where modification is necessary. First, the two lines that simulate the plant is removed and replaced by `yk = ADC(1)`. This is really a call to ADC.c to obtain in digital form the analog output value of the plant. Second, an additional line `DAC(uk)` is added. As the reader would have guessed, this is to send the control `uk` to the actuator of the plant via the UIO. The third modification is the introduction of the statements `start\_time = clock` which assigns the current time to the variable `start\_time` and `etime(clock,start\_time) <= sam-ple\_interval` which gives the difference between the current time and the time recorded in the variable `start\_time`. These statements ensure that the `while` loop is executed once every `sample\_interval` seconds.

Implementing the controller algorithms using m-files has another advantage.

```
% State space definition of nominal plant
A = [0.7  0 ; 0  0.8];
B = [0.5 ; 0.8];
C = [1  1.5];
D = [0];

% Definition of design parameter for LQG controller
Q_control = C'*C;
R_control = 1;
Q_estimator = 1;
R_estimator = 1;

% Design of LQG controller
K_control = dlqr(A,B,Q_control,R_control);
K_estimator = A*dlqe(A,B,C,Q_estimator,R_estimator);

% Initialization of variables
xhat = [0;0];

% Initialization for real-time control
sample_interval = 1;
start_time = clock;

while ( etime(clock,start_time) >= sample_interval ),
    start_time = clock;
    DAC(uk);
    yk = ADC(1);

    uk = K_control*xhat;
    xhat= A*xhat + B*uk + K_estimator*(yk - C*xhat);
end
```

FIGURE 4.2. Program to implement real-time LQG control

For most purposes, before an algorithm is implemented, it is designed and simulated to check that it is performing to expectation. This part is usually performed using a high level design and simulation package such as MATLAB. Often only after the simulation shows promising results does one decide to apply the algorithm to the real plant. The practice is then to recode the algorithm in some programming language such as C for implementation on the real plant. This step is usually done by another group of system programmers. However program bugs and communication between the design group and the implementation group may lead to problems and delays in the project. Our proposed approach, working with m-files makes implementation straightforward for the control designers. There are only three places where modifications are made. If the real-time control does not match the simulated results, the focus can then be on the control issues rather than the implementation issues.

A MATLAB M-file Compiler

The approach, working with m-files described in the last section, has at least four drawbacks. First, MATLAB is available only on general purpose computer systems. The approach is not possible on a computer platform not supported by MATLAB. As an example, the FUC described in the previous section is not supported by MATLAB and therefore the approach is not applicable. Second, the approach requires MATLAB to be running in order to execute the m-file program that implements the control algorithm. MATLAB is a huge program and requires considerable computer resources to run. This is wasteful since the m-file program may not require all the MATLAB resources. Third, the approach allows the m-file program to be easily altered. This is an advantage during development as it allows the designer to fine tune the control algorithm quickly. However once the system is in operation and is maintained by operators who are not familiar with the design of the control algorithms, easy alteration of the m-file is strongly discouraged. Fourth, MATLAB is an interpreter and has high execution overhead. To enhance execution speed, the obvious step is to generate compiled object code for the control algorithms using a MATLAB-to-C converter such as is now commercially available. Such a converter converts any m-file into its equivalence ANSI-compliant C source. The equivalent C source can then be compiled using a C cross compiler to an object code for any target processor. The object code can then be programmed into an EPROM to give a turnkey controller system. With the MATLAB-to-C converter, the four drawbacks described above are overcome.

Main Points of Section

In this section, we describe various ways to implement the control algorithms in software. The traditional way is to code in assembly language or C. We suggested an alternative way, to program in m-files format. This allows the use of high level m-file functions available in MATLAB toolboxes to be used in the development of the control algorithms. To facilitate the development of stand-alone systems, the role of a MATLAB-to-C converter that converts m-files into C sources is described. The C source can then be compiled using a C cross compiler into object code for any hardware platform.

9.5 Other Issues

There are a few other issues which the readers will probably encounter in the implementation of a control system. We will briefly highlight them here.

Implementation of Direct Feedthrough

Strictly speaking, a direct feedthrough is not realizable in the discrete-time implementation of controllers on a computer. This is because the processor will have to take some finite time to process any data that is read in through the input interface before it can be written back to the output interface. Thus it appears that there should be at least one sample delay in the controller. In other words, the controller must be strictly causal. However in the event that the computational time of the controller is small in comparison to the sampling period, a direct feedthrough can be approximated. In this case the control signal is sent to the actuator as soon as it become available. Of course, if the hardware constraints preclude a direct feedthrough, or an approximation to this, then the controller design should yield a strictly causal controller.

Integer Representation

With floating point microprocessors, users do not have to be concerned with overflow or underflow of arithmetic operations. However, in general, it is also true that floating point processors are significantly more expensive than integer based processors. In fact in many designs, preliminary implementation is made on floating point based processors. Once the algorithm is shown to be working, the design is then converted to run on an integer based processor for mass production.

The process to convert a floating point based program to an integer based program can be rather tedious. In essence, one has to examine all intermediate results of the program to determine the range of values they can take. If the range is beyond that offered by an n-bit integer representation, then either the number of bits used to represent the number has to be increased or that value has to be scaled back accordingly. Often this is achieved by shifting the binary point to the right by m-bits or equivalently by dividing by a 2^m operation. Correspondingly, if the range is too small, then we may not be getting sufficient resolution and there is a need to expand the resolution by shifting the binary point m-bits to the left.

There is also often a need to change the order in which certain computations are performed to avoid overflow or underflow. An example is to compute the equation

$$z = \sum_{i=1}^{n} a_i - \sum_{i=1}^{n} b_i,$$

where a_i, b_i are positive numbers as

$$z = \sum_{i=1}^{n} (a_i - b_i).$$

Finite Word Length

In the development of the algorithms in the book, infinite precision representation of numbers is assumed. This assumption is not valid in real-time implementation in digital computers. The numbers used in the calculations have finite word length. We will not do any analysis on the effects of finite word length implementations of the algorithms in the book. The reader is referred to Williamson (1991) and Gevers and Li (1993) and also to Middleton and Goodwin (1990) for in-depth treatments of the various issues. However we stress that there are different sources of truncations which can adversely affect the performance of the algorithms.

Main Points of Section

In this section, three issues are flagged. The first is the implementation of direct feedthrough in discrete-time systems. The second is the effect of integer representation in control algorithm implementation. The third is the effect of finite word length in microcomputer systems.

9.6 Notes and References

This chapter sets the stage for real-time implementation of the algorithms presented in the book. The chapter begins with an introduction to discretizing of continuous-time plants for computer implementation. It then moves on to hardware/software platforms. Most of these techniques are known in industries or in laboratories, or presented in restricted publications. The material presented is based on our experience in implementing these systems both in universities as well as in industry.

CHAPTER **10**

Laboratory Case Studies

10.1 Introduction

The aim in this chapter is to present some student laboratory case studies of the application of high performance control theory and its real time implementation. Also, some simulation feasibility studies are included. The case studies are of necessity limited and perhaps contrived to some degree since they do not arise from fully funded engineering research and development programs for real world applications. Each study is presented somewhat qualitatively in order to illustrate aspects of engineering design rather than to allow for complete reproducibility of the results or to represent a triumph of the approach. The way is opened for the reader to achieve any triumphs.

10.2 Control of Hard-disk Drives

Hard-disk drives are an important data-storage medium for computers and data-processing systems. In the hard-disk drive, rotating disks coated with a thin magnetic layer or recording medium are written with data in concentric circles or tracks. Data is read or written with a read/write head which consists of a small horseshoe shaped electromagnet. This read/write head is usually driven by a servomechanism system. Within the servomechanism, two different controllers are used. The task of the first controller is track seeking, that is to move the read/write head from track to track. Usually an optimal controller that minimizes the time taken to do this is used. The task of the second controller is track following, that is to maintain the head above a particular track while data is being read or written. This is a regulation problem. Currently a combination of classical control techniques, such as lead-lag compensators, PI compensators and notch filters are used in the track following algorithm.

The objective in hard-disk drive development is towards smaller drives with high data storage capacity and faster seek and read/write times. Track to track seek time is governed by the maximum deliverable power of the motor driving the read/write head and the mass of the read/write head. The rate of read/write operations is governed by the speed of the spindle motor that rotates the magnetic disks of the drive.

There are two ways to achieve a smaller disk drive with higher capacity. The first is to reduce the size of the footprint for each bit of data. The second is to reduce the width of the tracks where the data is stored. The first has to do with magnetic storage technology, the second has to do with the quality of control of the read/write head above the tracks while data is read or written. At the time of writing, the width of each track in a disk drive is still large compared to the footprint of each piece of data. It appears that high rewards can be achieved by improving the control aspects of the drive.

There are two primary sources of disturbance in the hard disk servo system. The first is a *repeatable run-out* (RRO). As the name implies, RRO is a periodic disturbance that stays locked to the disk rotation (both frequency and phase). This disturbance is due to imperfect or eccentric tracks. See Hara, Yamamoto, Omata and Nakano (1988), Chew and Tomizuka (1990) for work in this one aspect. The second is a *nonrepeatable run-out* (NRRO). NRRO is the cumulative result of disk drive vibrations, electrical noise in the electronic circuits and the measurement channels. Vibrations come from many sources. These sources include the spindle motor, spindle bearings, air movement between the head and the disk, the actuator and also force disturbances such as closing of covers. Unlike RRO, NRRO disturbances are not periodic or predictable. Hence, they are more difficult to reject than RRO disturbances. In this section we will present designs to reduce this NRRO form of disturbances.

System Model

Two hard disk drives are used in this case study. The first drive (Drive 1) is a commercially available 5.25 inch Winchester 2.4 GB drive with a dedicated servo system providing information on the relative position for the servo read/write head. The drive has a digital signal processor (DSP) to compute the necessary control. The spindle motor is rotating at 5 400 rpm and control is done at an interval of approximately 42 μs. The existing track following regulator for the servo system is a combination of PI compensator, notch filter and lead-lag compensator. A block diagram of the servo system and its own internal controller is shown in Figure 2.1.

The position error signal 'pes' is derived from the servo head representing the deviation of the read/write head from the required track center. Here 'return' is the output from the existing controller and 'err_out' is the input signal injection into the servo system. This servo system contains an input point, 'stimin', which is used for injecting test signals into the servo system for obtaining frequency response. Under normal operation of the system, this signal is set to zero, hence 'return' equals 'err_out'.

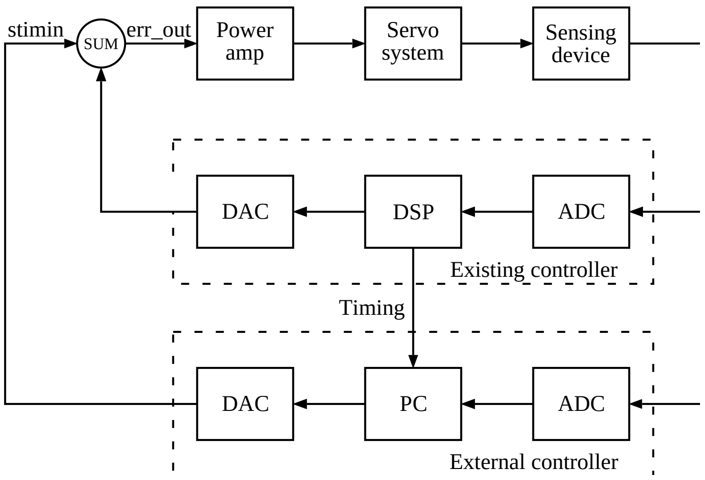


FIGURE 2.1. Block diagram of servo system

There are two possible ways to access the servo system. The first is to detach the existing controller and replace it with the new control algorithm. This would involve recoding all the functions, including hardware initialization routines currently performed by the DSP. This is not possible without a detailed description of the hardware and the addresses and functionalities of all peripherals. The other way, which is an easy way out is to treat the servo system together with its internal controller as the plant. An external controller is then designed to control this ‘plant’. However with this approach, the external controller may not be able to “see” the entire spectrum of the servo system. This is because the internal controller may have filtered out parts of the spectrum. Nevertheless this is the approach adopted in this case study. We will in the rest of the section consider the internal controller as part of the plant, termed here servo system.

Let us model the servo system as

$$\square(q^{-1})y_k = \square(q^{-1})u_k + w_k \tag{2.1}$$

where y_k = ‘pes’, u_k = ‘stimin’, $\square(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_n q^{-n}$ and $\square(q^{-1}) = b_1 q^{-1} + \cdots + b_n q^{-n}$. This can be written as a linear-in-the-parameter model as follows.

$$y_k = \phi_k' \theta + w_k \tag{2.2}$$

where $\phi = [\, -y_{k-1} \;\cdots\; -y_{k-n} \,]'$ and $\theta = [\, a_1 \;\cdots\; a_n \; b_1 \;\cdots\; b_n \,]'$.

Excitation signals applied to the inputs and the outputs are logged. A least squares algorithm is then used to estimate the parameters of the model (2.1) of the servo system. The magnitude of the frequency response of an estimated 18th, 19th and 20th order model is shown in Figure 2.2, which can be compared to a plot using a spectrum analyzer as shown in Figure 2.3. The two figures show close correlation.

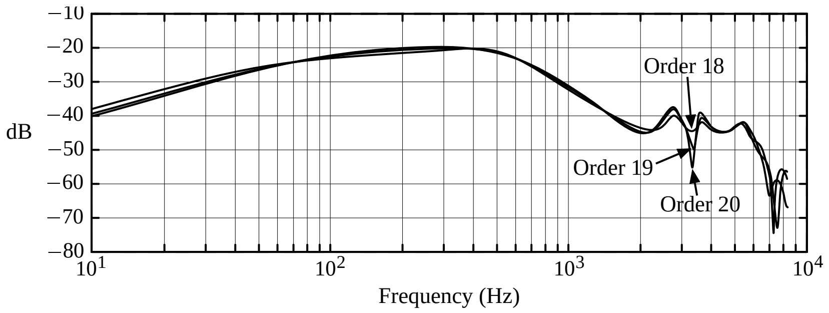


FIGURE 2.2. Magnitude response of three system models

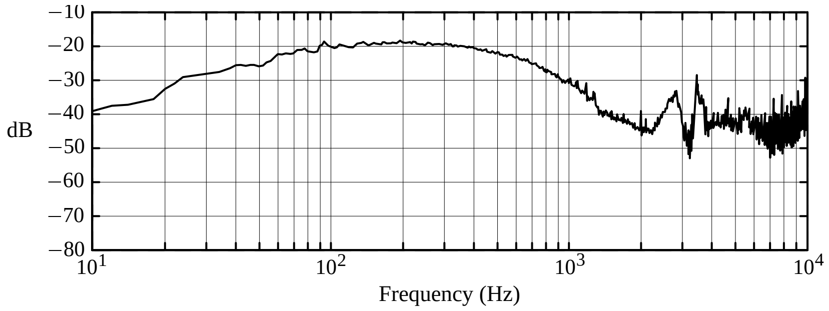


FIGURE 2.3. Measured magnitude response of the system

The second hard disk drive (Drive 2) is an IBM* Winchester 3.5 inch disk. It has a rotating speed of 4 316 rpm and control is performed at the rate of 6.9 kHz. The magnitude of the frequency response of an estimated third order model for this drive is shown in Figure 2.4.

An Optimal Disk Controller for Drive 1

A second order model is used in the design of the controller for the servo system of Drive 1. This model is obtained by approximating the high order models obtained in the previous section and is given as follows.

$$y_k - 1.83y_{k-1} + 0.86y_{k-2} = 0.001u_{k-1} - 0.007u_{k-2} + w_k. \qquad (2.3)$$

Qualitatively, the objective of the controller is to ensure that the deviation of the read/write head position from the center of each track ‘pes’, is small at all instants of time without saturating the actuator. This will then allow the track width to be set to twice the worst deviation (or slightly larger) of ‘pes’. It is clear such a qualitative specification translates quantitatively into an optimal controller that minimizes the infinity norm of ‘pes’.

*IBM® is a registered trademark of the International Business Machines Corporation.

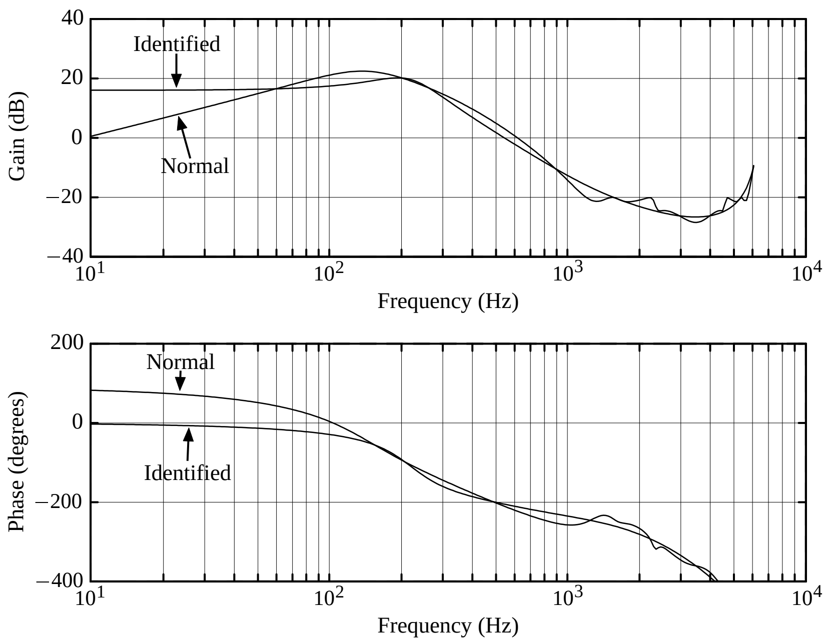


FIGURE 2.4. Drive 2 measured and model response

Let us select an error signal as $e_k = [\, y_k \;\; \lambda u_k \,]'$ where λ is a constant and set up a performance index as

$$J = \|e\|_\infty = \max(\|y_k\|_\infty \,, \|\lambda u_k\|_\infty). \tag{2.4}$$

Using the Q parameterization approach of Chapter 2, we have a closed-loop system given in terms of Q as F_Q. Using the notation of Chapter 4, we have the following optimization task.

$$\min_{Q \in RH_\infty} \left\| F_Q w \right\|_\infty . \tag{2.5}$$

Now the solution to this optimization task depends on the characteristic of the disturbance w into the system. In our case, w is certainly bounded. We could further assume w to be 2-norm bounded or infinity-norm bounded. Either assumption as shown in Chapter 4 leads to different optimization tasks.

$$\min_{Q \in RH_\infty} \left\| F_Q \right\|_2 \qquad \text{if } \|w\| \in \ell_2, \qquad \|w\|_2 \leq 1, \tag{2.6}$$

$$\min_{Q \in RH_\infty} \left\| F_Q \right\|_1 \qquad \text{if } \|w\| \in \ell_\infty, \qquad \|w\|_\infty \leq 1. \tag{2.7}$$

In this application, as mentioned, the targeted disturbances to be rejected by the controller are the NRRO and they can be attributed to disk drive vibrations and electrical noise in the electronic circuits and the measurement channels. Therefore either the 2-norm or infinity-norm bounds can be considered to be good approximations. We thus present the results of controllers designed for both optimization tasks (2.6) and (2.7).

Table 2.1 shows the amplitude and energy of ‘pes’ when a H_2 and a ℓ_1 optimal controller are used to control the system. This is compared against the case where no external controller is used. Each run consists of 200 000 samples. In general the ℓ_1 optimal controller reduces the maximum magnitude of ‘pes’ compared to when no external controller is used. The H_2 optimal controller on the other hand does little to reduce the maximum magnitude of ‘pes’. We have also computed the power of ‘pes’ in the three cases. As expected, for this criterion the H_2 controller turns in the best performance, reducing the power of ‘pes’ by about 20% compared to when no external controller is used. The power of ‘pes’ is also reduced by 10–15% when an ℓ_1 optimal controller is used.

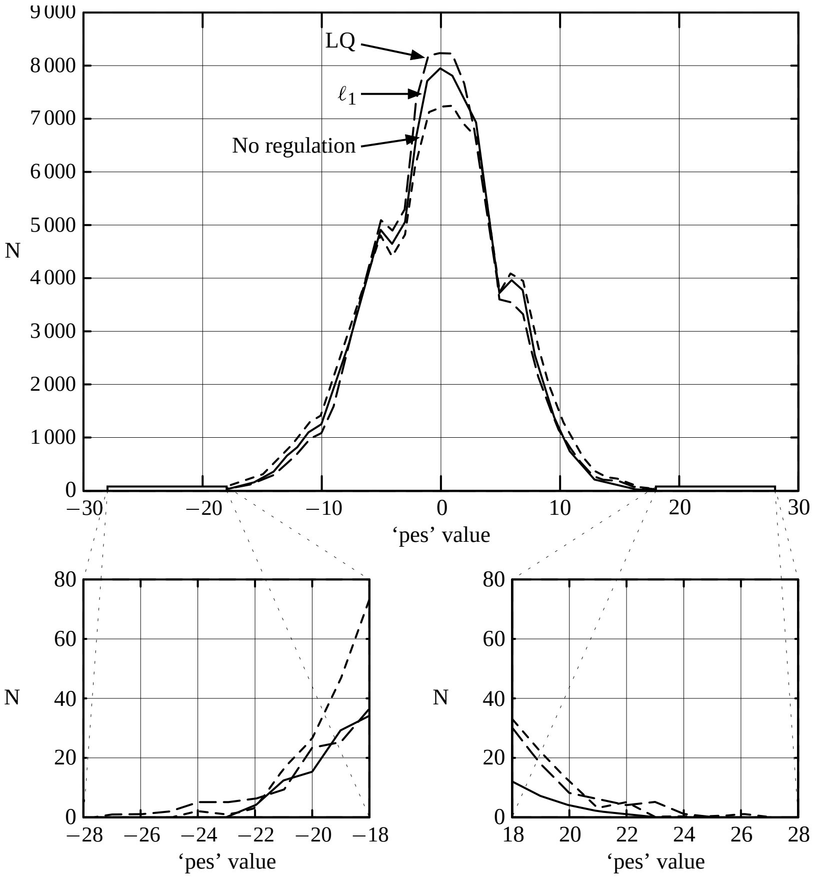


FIGURE 2.5. Histogram of ‘pes’ for a typical run

		1st Run	2nd Run	3rd Run	4th Run
Amplitude (Min/Max)	ℓ_1	−22/22	−25/23	−23/24	−22/23
	H_2	−27/26	−27/26	−25/26	−26/25
	NR	−24/26	−27/26	−25/26	−26/25
Energy	ℓ_1	29.2	33.1	32.4	33.2
	H_2	28.0	30.7	29.3	31.0
	NR	33.9	39.1	35.1	36.3

TABLE 2.1. Comparison of performance of ℓ_1 and H_2 controller

To understand the results better, a histogram of a typical run is shown in Figure 2.5. The axis shows the magnitude of ‘pes’ while the y-axis represents the number of occurrences N. Attention is drawn to the two extreme ends of the histogram shown enlarged. Here, we observe that the ℓ_1 optimal controller not only reduces the maximum amplitude of ‘pes’, but also reduces the number of occurrences at higher values of ‘pes’.

An Adaptive Controller for Drive 2

This part of the case study is extracted from Li (1995) and Horowitz and Li (1995). The adaptive controller used to control Drive 2 is depicted in Figure 2.6.

The adaptive scheme is a specialization of the adaptive techniques presented in Chapter 5 and Chapter 6. Assuming both the nominal plant G_0 and the nominal controller K_0 to be the zero operators, then the stable coprime factorizations of the nominal plant and controller are trivially given by

$$N = 0, \qquad M = 1, \qquad U = 0, \qquad V = 1.$$

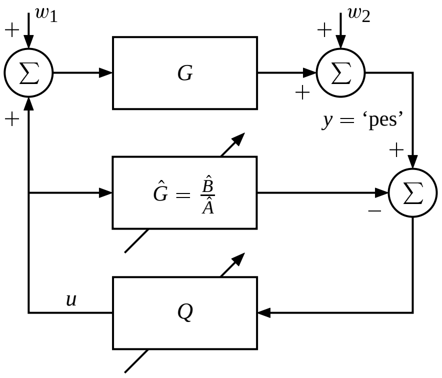


FIGURE 2.6. Adaptive controller for Drive 2

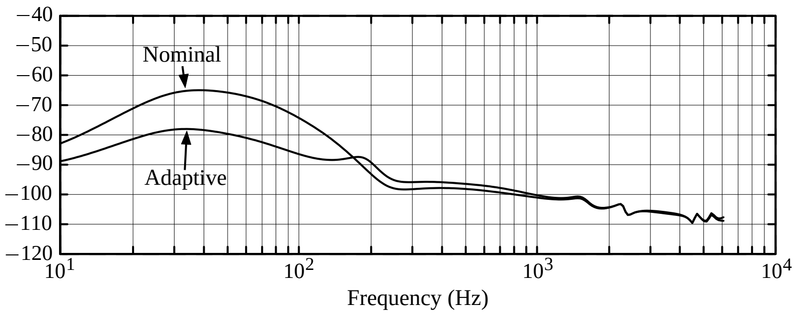


FIGURE 2.7. Power spectrum density of the ‘pes’—nominal and adaptive

With these factorizations, the class of all plants parameterized by S (which is here the servo system G), is then given by

$$G(S) = G = \frac{N + SV}{M + SV} = S,$$

and thus $S = G$.

The parameters of S are identified on-line as $\hat{B}/\hat{A}$ and the minimum variance controller is given by

$$\frac{Q}{1 + \frac{Q\hat{B}}{\hat{A}}}.$$

An input disturbance centered at about 60 Hz is injected into the system via w_1 of Figure 2.6.

Figure 2.7 shows the power spectrum density of the ‘pes’ obtained in the above experiment. With the adaptive augmentation, an improvement of about 43% over the nominal internal controller is recorded. Figure 2.8 shows the error rejection as a function of frequency.

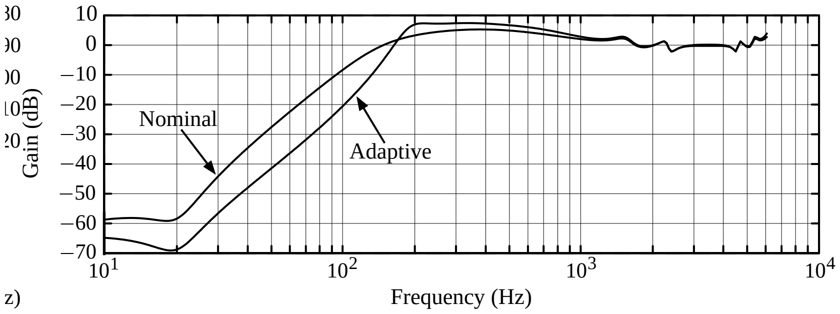


FIGURE 2.8. Error rejection function—nominal and adaptive

Main Points of Section

In this section, a case study on the control of two hard disk drives is described. Two optimal schemes are implemented, namely an ℓ_1 and an H_2 optimal scheme as described in Chapter 4 . The results show an improvement in performance over the case where these controllers are not present. For the second drive, an adaptive controller based on the scheme described in Chapter 5 is implemented. A further significant improvement in performance is recorded.

10.3 Control of a Heat Exchanger

Heat exchangers are extensively used in many industrial process installations such as power plants, chemical processing plants and oil refineries. In this study we use a laboratory scale heat exchanger as shown in Figure 3.1. Although many times smaller than its industrial counterparts, it is a good representation for studying the types of problems associated with such plants. A schematic of the heat exchanger is shown in Figure 9.3.5 which is redepicted here as Figure 3.2. The objective of the control is to maintain the temperature and fluid level of the hot tank at some preset value. These two variables can be manipulated by controlling the rate of flow of steam and cold fluid into the tank through appropriately placed electronic valves.

In this study, we will take the reader through the engineering cycle of commissioning a controller for such plants. We will first look at physical and structural

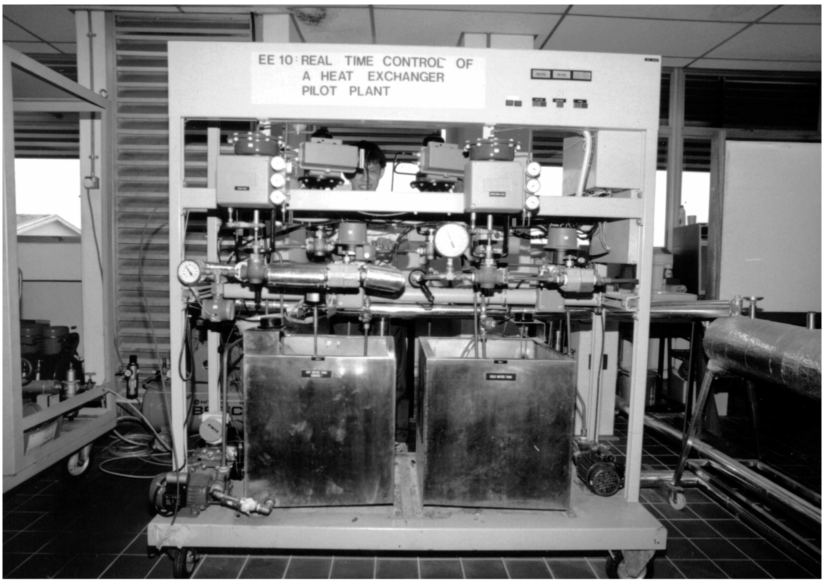


FIGURE 3.1. Laboratory scale heat exchanger

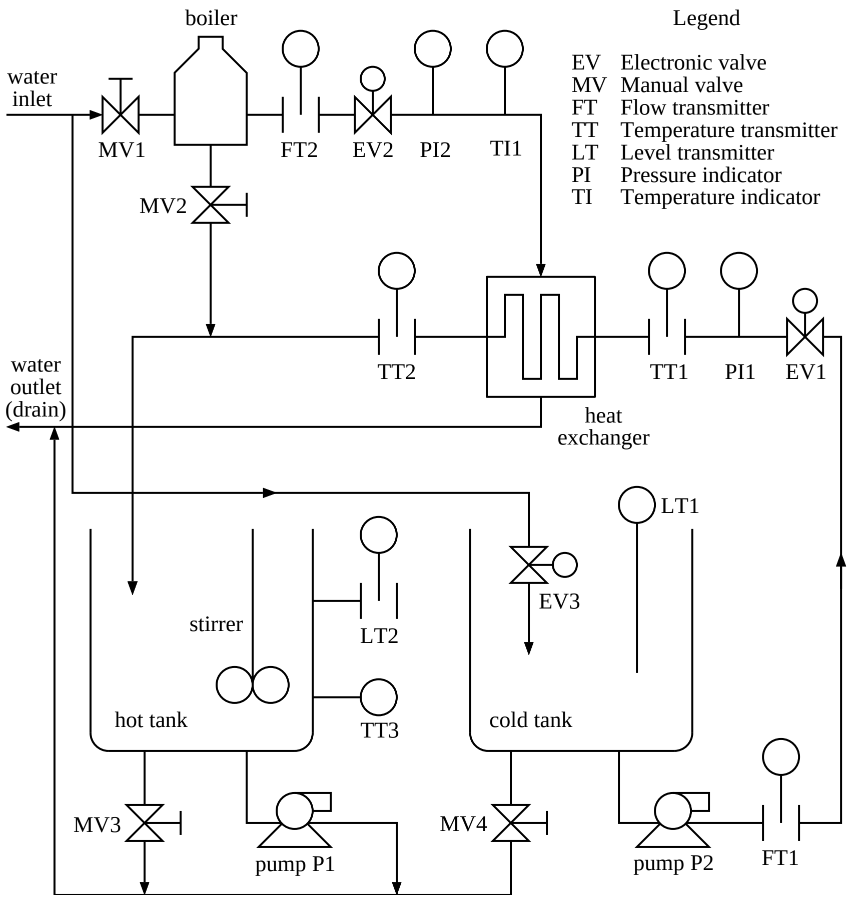


FIGURE 3.2. Schematic of heat exchanger

modeling to determine an approximate suitable representation of the plant. Once such a model structure is determined, parameter identification is then performed from data obtained through experimental trial runs. Based on the model obtained we design a simple LQG controller to control the plant. The results obtained from real-time control of the plant is then compared with simulation runs based on the model. In our study, simulation studies tally closely with that of the run on the actual plant, giving us good confidence in the accuracy of the plant model. We have also included a numerical plant model such that the interested reader may experiment with the high performance controllers described in the book.

Structural Modeling

The plant contains two 0.5 meter cubic steel tanks. One of the tanks serves as a buffer tank where cold water is pumped via the heat exchanger into the hot tank.

The flow of cold water into the heat exchanger is controlled by the pneumatic valve EV1. The hot tank is equipped with an outlet pump P1, a differential pressure level transducer (LT2) and a platinum resistance thermometer (TT3). The shell and tube heat exchanger consist of many round tubes mounted in a cylindrical shell with their axes parallel to that of the shell. The heat exchanger is designed as a steam heated system where water from the buffer tank flows through the inside of the tubes and is heated by steam flowing through the shell to supply the required heat. Steam required for the process is generated by a boiler. The flow of steam into the heat exchanger is controlled by the pneumatic valve EV2.

To perform an accurate physical modeling of the underlying process is complex. We have therefore simplified the derivation so that a good understanding of the process is obtained without being burdened with unnecessary complexities.

To obtain an understanding of the fluid level loop, we first maintain the steam input valve EV2 at a constant flow rate. Then the inflow rate $q(t)$ is related to level $h(t)$ in the hot tank by the equation

$$q(t) = A\frac{dh(t)}{dt}, \tag{3.1}$$

where A is the cross sectional area of the hot tank.

As designed and also observed, the dynamics of the pneumatic valves and level sensors are very much faster than the process. These dynamics are therefore ignored and the transfer characteristics are represented by the DC gains, respectively, G_{dc1}, G_{dcv}. The resultant transfer function from the valve EV1 to the level sensor LT2 is then given as

$$\frac{V_1(s)}{V(s)} = \frac{G_{dc1}G_{dcv}}{As}. \tag{3.2}$$

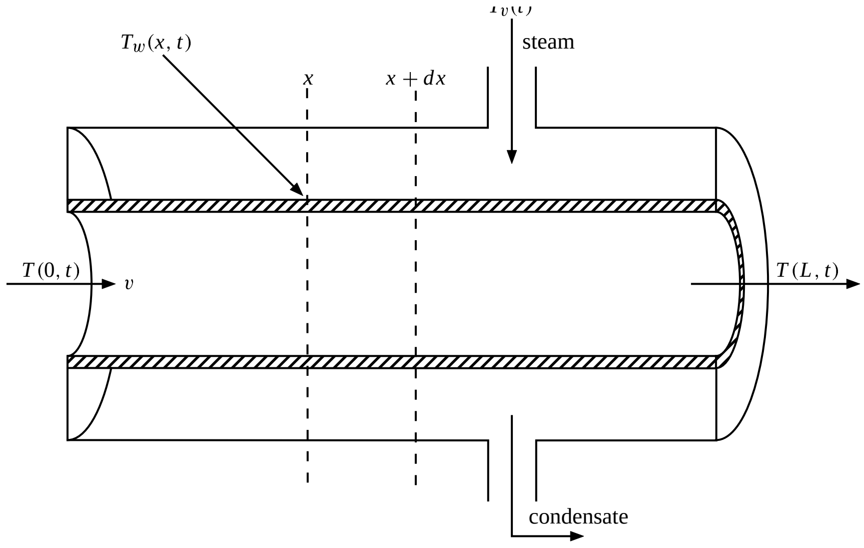
A simplified representation of the shell and tube heat exchanger is shown in Figure 3.3. The fluid that flows through the inner pipe at velocity v is heated by steam condensing outside the pipe. For simplicity let us ignore any spatial distribution of the steam temperature. The differential energy balance for the fluid inside the pipe over the volume element of length δx is given by

Rate of accumulation of internal energy

= Enthalpy in − Enthalpy out + Heat transferred, (3.3)

or symbolically,

$$\frac{\delta\left[A_i p\delta x C(T - T_r)\right]}{\delta t} = vA_i pC(T - T_r) - vA_i pC\left[(T + \frac{\delta T}{\delta x}\delta x) - T_r\right] + \pi D_i h_i \delta x(T_w - T), \tag{3.4}$$



Symbols

$T(x,t)$	fluid temperature
$T_W(x,t)$	wall temperature
$T_v(t)$	steam temperature
T_r	reference temperature for evaluating enthalpy
p	density of fluid
C	heat capacity of fluid
p_w	density of metal in wall
A_i	cross-sectional area inside pipe
A_w	cross-sectional area of metal wall
D_i	inside diameter of inner pipe
D_o	outside diameter of pipe
h_i	convective heat transfer coefficient inside pipe
h_o	heat-transfer coefficient for condensing steam
v	fluid velocity

FIGURE 3.3. Shell-tube heat exchanger

and therefore

$$\frac{\delta T}{\delta t} = -v\frac{\delta T}{\delta x} + \frac{T_w - T}{\tau_1}, \qquad \frac{1}{\tau_1} = \frac{\pi D_i h_i}{A_i pC}. \tag{3.5}$$

Now, the energy balance equation for the metallic wall over the volume of length δx is stated as follows:

Accumulation of energy in wall

$=$ Heat transfer in through steam $-$ Heat transfer out through fluid film, (3.6)

that is,

$$A_w \delta x p_w C_w \frac{\delta T_w}{\delta t} = \pi D_0 h_0 \delta x (T_V - T_w) - \pi D_i h_i \delta x (T_w - T), \tag{3.7}$$

giving rise to

$$\frac{\delta T_w}{\delta t} = \frac{1}{\tau_{22}}(T_V - T_w) - \frac{1}{\tau_{12}}(T_w - T), \tag{3.8}$$

where

$$\frac{1}{\tau_{12}} = \frac{\pi D_i h_i}{A_w p_w C_w}, \qquad \frac{1}{\tau_{22}} = \frac{\pi D_0 h_0}{A_w p_w C_w}. \tag{3.9}$$

Taking the Laplace transform of (3.8) and (3.5) and solving them simultaneously, we have

$$\frac{\delta T(x,s)}{\delta x} + \frac{a(s)}{v} T(x,s) = \frac{b(s)}{v} T_v(s), \tag{3.10}$$

where

$$\begin{aligned} a(s) &= s + \frac{1}{\tau_1} - \frac{\tau_{22}}{\tau_1(\tau_{12}\tau_{22}s + \tau_{12} + \tau_{22})}, \\ b(s) &= \frac{\tau_{12}}{\tau_1(\tau_{12}\tau_{22}s + \tau_{12} + \tau_{22})}, \end{aligned}$$

which is an ordinary first order differential equation with boundary condition $T(x,s) = T(0,s)$ at $x = 0$. Solving this equation gives

$$T(x,s) = T(0,s) + \left(1 - e^{-\frac{a(s)}{v}x}\right)\left(\frac{b(s)}{a(s)} T_v(s) - T(0,s)\right), \tag{3.11}$$

where $T(0,s)$ is the transform of the fluid temperature at the entrance to the heat exchanger and $T_v(s)$ is the transform of the steam temperature. If the temperature of the fluid entering the pipe does not vary significantly, which holds true if the buffer tank is kept fairly constant, the transfer function relating the exit fluid temperature to the steam temperature is

$$\frac{T(L,s)}{T_v(s)} = \frac{b(s)}{a(s)}\left(1 - e^{-(\frac{a(s)}{v})L}\right). \tag{3.12}$$

The transfer function relating the temperature of the inflow water from the outlet of the heat exchanger to the temperature of water in the hot tank can be derived from the energy balance equation

$$\begin{aligned} &\text{Accumulation of total energy in } \delta t \\ &\quad = \text{input energy} - \text{output energy} - \text{energy lost to the environment}, \end{aligned} \tag{3.13}$$

giving

$$p_f A_f h_f C_f \frac{\delta(T(t) - T_r)}{\delta t} = p_f q_0 C_f (T(L,t) - T_r) - p_f q_0 C_f (T - T_r) - 0. \tag{3.14}$$

Since the flow rate of fluid through the heat exchanger is assumed constant as q_0, then h_f is a constant. Taking the Laplace transform of (3.14) and then solving with (3.12), we get

$$\frac{T(s)}{T_v(s)} = \frac{b(s)\left(1 - e^{-(\frac{a(s)}{v})L}\right)}{a(s)(\frac{A_f h_f}{q_0}s + 1)}. \tag{3.15}$$

Again assuming that the dynamics of EV2 is very much faster than the change in temperature of the tank, so that it can be ignored and be replaced by its DC gain K_{dc}, the relationship between the input voltage to EV2 and the temperature of fluid in the hot tank is given by

$$\frac{T(s)}{V(s)} = \frac{K_{dc}b(s)\left(1 - e^{-(\frac{a(s)}{v})L}\right)}{a(s)\left(\frac{A_f h_f}{q_0}s + 1\right)}. \tag{3.16}$$

Parameter Identification

To determine the sampling rate, step inputs are applied separately to both the pneumatic valves, EV1 and EV2, and the corresponding effects on the temperature and level of the hot tank are determined. It is found that the temperature and level processes have time constants of approximately 46 and 13 minutes, respectively. A sampling period of one minute is therefore chosen to reflect a sampling rate about 13 times that of the faster process.

Next, pseudo random binary sequences (PRBS) of amplitude ± 1 are applied simultaneously to the two pneumatic valves. The corresponding temperature and level readings are then logged at one minute intervals. These are shown in Figures 3.4 and 3.5. An off-line least square algorithm is then used to estimate the parameters of a linearized autoregressive exogenous (ARX) input model of the form

$$\square(q^{-1})y_k = \square(q^{-1})u_{k-n} + e_k,$$

where n is number of delay samples.

Parameters are estimated for various model orders and sample delays. The following estimated transfer functions yield the smallest residual errors.

$$\begin{aligned}
G_{11} &= \frac{-0.0862z^{-1}}{1 - 0.9955z^{-1}},\\
G_{12} &= \frac{-0.0076z^{-1}}{1 - 0.9955z^{-1}},\\
G_{21} &= \frac{-0.0011z^{-1} + 0.0122z^{-2} + 0.0125z^{-3} + 0.0032z^{-4}}{1 - 1.2629z^{-1} + 0.3614z^{-2} - 0.1078z^{-3} + 0.058z^{-4}},\\
G_{22} &= \frac{-0.0055z^{-1} + 0.0130z^{-2} + 0.020z^{-3} + 0.002z^{-4}}{1 - 1.2629z^{-1} + 0.3614z^{-2} - 0.1078z^{-3} + 0.058z^{-4}}.
\end{aligned}$$

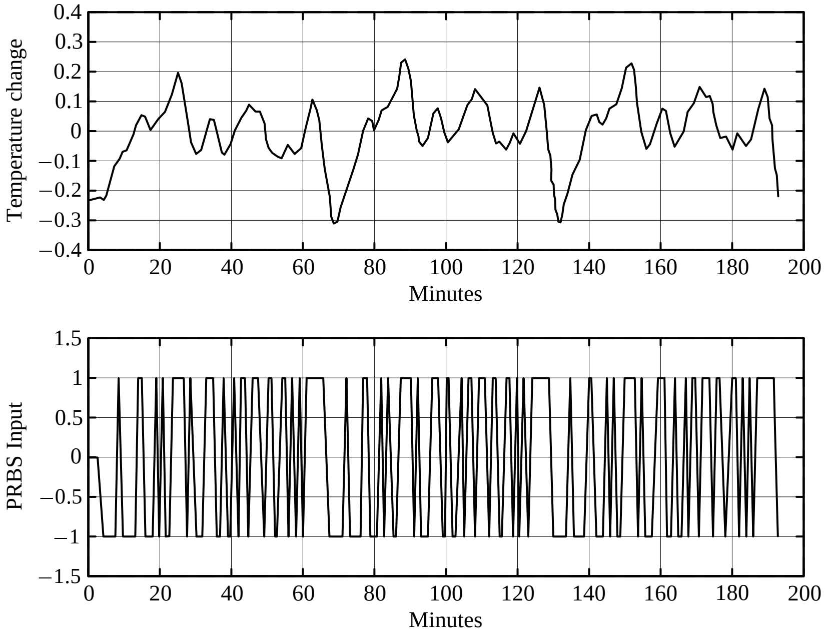
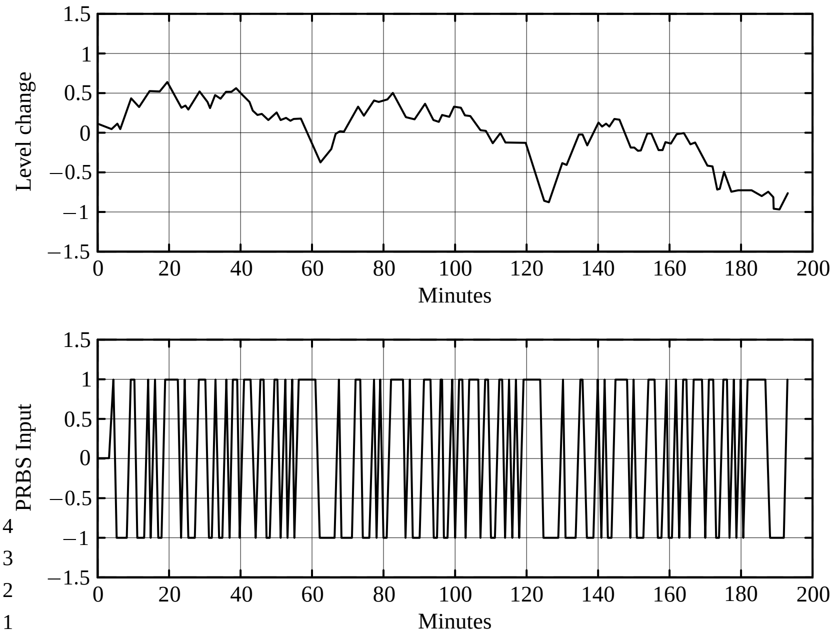


FIGURE 3.4. Temperature output and PRBS input signal



4
3
2
1

FIGURE 3.5. Level output and PRBS input signal

In state space form, G is given as follows.

$$A = \begin{bmatrix} -0.1441 & -0.3597 & 0.3392 & -0.2646 & 0 \\ 0.0859 & -0.0339 & 0.0912 & 1.772 & 0 \\ 0 & 1.3505 & 0.8111 & -0.230 & 0.3392 \\ 0 & 0 & 0.845 & 0.6298 & 0 \\ 0 & 0 & 0 & 0 & 0.9955 \end{bmatrix}, \tag{3.17}$$

$$B = \begin{bmatrix} 0.4116 & 0.2553 \\ 0.2946 & -0.5298 \\ 0.4488 & -0.6430 \\ -0.0363 & -0.1814 \\ -0.9961 & -0.0878 \end{bmatrix}, \tag{3.18}$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0.0865 \\ 0 & 0 & 0 & 0 & 0 & 0.0303 & 0 \end{bmatrix}, \tag{3.19}$$

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}. \tag{3.20}$$

Feedback Control

The model obtained is used to design an LQG controller. To achieve zero steady state offset in the closed-loop system, integral action is included into the system by defining two more states as follows.

$$x_{IL}(k+1) = x_{IL}(k) + Y_{\text{level}}(k) - R_{\text{levelref}}(k), \tag{3.21}$$

$$x_{TL}(k+1) = x_{TL}(k) + Y_{\text{temp}}(k) - R_{\text{tempref}}(k), \tag{3.22}$$

where R_{levelref}, R_{tempref} and Y_{level}, Y_{temp} are temperature and level reference inputs and outputs signals respectively.

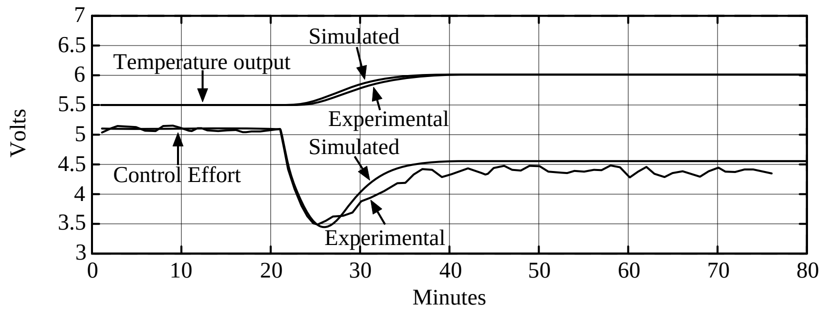


FIGURE 3.6. Temperature response and control effort of steam valve due to step change in both level and temperature reference signals

The resultant augmented state equation is then given as

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0.0865 \\ 0 & 1 & 0 & 0 & 0 & 0.0303 & 0 \\ 0 & 0 & -0.1441 & -0.3597 & 0.3392 & -0.2646 & 0 \\ 0 & 0 & 0.0859 & -0.0339 & 0.0912 & 1.772 & 0 \\ 0 & 0 & 0 & 1.3505 & 0.8111 & -0.230 & 0 \\ 0 & 0 & 0 & 0 & 0.845 & 0.6298 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.9955 \end{bmatrix}, \tag{3.23}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.4116 & 0.2553 \\ 0.2946 & -0.5298 \\ 0.4488 & -0.6430 \\ -0.0363 & -0.1814 \\ -0.9961 & -0.0878 \end{bmatrix}, \tag{3.24}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0865 \\ 0 & 0 & 0 & 0 & 0 & 0.0303 & 0 \end{bmatrix}. \tag{3.25}$$

The performance index is defined as

$$J = \frac{1}{2} \sum_{k=1}^{N} \left(x'(k) Q_c x(k) + u'(k) R_c u(k) \right), \tag{3.26}$$

where

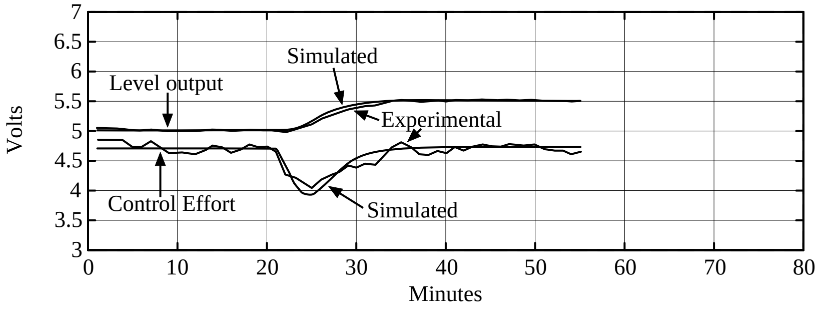


FIGURE 3.7. Level response and control effort of flow valve due to step change in both level and temperature reference signals

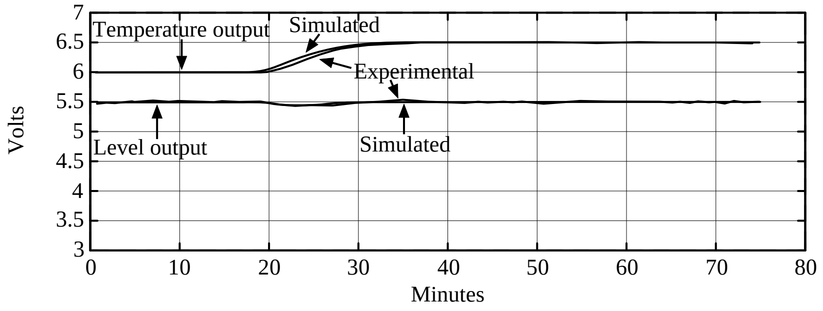


FIGURE 3.8. Temperature and level response due to step change in temperature reference signal

$$Q_c = \begin{bmatrix} 1.1 & 0 & 0 & 0 \\ 0 & 1.8 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 24 \end{bmatrix}, \qquad R_c = \begin{bmatrix} 0.8 & 0 \\ 0 & 1 \end{bmatrix}. \tag{3.27}$$

The estimator gain is designed using a process noise covariance R_v and measurement noise covariance R_w with

$$R_v = \begin{bmatrix} 0.024\,4^2 & 0 \\ 0 & 0.003^2 \end{bmatrix}, \qquad R_w = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{3.28}$$

The resultant controller is then implemented in real-time using MATLAB. The results are benchmarked against the ideal case, which is a simulation with the designed controller controlling the earlier model obtained.

Figures 3.6 and 3.7 show the plots when simultaneous 0.5 V step changes are applied to both the level and the temperature reference signal whereas Figures 3.8 and 3.9 show the responses to just a temperature reference signal step of 0.5 V.

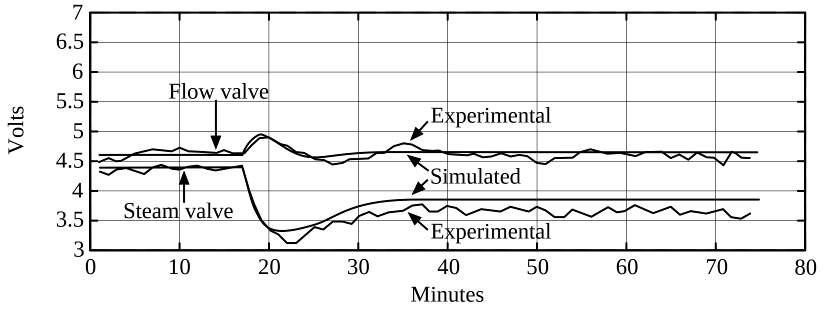


FIGURE 3.9. Control effort of steam and flow valves due to step change in temperature reference signal

10.4 Aerospace Resonance Suppression

This study is not strictly a case study in that the controllers have not been implemented on aircraft to our knowledge. Rather, it is a feasibility study using the best available linear aircraft models. Simulation results are presented for wing tip accelerometer control of high order models of a supersonic airplane. Of particular interest is the suppression of resonances in certain frequency bands. Similar results have been achieved for high speed civil transport aircraft models. We do not give the model or design details here, but simply show performance spectral plots and discuss the various design issues.

Introduction

Aircraft high frequency structural resonance modes can be excited in certain regions of the flight envelope. At the extremes of this envelope, such resonances lead to wing flutter and catastrophic failure. Because of a degree of uncertainty in aircraft models, such resonances are known to be extraordinarily difficult to suppress by active means.

In this study, a combination of a robust controller and an adaptive scheme is used to control high frequency structural modes for aircraft models of 100th order or so. The objective is to suppress wing flexure or body bending resonances in the vicinity of 20 to 80 rad/s by means of aileron, or rudder control. Certainly, it is imperative that these modes not be excited by naive control actions. The sensors could be accelerometers on the wing tips or body extremities. Robust controllers by themselves may not achieve adequate performance over the entire range of situations of interest, so that there would be a role for adaptive augmentations to such robust controllers, at least at the extremes such as in emergency dive situations. This study is a step to explore such a role using realistic high order aircraft models.

Technical Approach

There are three high order aircraft flying models which correspond to an aircraft flying at altitudes of 2 000, 10 000, and 30 000 ft flight conditions, respectively. Spectral analysis indicates that the models exhibit two excessive wing flexure resonances which are to be suppressed.

We select the aircraft model which corresponds to a flight condition at an altitude of 2 000 ft as the basis for the nominal plant. For this model, there are two high resonance peaks in its power spectral density function at frequencies 27.5 and 56.4 rad/s respectively. It makes sense then to limit interest to the frequency band below 60 rad/s.

Since the model is very high in order (107th order) and high frequency disturbance responses above 60 rad/s are beyond our interest, it is reasonable to reduce the plant model order as low as possible consistent with obtaining a good controller design. A benefit is to reduce the complexity for the nominal plant controller.

Next we design an LQG controller K for the reduced order plant, namely the nominal plant G. Of course, there is a quadratic performance index penalizing the disturbance responses e.

Now applying the rationale of the direct adaptive-Q schemes of Chapter 6, we consider further an augmentation of the stabilizing controller K for G to achieve an augmented controller, denoted $K(Q)$, which parameterizes all the stabilizing controllers for G in terms of an arbitrary proper stable transfer function Q. For different Q, the controller $K(Q)$ will have different robustness and performance properties when applied to the nominal plant. Here we take Q to be an adaptive filter.

The adaptive filter Q is included so that in any on-line adaptation to plants other than the nominal plant G, denoted $\bar{G}$, the filter Q can be selected so as to minimize some disturbance response, perhaps a frequency shaped version of e_k, the response penalized in the LQG design.

In this latter case, the adaptive scheme and fixed controller work towards the same objective in the appropriate frequency bands.

Model Reduction

For a first cut model reduction, we consider the eigenvalue distribution, and remove all the modes which are above the frequency 155 rad/s, being far beyond the frequency band of interest. This reduces the plant to 85th order. Then we discretize this model at the sampling time 0.03 s, and further reduce the model to 46th order by deleting those states which correspond to (stable) near pole/zero cancellations and also remove those states which are outside of the frequency band of interest. We select this reduced order discrete-time aircraft model as the nominal plant G. Other methods based on balanced realizations, see Anderson and Moore (1989), are not used at this stage because of possible numerical problems.

Design of Nominal Controller K

In order to design a nominal controller K, we proceed here with a straightforward LQG controller design for the nominal plant G. Since we aim at reducing the peaks of the power spectral density function, an LQ index is employed which weights states associated with the resonance peaks. We define the disturbance response to be $e = [\, e_1 \; e_2 \; e_3 \,]'$ where e_1 is the contribution of the states towards the first resonance mode, namely at 27.5 rad/s, e_2 is the contribution towards the second mode at 56.4 rad/s and e_3 is $e_3 = e_1 + e_2$. By having different weighting factors on those responses, we design an LQ controller for the nominal plant. Our selection of the kernel of the cost index chosen is $\left(4.5e_1^2 + e_2^2 + 10e_3^2 + 5\,500u^2\right)$. These are selected by a trial and error approach.

For the Kalman filter design, we select a stochastic disturbance input to the plant model which excites the resonances, details are omitted.

Frequency Shaping for the Disturbance Response

Since the disturbance response e for the LQ design has significant energy over a wide frequency range, and yet our concern is only with its energy in a narrow frequency range, it may appear natural to pass it through a frequency shaping filter so as to focus mainly on the responses within the frequency band of interest. However, with a negligible weighting outside the frequency band of interest, adaptation to minimize the energy of the frequency shaped e may excite resonances outside this frequency band and sacrifice performance. Here we exploit the Kalman filter in the controller design to achieve a filtered disturbance response achieved by replacing states by their state estimates.

Of course, an H_∞ design approach for K is also possible, and in our experience is more direct, but it obscures to some extent the useful engineering insight as to how the controller is achieving its objectives.

Frequency Shaping for the Residuals

The Kalman filter residuals r and adaptive Q filter output s are filtered in an adaptive Q scheme as discussed in Chapter 6. It may seem that these filters are as high order as the nominal plant. However, for our specific LQG design, there results a filter which can be reduced to as low as 4th order by just deleting unobservable states.

Order of Q Selection

The order of the adaptive controller Q directly affects the complexity of the adaptive scheme. The number of the coefficients in Q determine the order of the on-line least squares parameter updating scheme. Another consideration to be kept in mind is the stability of Q, because the closed-loop system is stable in the case of the nominal plant G only when Q is stable. With finite impulse response (FIR) Q, the stability of Q is trivially satisfied with bounded gains, and there is a consequent simplification of the adaptive scheme. In our simulations, a 4th order FIR model of Q is employed, there being diminishing returns from increasing the order. With different sampling times, a different order could be more appropriate.

It is also possible and in practice reasonable to include a fixed *prefilter* P_r in the adaptive-Q loop to limit the frequency band of adaptive feedback signals and avoid exciting those frequency modes which may destabilize the closed-loop system. This is particularly important when there are high frequency unmodeled dynamics. It is clear that the inclusion of any stable prefilter in the adaptive-Q loop will not alter the optimization task, but only change the space of adaptive-Q action. It could well make sense to incorporate the prefilter P_r into the state estimator, so that its output corresponds to a subset of the states. For example, in the case of a resonance or flutter suppression, the resonance or flutter state estimate could well be appropriate as an input to the adaptive-Q filter.

Three Flight Conditions

As mentioned above, we have defined a nominal plant G and designed a direct adaptive-Q scheme along the lines described in Chapter 6. The performance of interest is chosen as the peak of the power spectral density function. The simulation results when applied in a number of situations are reported to see its robustness and performance over a range of uncertainties and disturbances.

Nominal Condition

First, we apply to the nominal (reduced order) plant model G the adaptive-Q scheme based on the LQG controller K. The power spectral density function of the output versus noise is shown in Figure 4.1. The response indicated by (a) is that of the closed-loop scheme with the off-line designed robust linear controller, and the response indicated by (b) is that of the direct adaptive-Q scheme. From the figure it is clear that the adaptive scheme improves the performance, namely reduces the peaks of the power spectral density function of the output from the noise by approximately 30%, at the expense of boosting higher frequency modes. This improvement is achieved after the nominal controller has reduced the peaks by an order of magnitude.

Flight at 2 000 ft

As mentioned before, the nominal plant is a reduced order plant based on the aircraft model 2 000, namely the model corresponding to the flight condition at altitude of 2 000 ft. When we directly employ the adaptive scheme for the nominal plant to the full order model, all the high frequency resonances are activated and appear in the adaptive-Q loop. Recall that we can insert a prefilter in the adaptive-Q loop to limit the feedback of the residuals in only the frequency bands of interest and avoid exciting high frequency unmodeled dynamics. With a prefilter included, the comparison between the adaptive scheme and the closed-loop responses for the model 2 000, not shown here, tells us that performance is marginally improved by the adaptive scheme at the first resonance peak, and kept the same as that of the nonadaptive but robust scheme at high frequencies above

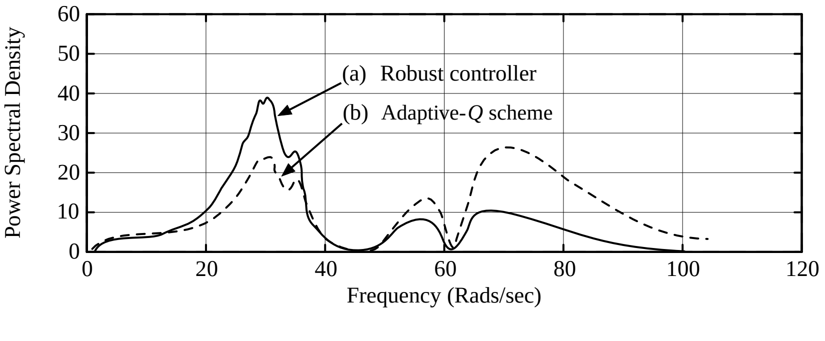


FIGURE 4.1. Comparative performance at 2 000 ft

60 rad/s. The prefilter employed in the adaptive scheme is a 16th order low pass filter.

Flight at 10 000 ft

The nominal plant is now replaced by a different plant, viz. model 10 000, which corresponds to the flight condition at an altitude of 10 000 ft. The nominal controller K, and the adaptive-Q scheme with prefilter are here taken to be the same as for the model 2 000. The performance of the adaptive control scheme is shown in Figure 4.2 as compared to that for the nonadaptive case. The adaptive scheme improves the performance of the closed-loop systems in this case but the improvement is very limited.

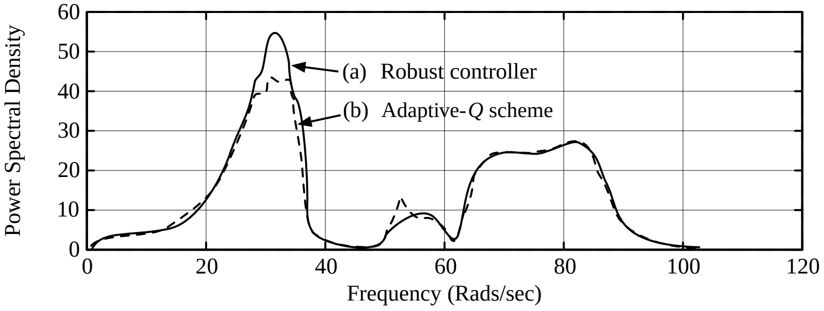


FIGURE 4.2. Comparative performance at 10 000 ft

Flight at 30 000 ft

The nominal plant is here replaced by a different plant again, viz. model 30 000, which corresponds to the flight condition at altitude of 30 000 ft. Again the nominal controller K and the adaptive scheme setup are unchanged. In this case the adaptive scheme does not deliver improvement which assures us of the quality of the robust design.

Remarks.

1. The off-line LQG controller for the nominal plant G is to us unexpectedly robust for the three plants: model 2 000, model 10 000 and 30 000. In fact the robustness-performance trade off is such that the potential for improvement via added adaptive loops can not be dramatic. On the other hand, should the controller not be suitably robust, performance enhancement by adaptive techniques could be futile. Furthermore, should there be a dramatic improvement due to the adaptive-Q loop, it would be important to question whether there should be an improved robust design so as to reduce this improvement to a more appropriate level.

2. The simulation results for the adaptive scheme are encouraging in that they demonstrate that resonance suppression can occur based on on-line processing. The adaptive results do not represent a dramatic improvement over an off-line robust LQG design, although a more dramatic improvement is not precluded for other flight conditions, or variations on the adaptive scheme. Our first objective has been a conservative one, rather than to achieve spectacular results which may be constructed as "lucky", as in earlier flutter suppression studies Moore, Hotz and Gangsaas (1982). It appears that our conservative objectives have been achieved.

3. With increases to the gain on the filtered disturbance response, or without a prefilter in the adaptive-Q loop to limit the frequency band of residuals fed back through the Q loop, the adaptive scheme can be made to destabilize. This is expected since there can only be local stability results in the presence of significant unmodeled dynamics, especially at high frequency.

4. The algorithm as studied in the simulation is impractical to implement because of the high order. Essentially the same results are achieved working with reduced order filters so as to achieve a more practical design.

5. In performing the simulations, the first 1 000 iterations are run with the only controller being K. During this period the least squares covariance matrix P_k is being updated. Then the loop is closed with zero initial condition on the $Q(z^{-1})$. After a further 100 iterations the adaptive $Q(z^{-1})$ has virtually "converged". For 10 000 iterations, a power spectral density measurement is taken. No attempt has been made at this stage to track time-varying plants.

6. Our approach has been applied to models of completely different aircraft with different resonance suppression problems, namely, body bending resonances rather than wing flexure resonances. Similar results seem to be achieved. To illustrate, results for two flight conditions of a transport aircraft model are presented in Figure 4.3, Figure 4.4. Here the open-loop resonances are shown since they are merely a factor of three above those under active control.

Flutter Suppression

In order to illustrate that dramatic performance improvement can be achieved in an adaptive-Q approach, we present simulation results from Moore, Hotz and Gangsaas (1982). The unstable flutter results from a wing bending mode and torsion mode coming together at the flutter frequency and one mode becoming unstable. This will happen to any wing at sufficiently high speed, termed the flutter speed.

In this study of a 65th order flexible wing aircraft model, the adaptive-Q filter is driven from the flutter state estimate. Indirect adaptive-Q techniques are applied.

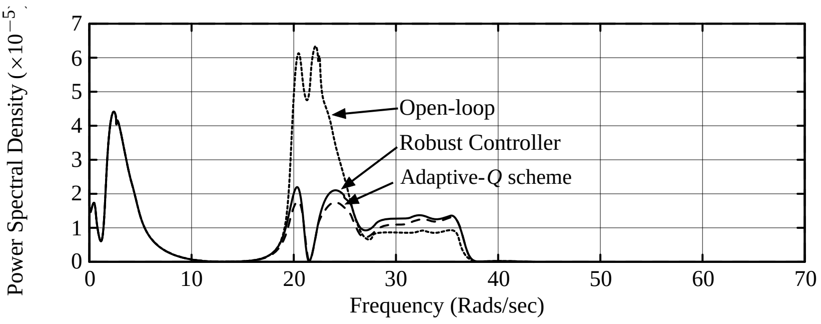


FIGURE 4.3. Comparisons for nominal model

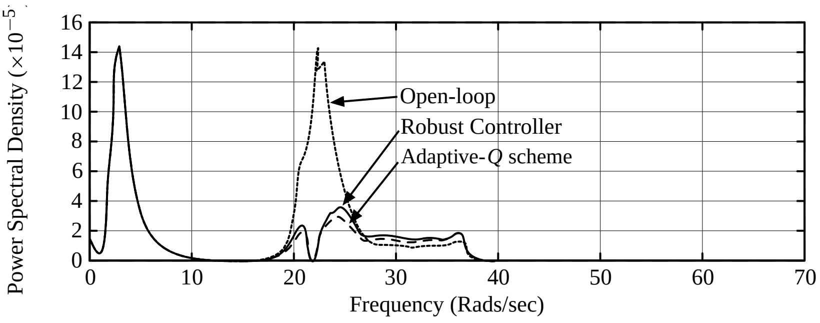


FIGURE 4.4. Comparisons for a different flight condition than for the nominal case

In particular, a second order model uncertainty is identified looking at the control input (wing tip aileron) prior to its entry to the estimator and the flutter state estimate (from wing tip accelerometers). An adaptive-Q filter is applied to assign the closed-loop poles to stable locations at the flutter frequency. The degree of assigned stability is not set to be “too” large so as to avoid excessive control actions which could excite lightly damped modes. Figure 4.5 shows that a flutter instability is controlled effectively in a few cycles, before the wing “falls off”. Indeed there is demonstrated in the simulations “180° phase margin” in a region of loop gain greater than unity!

Main Points of Section

The simulation results for resonance suppression at this stage are encouraging in that the off-line designed fixed LQG controller gives robust performance, and the adaptive-Q scheme is seen to only improve the performance further. Of course some engineering is required to achieve robust LQG designs, associated pre-filters, and adjustment law gains to achieve such success. In some situations the adaptive-Q methodology can achieve dramatic results by achieving “180° phase margins”, as in the case of flutter suppression.

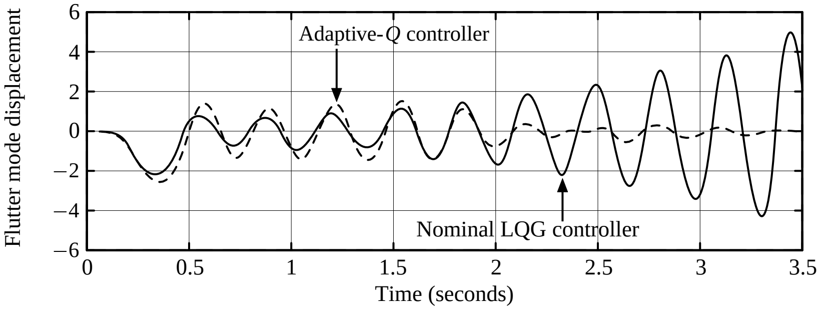


FIGURE 4.5. Flutter suppression via indirect adaptive-Q pole assignment

10.5 Notes and References

Resonance Suppression

Our first studies in this topic are reported in Moore, Hotz and Gangsaas (1982) where an indirect adaptive-Q approach is used to suppress catastrophic wing flutter. Indeed, this study provided many of the insights used subsequently for developing the theory of the adaptive-Q approach. The Q filter in this study achieved a weakened form of minimum variance control. Later studies sought to achieve adaptive-Q filters based on LQG and pole assignment indices(Chakravarty and Moore, 1985; Chakravarty and Moore, 1986). Subsequently, less ambitious studies in resonance suppression of lightly damped modes using direct adaptive-Q techniques were employed, thereby checking the validity of this approach, see also Moore, Xia and Xia (1989). Other resonance suppression studies have helped develop our understanding, namely Irlicht, Mareels and Moore (1993) and Telford and Moore (1990).

APPENDIX **A**

Linear Algebra

This appendix summarizes the key results of matrix theory and linear algebra results used in this text. For more complete treatments, see Barnett (1971) and Bellman (1970).

A.1 Matrices and Vectors

Let R and C denote the fields of real numbers and complex numbers, respectively. The set of integers is denoted $\mathsf{N} = \{1, 2, \ldots\}$.

An $n \times m$ *matrix* is an array of n rows and m columns of elements x_{ij} for $i = 1, \ldots, n, \;\; j = 1, \ldots, m$ as

$$X = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1m} \\ x_{12} & x_{22} & \ldots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} \end{bmatrix} = (x_{ij}), \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = (x_i).$$

The matrix is termed *square* when $n = m$. A column *n-vector* x is an $n \times 1$ matrix. The set of all n-vectors (row or column) with real arbitrary entries, denoted R^n, is called n-space. With complex entries, the set is denoted C^n and is called complex n-space. The term *scalar* denotes the elements of R, or C. The set of real or complex $n \times m$ matrices is denoted by $\mathsf{R}^{n \times m}$ or $\mathsf{C}^{n \times m}$, respectively. The *transpose* of an $n \times m$ matrix X, denoted X', is the $m \times n$ matrix $X' = (x_{ji})$. When $X = X'$, the square matrix is termed *symmetric*. When $X = -X'$, then the matrix is *skew symmetric*. Let us denote the complex conjugate transpose of a matrix X as $X^* = \bar{X}'$. Then matrices X with $X = X^*$ are termed *Hermitian* and with $X = -X^*$ are termed *skew Hermitian*. The *direct sum*, of two square matrices X, Y, denoted $X \dot{+} Y$, is $\left[\begin{smallmatrix} X & 0 \\ 0 & Y \end{smallmatrix}\right]$ where 0 denotes a zero matrix of appropriate dimensions consisting of zero elements.

A.2 Addition and Multiplication of Matrices

Consider matrices $X, Y \in \mathsf{R}^{n\times m}$ or $\mathsf{C}^{n\times m}$, and scalars $k, \ell \in \mathsf{R}$ or C. Then $Z = kX + \ell Y$ is defined by $z_{ij} = kx_{ij} + \ell y_{ij}$. Thus $X + Y = Y + X$ and addition is commutative. Also, $Z = XY$ is defined for X an $n \times p$ matrix and Y an $p \times m$ matrix by $Z = (z_{ij}),\ z_{ij} = \sum_{k=1}^{p} x_{ik}y_{kj}$, and is an $n \times m$ matrix. Thus $W = XYZ = (XY)Z = X(YZ)$ and multiplication is associative. Note that when $XY = YX$, which is not always the case, we say that X, Y are commuting matrices.

When $X'X = I$ and X is real then X is termed an *orthogonal* matrix and when $X^*X = I$ with X complex it is termed a *unitary* matrix. Note real vectors x, y are *orthogonal* if $x'y = 0$ and complex vectors are orthogonal if $x^*y = 0$. A *permutation matrix* π has exactly one unity element in each row and column and zeros elsewhere. Every permutation matrix π is orthogonal. An $n \times n$ square matrix X with only diagonal elements and all other elements zero is termed a *diagonal* matrix and is written $X = \operatorname{diag}(x_{11}, x_{22} \dots, x_{nn})$. When $x_{ii} = 1$ for all i and $x_{ij} = 0$ for $i \neq j$, then X is termed an *identity* matrix, and is denoted I_n, or just I. Thus for an $n \times m$ matrix Y, then $YI_m = Y = I_nY$. A *sign* matrix S is a diagonal matrix with diagonal elements $+1$ or -1. Every sign matrix is orthogonal.

For $X \in \mathsf{R}^{m\times n}$ and $Y \in \mathsf{R}^{p\times r}$ denote by $X \otimes Y$ the matrix of dimension $mp \times rn$, a block matrix having as ijth block the matrix $x_{ij}Y$.

A.3 Determinant and Rank of a Matrix

A recursive definition of the determinant of a square $n \times n$ matrix X, denoted $\det(X)$, is

$$\det(X) = \sum_{j=1}^{n} (-1)^{i+j} x_{ij} \det(X_{ij}),$$

where $\det(X_{ij})$ denotes the determinant of the submatrix of X constructed by deleting the ith row and the jth column. The determinant of a scalar x is the scalar x itself. The element $(-1)^{i+j}\det(X_{ij})$ is termed the *cofactor* of x_{ij}. The square matrix X is said to be a *singular* matrix if $\det(X) = 0$, and a *nonsingular* matrix otherwise. It can be proved that for square matrices $\det(XY) = \det(X)\det(Y)$.

For $X \in \mathsf{R}^{n\times p},\ Y \in \mathsf{R}^{p\times n}$ we have $\det(I_n + XY) = \det(I_p + YX)$. In particular with $p = 1$, for $x, y \in \mathsf{R}^n$ $\det(I_n + XY') = 1 + Y'X$.

The *rank* of an $n \times m$ matrix X, denoted by $\operatorname{rk}(X)$ or $\operatorname{rank}(X)$, is the maximum positive integer r such that some $r \times r$ submatrix of X, obtained by deleting rows and columns is nonsingular. Equivalently, the rank is the maximum number of linearly independent rows and columns of X. If r is either m or n then X is *full rank*. It is readily seen that

$$\operatorname{rank}(X) + \operatorname{rank}(Y) - m \le \operatorname{rank}(XY) \le \min(\operatorname{rank}(X), \operatorname{rank}(Y)).$$

A.4 Range Space, Kernel and Inverses

For an $n \times m$ matrix X, the *range space* or the *image space* $\square(X)$, also denoted by $\mathrm{Im}(X)$, is the set of vectors Xy where y ranges over the set of all m vectors. Its dimension is equal to the rank of X. The *kernel* $\ker(X)$ of X is the set of vectors z for which $Xz = 0$. It can be seen that for real matrices, $\square(X')$ is orthogonal to $\ker(X)$, or equivalently, with $y_1 = X'y$ for some y and if $Xy_2 = 0$, then $y_1' y_2 = 0$.

For a square *nonsingular matrix* X, there exists a unique *inverse* of X, denoted X^{-1}, such that $X^{-1}X = XX^{-1} = I$. The ijth element of X^{-1} is given from $\det(X)^{-1} \times$ cofactor of x_{ji}. Thus $(X^{-1})' = (X')^{-1}$ and $(XY)^{-1} = Y^{-1}X^{-1}$ where the inverses exist. More generally, a unique (Moore-Penrose) *pseudo-inverse* of X, denoted $X^{\#}$, is defined by the characterizing properties $X^{\#}Xy = y$ for all $y \in \square(X')$ and $X^{\#}y = 0$ for all $y \in \ker(X')$. Thus if $\det(X) \neq 0$ then $X^{\#} = X^{-1}$, if $X = 0$, $X^{\#} = 0$, $(X^{\#})^{\#} = X$, $X^{\#}XX^{\#} = X^{\#}$, $XX^{\#}X = X$.

For a nonsingular $n \times n$ matrix X, a nonsingular $p \times p$ matrix A and an $n \times p$ matrix B, then provided inverses exist, the *Matrix Inversion Lemma* states

$$\begin{aligned}(I + XBA^{-1}B')^{-1}X &= (X^{-1} + BA^{-1}B')^{-1} \\ &= X - XB(B'XB + A)^{-1}B'X,\end{aligned}$$

and

$$\begin{aligned}(I + XBA^{-1}B')^{-1}XBA^{-1} &= (X^{-1} + BA^{-1}B')^{-1}BA^{-1} \\ &= XB(B'XB + A)^{-1}.\end{aligned}$$

A.5 Eigenvalues, Eigenvectors and Trace

For a square $n \times n$ matrix X, the *characteristic polynomial* of X is $\det(zI - X)$ and its *real or complex* zeros are the *eigenvalues* of X, denoted λ_i. The spectrum $\mathrm{spec}(X)$ of X is the set of its eigenvalues. The *Cayley-Hamilton Theorem* tells us that X satisfies its own *characteristic equation* with $\det(zI - X) = p(z)$, $j(X) = 0$. For eigenvalues λ_i, then $Xv_i = \lambda_i v_i$ for some nonzero real or complex vector v_i, termed an *eigenvector*. The real or complex vector space of such vectors is termed the *eigenspace*. If λ_i is not a repeated eigenvalue, then v_i is unique to within a scalar factor. When X is diagonal then $X = \mathrm{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$. Also, $\det(X) = \Pi_{i=1}^{n}\lambda_i$ so that $\det(X) = 0$, if and only if at least one eigenvalue is zero. As $\det(zI - XY) = \det(zI - YX)$, XY has the same nonzero eigenvalues as YX.

A symmetric, or Hermitian, matrix has only real eigenvalues, a skew symmetric, or skew-Hermitian, matrix has only imaginary eigenvalues, and an orthogonal, or unitary matrix has unity magnitude eigenvalues.

The *trace* of X, denoted $\mathrm{tr}(X)$, is the sum $\sum_{i=1}^{n} x_{ii} = \sum_{i=1}^{n} \lambda_i$. Notice that $\mathrm{tr}(X + Y) = \mathrm{tr}(X) + \mathrm{tr}(Y)$, and with XY square, then $\mathrm{tr}(XY) = \mathrm{tr}(YX)$. Also,

$\operatorname{tr}(X'X) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij}^2$ and $\operatorname{tr}^2(XY) \leq \operatorname{tr}(X'X)\operatorname{tr}(Y'Y)$. A useful identity is $\det(e^X) = e^{\operatorname{tr}(X)}$.

A.6 Similar Matrices

Two $n \times n$ matrices X, Y are called *similar* if there exists a nonsingular T such that $Y = T^{-1}XT$. Thus X is similar to X. Also, if X is similar to Y, then Y is similar to X. Moreover, if X is similar to Y and if Y is similar to Z, then X is similar to Z. Indeed, similarity of matrices is an *equivalence relation*. Similar matrices have the same eigenvalues.

If for a given X, there exists a *similarity transformation* T such that $\Lambda = T^{-1}XT$ is diagonal, then X is termed *diagonalizable* and $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ where λ_i are the eigenvalues of X. The columns of T are then the eigenvectors of X. All matrices with distinct eigenvalues are diagonalizable, as are orthogonal, symmetric, skew symmetric, unitary, Hermitian, and skew Hermitian matrices. In fact, if X is symmetric, it can be diagonalized by a real orthogonal matrix and when unitary, Hermitian, or skew-Hermitian, it can be diagonalized by a unitary matrix. If X is Hermitian and T is any invertible transformation, then Sylvester's *Inertia Theorem* asserts that T^*XT has the same number P of positive eigenvalues and the same number N of negative eigenvalues as X. The difference $S = P - N$ is called the *signature* of X, denoted $\operatorname{sig}(X)$.

A.7 Positive Definite Matrices and Matrix Decompositions

With $X = X'$ and real, then X is *positive definite (positive semidefinite or nonnegative definite)* if and only if the scalar $x'Xx > 0$ $(x'Xx \geq 0)$ for all nonzero vectors x. The notation $X > 0$ $(X \geq 0)$ is used. In fact $X > 0$ $(X \geq 0)$ if and only if all eigenvalues are positive (nonnegative). If $X = YY'$ then $X \geq 0$ and $YY' > 0$ if and only if Y is an $m \times n$ matrix with $m \leq n$ and $\operatorname{rk} Y = m$. If $Y = Y'$, so that $X = Y^2$, then Y is unique and is termed the symmetric square root of X, denoted $X^{1/2}$. If $X \geq 0$, then $X^{1/2}$ exists.

If Y is lower triangular with positive diagonal entries, and $YY' = X$, then Y is termed a *Cholesky factor* of X. A successive row by row generation of the nonzero entries of Y is termed a *Cholesky decomposition*. A subsequent step is to form $Y\Lambda Y' = X$ where Λ is diagonal positive definite, and Y is lower triangular with 1s on the diagonal. The above decomposition also applies to Hermitian matrices with the obvious generalizations.

For X a real $n \times n$ matrix, then there exists a *polar decomposition* $X = \Theta P$ where P is positive semidefinite symmetric and Θ is orthogonal satisfying $\Theta'\Theta = \Theta\Theta' = I_n$. While $P = (X'X)^{1/2}$ is uniquely determined, Θ is uniquely determined only if X is nonsingular.

The *singular values* of possibly complex rectangular matrices X, denoted $\sigma_i(X)$, are the positive square roots of the eigenvalues of X^*X. There exist unitary matrices U, V such that

$$V'XU = \left[\begin{array}{cccc} \sigma_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \sigma_n \\ \hline 0 & \dots & \dots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{array}\right] =: \Sigma.$$

If unitary U, V yield $V'XU$, a diagonal matrix with nonnegative entries, then the diagonal entries are the singular values of X. Also, $X = V\Sigma U'$ is termed a *singular value decomposition* (SVD) of X.

Every real $m \times n$ matrix A of rank r has a factorization $A = XY$ by real $m \times r$ and $r \times n$ matrices X and Y with $\operatorname{rk} X = \operatorname{rk} Y = r$. With $X \in \mathsf{R}^{m\times r}$ and $Y \in \mathsf{R}^{r\times n}$, then the pair (X, Y) belong to the product space $\mathsf{R}^{m\times r} \times \mathsf{R}^{r\times n}$. If $(X, Y), (X_1, Y_1) \in \mathsf{R}^{m\times r} \times \mathsf{R}^{r\times n}$ are two full rank factorizations of A, i.e. $A = XY = X_1Y_1$, then there exists a unique invertible $r \times r$ matrix T with $(X, Y) = (X_1T^{-1}, TY_1)$.

For X a real $n \times n$ matrix, the *QR decomposition* is $X = \Theta R$ where Θ is orthogonal and R is upper triangular (zero elements below the diagonal) with nonnegative entries on the diagonal. If X is invertible then Θ, R are uniquely determined.

A.8 Norms of Vectors and Matrices

The norm of a vector x, written $\|x\|$, is any positive valued function satisfying $\|x\| \geq 0$ for all x, with equality if and only if $x = 0$, $\|sx\| = |s|\,\|x\|$ for any scalar s, and $\|x + y\| \leq \|x\| + \|y\|$ for all x, y. The *Euclidean norm* or the 2-norm is $\|x\| = (\sum_{i=1}^n x_i^2)^{1/2}$, and satisfies the *Schwartz inequality* $\left|x'y\right| \leq \|x\|\,\|y\|$, with equality if and only if $y = sx$ for some scalar s. Other norms are $\|x\|_\infty = \max |x_i|$ and $\|x\|_1 = \sum_{i=1}^n |x_i|$.

The induced norm of a matrix X with respect to a given vector norm is defined as $\|X\| = \max_{\|x\|=1} \|Xx\|$. Corresponding to the Euclidean norm is the 2-norm $\|X\|_2 = \lambda_{\max}^{1/2}(X'X)$, being the largest singular value of X. Corresponding to the vector norms $\|x\|_\infty$, $\|x\|_1$ there are induced matrix norms $\|X\|_\infty =$

$\max_i \sum_{j=1}^{n} |x_{ij}|$ and $\|X\|_1 = \max_j \sum_{i=1}^{n} |x_{ij}|$. The *Frobenius norm* is $\|X\|_F = \operatorname{tr}^{1/2}(X'X)$. The subscript F is deleted when it is clear that the Frobenius norm is intended. For all induced norms and also for the Frobenius norm $\|Xx\| \leq \|X\|\,\|x\|$. Also, $\|X+Y\| \leq \|X\| + \|Y\|$ and $\|XY\| \leq \|X\|\,\|Y\|$. Note that $\operatorname{tr}(XY) \leq \|X\|_F\,\|Y\|_F$. The *condition number* of a nonsingular matrix X relative to a norm $\|\cdot\|$ is $\|X\|\,\|X^{-1}\|$.

A.9 Differentiation and Integration

Suppose X is a matrix valued function of the scalar variable t. Then $X(t)$ is called differentiable if each entry $x_{ij}(t)$ is differentiable. Also,

$$\frac{dX}{dt} = \left(\frac{dx_{ij}}{dt}\right), \qquad \frac{d}{dt}(XY) = \frac{dX}{dt}Y + X\frac{dY}{dt}, \qquad \frac{d}{dt}e^{tX} = Xe^{tX} = e^{tX}X.$$

Also, $\int X dt = (\int x_{ij} dt)$. Now with ϕ a scalar function of a matrix X, then

$$\frac{\partial \phi}{\partial X} = \text{ the matrix with } ij\text{th entry } \frac{\partial \phi}{\partial x_{ij}}.$$

If Φ is a matrix function of a matrix X, then

$$\frac{\partial \Phi}{\partial X} = \text{ a block matrix with } ij\text{th block } \frac{\partial \phi_{ij}}{\partial X}.$$

The case when X, Φ are vectors is just a specialization of the above definitions. If X is square $(n \times n)$ and nonsingular, $(\partial/\partial X)(\operatorname{tr}(WX^{-1})) = -X^{-1}WX^{-1}$. Also $\log\det(X) \leq \operatorname{tr} X - n$ and with equality if and only if $X = I_n$. Furthermore, if X is a function of time, then $(d/dt)X^{-1}(t) = -X^{-1}(dX/dt)X^{-1}$, which follows from differentiating $XX^{-1} = I$.

If $P = P'$, $(\partial/\partial x)(x'Px) = 2Px$.

A.10 Lemma of Lyapunov

If A, B, C are known $n \times n, m \times m$ and $n \times m$ matrices, then the linear equation $AX + XB + C = 0$, has a unique solution for an $n \times m$ matrix X if and only if $\lambda_i(A) + \lambda_j(B) \neq 0$ for any i and j. In fact $[I \otimes A + B' \otimes I]\operatorname{vec}(X) = -\operatorname{vec}(C)$ and the eigenvalues of $[I \otimes A + B' \otimes I]$ are precisely given by $\lambda_i(A) + \lambda_j(B)$. Here $\operatorname{vec}(X)$ stands for the column vector obtained from the matrix X by stacking the columns of X from left to right under one another in the vector $\operatorname{vec}(X)$. If $C > 0$ and $A = B'$, the *Lemma of Lyapunov* for $AX + XB + C = 0$ states that $X = X' > 0$ if and only if all eigenvalues of B have negative real parts.

The linear equation $X - AXB = C$, or equivalently, $[I_{n^2} - B' \otimes A]\operatorname{vec}(X) = \operatorname{vec}(C)$ has a unique solution if and only if $\lambda_i(A)\lambda_j(B) \neq 1$ for any i, j. If $A = B'$

and $|\lambda_i(A)| < 1$ for all i, then for $X - AXB = C$, the *Lemma of Lyapunov* states that $X = X' > 0$ for all $C = C' > 0$.

Actually, the condition $C > 0$ in the lemma can be relaxed to requiring for any D such that $DD' = C$ that (A, D) be completely controllable, or (D, A) be completely detectable, see definitions Appendix B.

A.11 Vector Spaces and Subspaces

Let us restrict to the real field R (or complex field C), and recall the spaces R^n (or C^n). These are in fact special cases of vector spaces over R (or C) with the vector additions and scalar multiplications properties for its elements spelled out in Section A.2. They are denoted *real (or complex) vector spaces*. Any space over an arbitrary field K which has the same properties is in fact a *vector* space V. For example, the set of all $m \times n$ matrices with entries in the field as R (or C), is a vector space. This space is denoted by $\mathsf{R}^{m\times n}$ (or $\mathsf{C}^{m\times n}$).

A *subspace* W of V is a vector space which is a subset of the vector space V. The set of all linear combinations of vectors from a nonempty subset S of V, denoted $L(S)$, is a subspace (the smallest such) of V containing S. The space $L(S)$ is termed the subspace *spanned* or *generated* by S. With the empty set denoted ϕ, then $L(\phi) = \{0\}$. The rows (columns) of a matrix X viewed as row (column) vectors span what is termed the row (column) space of X denoted here by $[X]_r$ ($[X]_c$). Of course, $\square(X') = [X]_r$ and $\square(X) = [X]_c$.

The orthogonal complement of a subspace W of V is denoted as $W^\perp$. It is a subspace of V, consisting of all vectors $v \in V$ such that $v'w = 0$ for all $w \in W$. For a matrix X we have $\square(X')^\perp = \ker(X)$. For two subspaces $W, U \in V$ we denote by $W \oplus U$ the space spanned by any combination of W and U, i.e. $z \in W \oplus U$ implies that $z = w + u$ for some $w \in W$ and $u \in U$. For a square matrix X, dimension $n \times n$ we have that $\ker(X)^\perp \oplus \square(X) = \mathsf{R}^n$.

A.12 Basis and Dimension

A vector space V is *n-dimensional* ($\dim V = n$) if there exists linearly independent vectors, the *basis vectors*, $\{e_1, e_2, \ldots, e_n\}$ which span V. A basis for a vector space is nonunique, yet every basis of V has the same number n of elements. A subspace W of V has the property $\dim W \leq n$, and if $\dim W = n$, then $W = V$. The dimension of the row (column) space of a matrix X is the *row (column) rank* of X. The row and column ranks are equal and are in fact the rank of X. The coordinates of a vector x in V with respect to a basis are the (unique) tuple of coefficients of a linear combination of the basis vectors that generate x. Thus with $x = \sum_i a_i e_i$, then $a_1, a_2, \ldots, a_n$ are the coordinates.

For a square $n \times n$ matrix X over R we have that $\dim\ker(X) + \dim\square(X) = n$.

A.13 Mappings and Linear Mappings

For A, B arbitrary sets, suppose that for each $a \in A$ there is assigned a single element $f(a)$ of B. The collection f of such is called a *function, or map* and is denoted $f : A \to B$. The *domain* of the mapping is A, the *codomain* is B. For subsets A_s, B_s, of A, B then $f(A_s) = \{f(a) : a \in A_s\}$ is the *image* of A_s, and $f^{-1}(B_s) = \{a \in A : f(a) \in B_s\}$ is the *preimage or fiber* of B_s. If $B_s = \{b\}$ is a singleton set we also write $f^{-1}(b)$ instead of $f^{-1}(\{b\})$. Also, $f(A)$ is the *image* or *range* of f. The notation $x \mapsto f(x)$ is used to denote the image $f(x)$ of an arbitrary element $x \in A$.

The composition of mappings $f : A \to B$ and $g : B \to C$, denoted gof, is an associative operation. The identity map $\mathrm{id}_A : A \to A$ is the map defined by $a \mapsto a$ for all $a \in A$.

A mapping $f : A \to B$ is *one-to-one* or *injective* if different elements of A have distinct images, i.e. if $a_1 \neq a_2 \Rightarrow f(a_1) \neq f(a_2)$. The mapping is *onto* or *surjective* if every $b \in B$ is the image of at least one $a \in A$. A *bijective* mapping is one-to-one and onto (surjective and injective). If $f : A \to B$ and $g : B \to A$ are maps with $g \circ f = \mathrm{id}_A$, then f is injective and g is surjective.

For vector spaces V, W over R or C (denoted K) a mapping $F : V \to W$ is a *linear mapping* if $F(v + w) = F(v) + F(w)$ for any $v, w \in V$, and as $F(kv) = kF(v)$ for any $k \in \mathsf{K}$ and any $v \in V$. Of course $F(0) = 0$. A linear mapping is called an *isomorphism* if it is bijective. The vector spaces V, W are *isomorphic* if there is an isomorphism of V onto W. A linear mapping $F : V \to U$ is called *singular* if it is not an isomorphism.

For $F : V \to U$, a linear mapping, the *image* of F is the set $\mathrm{Im}(F) = \{u \in U \mid F(v) = u$ for some $v \in V\}$. The *kernel* of F is $\ker(F) = \{u \in V \mid F(u) = 0\}$. In fact for finite dimensional spaces $\dim V = \dim \ker(F) + \dim \mathrm{Im}(F)$.

Linear operators or transformations are linear mappings $T : V \to V$, i.e. from V to itself.

The *dual vector space* V^* of a K-vector space V is defined as the K-vector space of all K-linear maps $\lambda : V \to \mathsf{K}$. It is denoted by $V^* = \mathrm{Hom}(V, \mathsf{K})$.

APPENDIX **B**

Dynamical Systems

This appendix summarizes the key results of both linear and nonlinear dynamical systems theory required as a background in this text. For more complete treatments, see Irwin (1980), Isidori (1989), Kailath (1980) and Sontag (1990).

B.1 Linear Dynamical Systems

State equations

Discrete-time linear, finite-dimensional, dynamical systems for $k = k_0, k_0+1, \ldots$ with initial state $x_{k_0} \in \mathsf{R}^n$ are described by

$$x_{k+1} = A_k x_k + B_k u_k, \qquad y_k = C_k x_k + D_k u_k, \tag{1.1}$$

where $x_k \in \mathsf{R}^n,\ u_k \in \mathsf{R}^m,\ y_k \in \mathsf{R}^p$ and A_k, B_k, C_k, D_k are matrices of appropriate dimension, possibly time varying. The solution for $k > k_0$ is,

$$x_k = \Phi_{k,k_0} x_{k_0} + \sum_{i=k_0}^{k-1} \Phi_{k-1,i} B_i u_i .$$

where $\Phi_{k_0,k_0} = I;\ \Phi_{k+1,k_0} = A_k \Phi_{k,k_0}$ or $\Phi_{k,k_0} = A_{k-1} \ldots A_{k_0}$ for $k > k_0$. In the time invariant case $\Phi_{k,k_0} = A^{k-k_0},\ k \geq k_0$.

A discrete-time, time-invariant system (1.1) is called *stable*, if the eigenvalues of A are all located in the open unit disc $\{z \in \mathsf{C} \mid |z| < 1\}$. This implies $\lim_{k\to\infty} A^k = 0$.

Transfer functions

The *Z-transform* of a sequence $\{h_k \mid k \in \mathsf{N}_0\}$ is the formal power series in z^{-1}

$$H(z) = \sum_{k=0}^{\infty} h_k z^{-k}.$$

In discrete-time, the *Z-transform* yields the *transfer function* for the system (1.1)

$$H(z) = C(zI - A)^{-1}B + D,$$

in terms of the Z-transform variable z. For $x_0 = 0$, then $Y(z) = (C(zI - A)^{-1}B + D)U(z)$ expresses the relation between the Z-transforms of the sequences $\{u_k\}$ and $\{y_k\}$. The transfer function z^{-1} corresponds to a unit delay.

A (matrix) transfer function for the system (1.1) with a state space realization given via the matrices (A, B, C, D) is represented as

$$H : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right].$$

Continuous-time linear systems

In continuous time we have

$$\begin{aligned} \dot{x} := \frac{dx}{dt} &= A(t)x + B(t)u, \\ y &= C(t)x + D(t)u. \end{aligned} \tag{1.2}$$

The transition matrix $\Phi(t, t_0)$ satisfies $\Phi(t_0, t_0) = I$ and $\dot{\Phi}(t, t_0) = A(t)\Phi(t, t_0)$. It has the semigroup property that $\Phi(t_2, t_1)\Phi(t_1, t_0) = \Phi(t_2, t_0)$. The solution of (1.2) starting at time t_0 in x_0 is given by:

$$x(t) = \Phi(t, t_0)x_0 + \int_{t_0}^{t} \Phi(t, \tau)B(\tau)u(\tau)d\tau.$$

In the time invariant case $A(t) = A$, we have $\Phi(t, t_0) = e^{A(t-t_0)}$. A time invariant linear system (1.2) is called stable if the real part of the eigenvalues of A are negative. This implies $\lim_{t\to\infty} e^{At} = 0$.

Controllability and Stabilizability

In the time-invariant, continuous-time, case, the pair (A, B) with $A \in \mathrm{R}^{n\times n}$ and $B \in \mathrm{R}^{n\times m}$ is termed *completely controllable* (or more simply *controllable*) if one of the following equivalent conditions holds:

- There exists a control u taking $\dot{x} = Ax + Bu$ from arbitrary state x_0 to another arbitrary state x_1, in finite time.
- $\operatorname{rank}(B, AB, \ldots, A^{n-1}B) = n$.
- $(\lambda I - A, B)$ has full rank for all (complex) λ.
- $W_c(T) = \int_0^T e^{tA}BB'e^{tA'}dt > 0$ for all $T > 0$.

- $AX = XA$ and $XB = 0$ implies $X = 0$.
- $w'e^{tA}B = 0$ for all t implies $w = 0$.
- $w'A^iB = 0$ for all i implies $w = 0$.
- There exists a K of appropriate dimension such that $A + BK$ has arbitrary eigenvalues.
- There exists no coordinate basis change such that

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \qquad B = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}.$$

The symmetric matrix $W_c(T) = \int_0^T e^{tA}BB'e^{tA'}dt > 0$ is the *controllability Gramian*, associated with $\dot{x} = Ax + Bu$. It can be found as the solution at time T of $\dot{W}_c(t) = AW_c(t) + W_c(t)A' + BB'$ initialized by $W_c(0) = 0$. If A has only eigenvalues with negative real part, in short $\operatorname{Re}\lambda(A) < 0$, then $W_c(\infty) = \lim_{t\to\infty} W_c(t)$ exists.

In the time-varying case, only the first definition of controllability applies. It is equivalent to requiring that the Gramian

$$W_c(t, T) = \int_t^{t+T} \Phi(t + T, \tau)B(\tau)B'(\tau)\Phi'(t + T, \tau)d\tau$$

be positive definite for all t and some T. The concept of *uniform complete controllability* requires that $W_c(t, T)$ be uniformly bounded above and below from zero for all t and some $T > 0$. This condition ensures that a bounded energy control can take an arbitrary state vector x to zero in an interval $[t, t + T]$ for arbitrary t. A uniformly controllable system has the property that a bounded $K(t)$ exists such that $\dot{x} = (A(t) + B(t)K(t))x$ has an arbitrary degree of (exponential) stability.

The discrete-time controllability conditions, Gramians, etcetera are analogous to the continuous-time definitions and results. In particular, the N-controllability Gramian of a discrete-time system is defined by

$$W_c^{(N)} := \sum_{k=0}^{N} A^k BB'(A')^k,$$

for $N \in \mathsf{N}$. The pair (A, B) is controllable if and only if $W_c^{(N)}$ is positive definite for all $N \geq n-1$. If A has all its eigenvalues in the open unit disc $\{z \in \mathsf{C} \mid |z| < 1\}$, in short $|\lambda(A)| < 1$, then

$$W_c := \sum_{k=0}^{\infty} A^k BB'(A')^k$$

exists and is positive definite if and only if (A, B) is controllable.

Observability and Detectability

The pair (A, C) has observability/detectability properties according to the controllability/stabilizability properties of the pair (A', C'); likewise for the time-varying and uniform observability cases. The observability Gramians are known as duals of the controllability Gramians, e.g. in the continuous-time case

$$W_o(T) := \int_o^T e^{tA'} C' C e^{tA} dt, \qquad W_o := \int_0^\infty e^{tA'} C' C e^{tA} dt,$$

and in the discrete-time case,

$$W_o^{(N)} := \sum_{k=0}^{N} (A')^k C' C A^k, \qquad W_o := \sum_{k=0}^{\infty} (A')^k C' C A^k.$$

Minimality

The state space systems of (1.1) and (1.2) denoted by the triple (A, B, C) are termed *minimal realizations*, in the time-invariant case, when (A, B) is completely controllable and (A, C) is completely observable. The *McMillan degree* of the transfer functions $H(s) = C(sI - A)^{-1}B$ or $H(z) = C(zI - A)^{-1}B$ is the state space dimension of a minimal realization. *Kalman's Realization Theorem* asserts that any $p \times m$ rational matrix function $H(s)$ with $H(\infty) = 0$ (that is $H(s)$ is *strictly proper*) has a minimal realization (A, B, C) such that $H(s) = C\,(sI - A)^{-1}\,B$ holds. Moreover, given two minimal realizations denoted (A_1, B_1, C_1) and (A_2, B_2, C_2). then there always exists a unique nonsingular transformation matrix T such that $TA_1T^{-1} = A_2,\ TB_1 = B_2,\ C_1T^{-1} = C_2$. All minimal realizations of a transfer function have the same dimension.

Balanced Realizations

For a stable system (A, B, C), a realization in which the Gramians are equal and diagonal as

$$W_c = W_o = \operatorname{diag}(\sigma_1, \dots, \sigma_n)$$

is termed a *diagonally balanced* realization. For a minimal realization (A, B, C), the *singular values* σ_i are all positive. For a nonminimal realization of McMillan degree $m < n$, then $\sigma_{m+i} = 0$ for $i > 0$. Corresponding definitions and results apply for Gramians defined on finite intervals T. Also, when the controllability and observability Gramians are equal but not necessarily diagonal, the realizations are termed *balanced*. Such realizations are unique only to within orthogonal basis transformations.

Balanced truncation is where a system (A, B, C) with $A \in \mathrm{R}^{n\times n},\ B \in \mathrm{R}^{n\times m}$, $C \in \mathrm{R}^{p\times n}$ is approximated by an rth order system with $r < n$ as follows: Assuming an ordering $\sigma_i > \sigma_{i+1}$, for all i the last $(n - r)$ rows of (A, B) and last

$(n-r)$ columns of $\left[\begin{smallmatrix} A \\ C \end{smallmatrix}\right]$ of a balanced realization are set to zero to form a reduced rth order realization $(A_r, B_r, C_r) \in \mathsf{R}^{r\times r} \times \mathsf{R}^{r\times m} \times \mathsf{R}^{p\times r}$. A theorem of Pernebo and Silverman states that if (A, B, C) is balanced and minimal, and $\sigma_r > \sigma_{r+1}$, then the reduced rth order realization (A_r, B_r, C_r) is also balanced and minimal.

B.2 Norms, Spaces and Stability Concepts

The key stability concept for our purposes is that of bounded-input stability *(BIBO stability)*. In the case of linear systems this is equivalent to *asymptotic stability* of the state space descriptions.

To be precise, and focusing on discrete-time signals and systems, we work first with *signals* which are simply functions which map the integers Z to R^n. The set of signals is $S = \{f : \mathsf{Z} \to \mathsf{R}^n\}$. The size of a signal is measured by some norm. Let us consider a 2-norm over an infinite horizon as

$$\| f \|_2 = \left(\sum_{-\infty}^{\infty} \| f_k \|^2 \right)^{1/2},$$

where $\|x\| = \sqrt{x'x}$ is the Euclidean norm.

The *Lebesque 2-space* is defined by

$$\ell_2(-\infty, \infty) = \left\{ f \in S : \| f \|_2 < \infty \right\}.$$

Variations $\ell_2(0, \infty)$, $\ell_2(-\infty, 0)$ are defined similarly. When the time interval is understood, then the space is referred to as the ℓ_2 space.

The space ℓ_2 is a *Hilbert space* with inner product

$$\langle f, g \rangle = \sum_{-\infty}^{\infty} g_k' f_k,$$

satisfying $|\langle f, g\rangle| \le \| f \|_2 \, \|h\|_2$ and $\| f \|_2^2 = \langle f, f \rangle$.

In the Z-domain, the 2-norm is, which follows from Parseval's Theorem

$$\| f(z) \|_2 = \left\{ \sup_{\epsilon > 0} \frac{1}{2\pi} \oint_{|z|=1+\epsilon} f^-(z) f(z) \frac{dz}{z} \right\}^{1/2},$$

where $f^-(z) = f'(z^{-1})$. The *Hardy 2-space* H_2 is defined as

$$H_2 = \left\{ f(z) \text{ is analytic in } |z| > 1 \quad \text{and} \quad \| f \|_2 < \infty \right\}.$$

For any $f \in H_2$, the boundary function

$$f_b(z)_{|z|=1} = \lim_{\epsilon \to 0} f(z)|_{|z|=1+\epsilon}$$

exists for almost all $|z| = 1$ (*Fatou's theorem*) and $\| f_b \|_2 = \| f \|_2$ so that

$$\| f \|_2 = \left\{ \frac{1}{2\pi} \oint_{|z|=1} f_b^-(z)\, f_b(z) \frac{dz}{z} \right\}^{1/2} .$$

The mapping $f \in H_2$ to $f_b \in \ell_2$ is linear, injective and norm preserving.

Consider now linear time-invariant systems in discrete-time with matrix transfer function $H(z) : \ell_2 \mapsto \ell_2$.

The space ℓ_∞ is defined from

$$\ell_\infty = \{ H : \|H\|_\infty < \infty \} ,$$

where the ℓ_∞ norm is defined from

$$\|H\|_\infty = \sup_{|z|=1} \sigma_{\max} (H(z)) .$$

Here $\sigma_{\max}(\cdot)$ denotes the maximum singular value.

Stable Systems

Stable systems $H(z)$ are such that $H(z)w(z) \in H_2$ whenever $w(z) \in H_2$. A necessary condition is that $H(z)$ is analytic in $|z| > 1$.

The H_∞ space is defined from

$$H_\infty = \left\{ H(z) : H(z) \text{ is analytic in } |z| > 1 \quad \text{and} \quad \|H\|_{2-gn} < \infty \right\} ,$$

where the norm is an H_∞ norm

$$\|H\|_{2-gn} = \sup_{\epsilon > 0} \sup_{|z|=1+\epsilon} \sigma_{\max} (H(z)) ,$$

being but a mild version of the ℓ_∞ norm, frequently termed an H_∞ norm and written $\|H\|_\infty$.

Now $H(z)$ defines a stable system if and only if $H(z) \in H_\infty$.

In the case that $H(z) \in H_\infty$ is rational then we use the notation $H(z) \in RH_\infty$. $H(z) \in RH_\infty$ if and only if $H(z)$ has no pole in $|z| \geq 1$.

Corresponding definitions and results apply as before for continuous-time signals and systems with $\sum_{-\infty}^{\infty}$ and $\oint_{|z|=1}$ is replaced by $\int_{-\infty}^{\infty}$ and $\int_{s=j\omega}$. Also f^- replaced by f^* where $f^*(s) = f'(s^*)$ where s^* is the conjugate of s. There are additional technical issues concerning signals which differ only on (Lebesque) measure zero. Just to state explicitly one result, we have that $H(s) \in RH_\infty$ if and only if $H(s)$ has no pole in $Re(s) \geq 0$.

B.3 Nonlinear Systems Stability

We first summarize the basic results from stability theory for ordinary differential equations on R^n, and subsequently consider corresponding results for difference equations.

Consider

$$\dot{x} = f(x), \qquad x \in \mathrm{R}^n. \tag{3.1}$$

Let f be a smooth vector field on R^n. We assume that $f(0) = 0$, so that $x = 0$ is an equilibrium point of (3.1). Let $D \subset \mathrm{R}^n$ be a compact neighborhood of 0 in R^n.

A *Lyapunov function* of (3.1) on D is a smooth function $V : D \to \mathrm{R}$ having the properties

1. $V(0) = 0$, $V(x) > 0$ for all $x \neq 0$ in D.
2. For any solution $x(t)$ of (3.1) with $x(0) \in D$,

$$\dot{V}(x(t)) = \frac{d}{dt} V(x(t)) \leq 0. \tag{3.2}$$

Also, $V : D \to \mathrm{R}$ is called a *strict Lyapunov function* if the strict inequality holds

$$\dot{V}(x(t)) = \frac{d}{dt} V(x(t)) < 0 \quad \text{for } x(t) \in D - \{0\}. \tag{3.3}$$

Theorem 3.1 (Stability). *If there exists a Lyapunov function $V : D \to \mathrm{R}$ defined on some compact neighborhood of $0 \in \mathrm{R}^n$, then $x = 0$ is a stable equilibrium point.*

Theorem 3.2 (Asymptotic Stability). *If there exists a strict Lyapunov function $V : D \to \mathrm{R}$ defined on some compact neighborhood of $0 \in \mathrm{R}^n$, then $x = 0$ is an asymptotically stable equilibrium point.*

Theorem 3.3 (Global Asymptotic Stability). *If there exists a proper map $V : \mathrm{R}^n \to \mathrm{R}$ which is a strict Lyapunov function with $D = \mathrm{R}^n$, then $x = 0$ is globally asymptotically stable.*

Here properness of $V : \mathrm{R}^n \to \mathrm{R}$ is equivalent to $V(x) \to \infty$ for $\|x\| \to \infty$.

Theorem 3.4 (Exponential Asymptotic Stability). *If in Theorem 3.2 one has $\alpha_1 \|x\|^2 \leq V(x) \leq \alpha_2 \|x\|^2$ and $-\alpha_3 \|x\|^2 \leq \dot{V}(x) \leq -\alpha_4 \|x\|^2$ for some positive α_i, $i = 1, \ldots, 4$, then $x = 0$ is exponentially asymptotically stable.*

Consider nonlinear systems with external inputs u as

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

then BIBO stability is as for linear systems namely, that bounded inputs lead to bounded signals. We refer to this stability as ℓ_∞ BIBO stability.

Similarly for discrete time systems

$$x_{k+1} = f(x_k); \qquad x \in \mathrm{R}^n. \tag{3.4}$$

Here f is a continuous map of R^n to R^n. Assume that $f(0) = 0$, so that $x = 0$ is a trivial solution of (3.4). A solution of (3.4) starting in x_0 is denoted by $x_k(x_0)$.

Let D be a compact neighborhood of $f(0)$ in R^n. A Lyapunov function $V : D \to \mathrm{R}^+$ is a continuous map such that

1. $V(0) = 0,\; V(x) > 0$ for $x \neq 0$.
2. for any solution $x_k(x_0)$ of (3.4) with $x_0 \in D$
$$V\left(x_{k+1}(x_0)\right) \leq V\left(x_k(x_0)\right).$$

V is a strict Lyapunov function if

3. for any solution $x_k(x_0)$ of (3.4) with $x_o \in D$
$$V\left(x_{k+1}(x_0)\right) < V\left(x_k(x_0)\right); \quad \text{if } x_k(x_0) \neq 0.$$

The above Theorems 3.1, 3.2 and 3.3, for stability, asymptotic stability and global asymptotic stability, then also hold for the system (3.4).

The trivial solution of (3.4) is called exponentially stable provided its linearization is stable. The linearization of (3.4) is given by

$$z_{k+1} = Df(0)z_k,$$

where $Df(0)$ is the Jacobian of f evaluated at 0. The Jacobian is defined as the matrix of partial derivatives:

$$Df(z) =: \left(Df_{ij}(z)\right) = \left(\frac{\partial f_i}{\partial z_j}(z)\right).$$

APPENDIX C

Averaging Analysis For Adaptive Systems

C.1 Introduction

We present here in a nutshell some ideas from *averaging analysis* which is a powerful technique to study systems whose dynamics split naturally over different *time scales*. No proofs are provided, we refer the reader to, e.g. Mareels and Polderman (1996).

C.2 Averaging

Averaging in its basic form is concerned with systems of the form:

$$x_{k+1} = x_k + \mu f_k(x_k); \qquad x_0; \qquad k = 0, 1, \ldots . \tag{2.1}$$

The parameter μ is a small positive constant that characterizes the *time scale separation* between the variation of the state variable x over time and the time variations in the driving term $f_k(\cdot)$. With time scale separation we mean the following. Assume for the moment that $\| f_k(x)\| \leq F$. On a time interval of length N, a solution of (2.1), say x, can at most change by an amount $\|x_k - x_\ell\| \leq \mu N F$, for $|k - \ell| \leq N$. On the same time interval $\| f_k(x_k) - f_l(x_\ell)\| \leq 2F$. The ratio of change between x and $f_k(\cdot)$ is therefore of magnitude μ; the time variations of the state x being (potentially) μ times slower than the time variations in f. It is this time scale separation that hints at replacing the time-varying $f_k(\cdot)$ by the time invariant *averaged* driving term

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} f_k(z) = f^a(z), \tag{2.2}$$

provided of course that the latter exists.

In *adaptive systems* the time scale separation is less explicit than portrayed by (2.1). Part of the analysis will be precisely concerned with transforming an adaptive system into the above format (2.1), at least approximately, such that standard averaging results can be applied.

More precisely, we consider adaptive systems of the general form:

$$\begin{aligned} x_{k+1} &= A(\theta_k)x_k + B_k(\theta_k), \qquad && x_0, \\ \theta_{k+1} &= \theta_k + \mu g_k(\theta_k, x_k), \qquad && \theta_0. \end{aligned} \tag{2.3}$$

The adapted parameter vector is θ_k. The positive parameter μ scales the adaptation gain. We assume μ to be small, it expresses explicitly that the adaptation mechanism progresses slowly. The rest of the state vector x_k contains mixed time scale behavior. Partly it contains the fast time variations due to the driving functions $B_k(\cdot)$, partly it contains the effect of the slowly varying θ_k via the functions $A(\theta)$ and $B(\theta)$. The time variations in B are typically due to external signals, such as reference signals, disturbances and or plant variations. It will be shown that under very mild assumptions the *zero adaptation* situation can be used as an approximation for the *slow adaptation* case. This in turn will enable standard averaging techniques to be used to analyze the behavior of the adaptive system.

The methodology we are about to discuss, consisting of a zero adaptation approximation followed by an averaging analysis, is applicable to a large class of adaptive systems operating under a wide variety of assumptions, not necessarily requiring that the model class encompasses the actual plant dynamics.

We now introduce and discuss the basic averaging technique.

Some Notation and Preliminaries

In order not to overload the expressions we introduce some notation and definitions. We often need to estimate functional dependence on μ. This is done via so-called *order functions* (Sanders and Verhulst, 1985):

Definition. A scalar valued function $\delta(\mu)$ is called an *order function* if it is positive valued and continuous on an interval $(0, \mu^*)$ for some $\mu^* > 0$ and $\lim_{\mu\to 0} \delta(\mu)$ exists, perhaps ∞.

Order functions can be defined in a more general sense. However, as we mainly need to compare functions in terms of orders of μ and are only interested in small μ, the above, more restrictive definition suffices.

Example. The terms μ, $\sin(\mu)$, $\sqrt{\mu}$ and $1/\mu$ are order functions. The function $\sin(1/\mu) + 1$ is not an order function.

The *size or order* of order functions can be compared as follows:

Definition. Let $\delta_1(\mu)$ and $\delta_2(\mu)$ be two order functions. Then $\delta_1(\mu)$ is said to be of *order* of $\delta_2(\mu)$, denoted as $\delta_1(\mu) = O(\delta_2(\mu))$, if there exists positive constants

μ^* and C such that

$$\delta_1(\mu) \leq C\delta_2(\mu) \quad \text{for all } \mu \in [0, \mu^*). \tag{2.4}$$

If $\delta_1(\mu) = O(\delta_2(\mu))$ and $\delta_2(\mu) = O(\delta_1(\mu))$ then we say that δ_1 and δ_2 are equivalent order functions.

Definition. Let $\delta_1(\mu)$ and $\delta_2(\mu)$ be two order functions. $\delta_1(\mu)$ is said to be of *small order* of $\delta_2(\mu)$, denoted as $\delta_1(\mu) = o(\delta_2(\mu))$ if

$$\lim_{\mu \to 0} \frac{\delta_1(\mu)}{\delta_2(\mu)} = 0. \tag{2.5}$$

Example. Now μ is $o(1)$, as indeed $\lim_{\mu\to 0} \mu = 0$, and obviously $\mu \leq 1$ for all $\mu \in [0, 1]$. However, $\sin(\mu)$ is $O(\mu)$ on $\mu \in [0, \pi)$. Also μ is $O(\sin(\mu))$ on $\mu \in [0, \pi/2)$. Hence μ and $\sin(\mu)$ are equivalent order functions.

Functions that do not only depend on μ can also be compared with order functions, using the following conventions:

Definition. Let $f : \mathsf{R}^+ \times \mathsf{N} \to \mathsf{R}^n$, $(\mu, k) \to f_k(\mu)$ be continuous in μ. Let $\delta(\mu)$ be an order function. We say that f is of order δ, denoted $f_k(\mu) = O(\delta(\mu))$, if there exist positive constants μ^* and C such that

$$\| f_k(\mu)\| \leq C\delta(\mu) \quad \text{for all } k \quad \text{and} \quad \mu \in [0, \mu^*). \tag{2.6}$$

Definition. Let $f : \mathsf{R}^+ \times \mathsf{N} \times \mathsf{R}^n \to \mathsf{R}^n$, $(\mu, k, x) \to f(\mu, k, x)$ be uniformly (in k) continuous in μ, x on a set $[0, \mu_c) \times D$ [*]. We say that f is order δ for some order function $\delta(\mu)$ if there exist a compact domain $D' \subset D$ and positive constants $\mu^* \leq \mu_c$ and C such that

$$\| f(\mu, k, x)\| < C\delta(\mu) \quad \text{for all } k, \qquad \mu \in [0, \mu^*), \qquad x \in D'. \tag{2.7}$$

Example. Now $\mu \sin(k) = O(\mu)$ and $\sqrt{1 + \mu x} - 1$ is also $O(\mu)$. Indeed for all $|x| \leq 1$ and $\mu \in [0, 1]$, one has $\sqrt{1 + \mu x} - 1 \leq 0.5\mu$.

We will want to estimate approximation errors over a particular time interval that increases as μ becomes smaller. In this respect the following convention is useful:

Definition. Let $f : \mathsf{R}^+ \times \mathsf{N} \times \mathsf{R}^n \to \mathsf{R}^n$, $(\mu, k, x) \to f(\mu, k, x)$ be uniformly (in k) continuous in μ, x on a domain $[0, \mu_c) \times D$. Let $\delta_1(\mu)$ and $\delta_2(\mu)$ be two order functions. We say that $f(\mu, k, x)$ is of order $\delta_1(\mu)$ on *a time scale* $\delta_2(\mu)$ on the set $D \subset \mathsf{R}^n$ provided that for any integer L there exist positive constants μ^* and C such that for all $\mu \in [0, \mu^*)$ and for all $x \in D$

$$\| f(\mu, k, x)\| < C\delta_1(\mu) \quad \text{for all } k \in [0, L\delta_2(\mu)]. \tag{2.8}$$

[*] $g(x, k)$ is uniformly (in k) continuous in x on a domain D if for all $\epsilon > 0$ there is a $\delta > 0$ such that for all k and all $x, y \in D$, we have that $|x - y| < \delta$ implies that $|g(x, k) - g(y, k)| < \epsilon$.

Example. Let $k \in \mathsf{N}$, $|x| < 1$ and $\mu \in [0, 1)$.

$$\begin{aligned}
x^3 \sin(\mu k) &= O(\mu) && \text{on time scale } O(1), \\
x^3 \sin(\mu k) &= O(1) && \text{on time scale } O\left(\frac{1}{\mu}\right), \\
\mu k &= O(\sqrt{\mu}) && \text{on time scale } O\left(\frac{1}{\sqrt{\mu}}\right), \\
(1+\mu x)^k &= O(1) && \text{on time scale } O\left(\frac{1}{\mu}\right).
\end{aligned} \tag{2.9}$$

The last statement can be derived under the given restrictions for μ and x from the following inequalities: $0 < 1 + \mu x < 1 + \mu < e^{\mu}$ and thus $(1+\mu)^k \leq e^{\mu k}$; considering the time interval $k \in [0, 1/\mu)$ we get $(1+\mu x)^k \leq e$.

When discussing solutions of a difference equation such as $x_{k+1} = x_k + \mu f_k(x_k)$ a solution is denoted by $x(k, x_0, k_0, \mu)$ to indicate a solution that at time k_0 equals x_0. The parameter μ is included in the argument to make explicit the dependence on this parameter. Where no confusion can arise the shorthand x_k will be used, and if the function f is time invariant the notation $x(k, x_0, \mu)$ or x_k used.

Finite Horizon Averaging Result

We are now in a position to formulate an approximation result valid on a timescale $1/\mu$. The result is stated under weak regularity conditions and in a format to facilitate its application to adaptive systems.

Consider the following nonlinear and time dependent difference equation in *standard* form:

$$x_{k+1} = x_k + \mu f_k(x_k); \qquad k \in \mathsf{N}, \qquad x(0) = x_0. \tag{2.10}$$

The parameter μ is to be thought of as a small positive constant.

We want to approximate the solution $x_k(x_0, k_0, \mu)$ of (2.10) by some $x_k^a(x_0, \mu)$ solution of $x_{k+1}^a = x_k^a + \mu f^a(x_k^a)$, where f^a is a suitable average of f. The approximation error, $x - x^a$, should be $o(1)$ on a time scale $1/\mu$.

The following regularity properties are assumed:

Assumption 2.1. *Consider the difference equation* (2.10), $f : \mathsf{N} \times \mathsf{R}^n \to \mathsf{R}^n$ *is locally bounded and* locally Lipschitz continuous *in x uniformly in k. That is, for each compact subset $D \subset \mathsf{R}^n$, there exist positive constants F_D and λ_D possibly dependent on D, but independent of k, leading to the following three properties;*

1. *f is locally uniformly bounded*

$$F_D > 0 : \quad \text{for all } x \in D, \quad \text{for all } k : \| f_k(x) \| \leq F_D. \tag{2.11}$$

2. *f is locally uniformly Lipschitz continuous*

$$\lambda_D : \quad \text{for all } x, y \in D, \quad \text{for all } k : \| f_k(x) - f_k(y) \| \leq \lambda_D \|x - y\| . \tag{2.12}$$

3. *f has a* well defined average, *denoted f^a, in that for all $x \in D$ the following limit exists:*

$$f^a(x) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} f_k(x). \tag{2.13}$$

Before continuing with our main result we offer the following observations on the notion of average.

Remarks.

1. The average f^a is also Lipschitz continuous with the same Lipschitz constant λ_D and locally bounded with constant F_D in the domain D as f.

   Often the average f^a will have better continuity properties than the f from which it is derived. This may be illustrated with $f_k(x) = \text{sign}(\sin(k) + x)$; f is not continuous but $f^a(x)$ is Lipschitz continuous in $x \in (-1, 1)$.

$$f^a(x) = \begin{cases} 1 & x \geq 1, \\ \frac{2}{\pi}\arcsin(x) & 1 \geq x \geq -1, \\ -1 & -1 \geq x. \end{cases} \tag{2.14}$$

   It is a nontrivial exercise to demonstrate that the limit

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \text{sign}(\sin(k) + x)$$

   indeed equals the above expression (2.14). It relies on the fact that the points $k \pmod{2}\pi$ are in some sense uniformly distributed over the interval $[0, 2\pi)$.

2. In the literature see, e.g. Sanders and Verhulst (1985), one sometimes speaks of f satisfying Assumption 2.1, Property 3 as a *KBM function*, because of the contributions to averaging theory made by the researchers Krylov, Boguliobov and Mitropolski.

3. In the following situations the existence of an average is guaranteed :

   - Any function $f_k(x)$ that converges to a function independent of k, i.e. $\lim_{k\to\infty} f_k(x) = g(x)$ has an average given by this limit, i.e. $f^a(x) = g(x)$.
   - Any k*-periodic function* $f_k(x) = f_{k+K}(x)$ (with period K) has an average given by $f^a(x) = (1/K)\sum_{k=1}^{K} f_k(x)$.
   - $f_k(x)$ is a k*-almost periodic function* uniformly in x if for all $\epsilon > 0$ there exists a $K(\epsilon) > 0$ such that for all k, x $\| f_k(x) - f_{k+K}(x)\| \leq \epsilon$. Here $K(\epsilon)$ is called an ϵ*-almost period*. Any finite sum of sinusoidal functions is an almost periodic function, e.g. $\sin(k)$ is an almost periodic function and so is $\sin(k) + \cos(\pi k)$. Any almost periodic function has an average.

The above particular cases do not exhaustively describe all functions for which there exists an average, e.g. $\sin(\sqrt{k})$ has an average, but does not belong to any of the above categories.

The following result is the basic averaging result we are going to exploit.

Theorem 2.2. *Consider* (2.10). *Let $D \subset \mathrm{R}^n$ be compact, $L \in \mathrm{N}$ and $\epsilon > 0$. Define $\delta(\mu)$ as:*

$$\delta(\mu) = \sup_{x\in D} \sup_{k\in[0,L/\mu]} \mu \left\| \sum_{i=0}^{k} \left[f_i(x) - f^a(x) \right] \right\|. \tag{2.15}$$

Suppose that Assumption 2.1 holds. Then $\delta(\mu) = o(1)$. Moreover, the solution $x_k(x_0, 0, \mu)$ of (2.10)*:*

$$x_{k+1} = x_k + \mu f_k(x_k); \qquad k \in \mathrm{N}, \qquad x(0) = x_0, \tag{2.16}$$

can be approximated by $x_k^a(x_0, \mu)$ the solution of

$$x_{k+1}^a = x_k^a + \mu f^a(x_k^a); \qquad k \in \mathrm{N}, \qquad x^a(0) = x_0. \tag{2.17}$$

Furthermore, for any $x_0 \in D$ such that $\inf_{x\in\partial(D)} \|x_0 - x\| \geq \epsilon$ ($\partial(D)$ denotes the boundary of the domain D), there exists a positive constant $\mu^*(D, \epsilon, L)$ such that for all $\mu \in [0, \mu^*)$ and all $k \in [0, L/\mu]$ the approximation error is

$$\left\| x_k(x_0, 0, \mu) - x_k^a(x_0, \mu) \right\| = O(\sqrt{\delta(\mu)}). \tag{2.18}$$

Remark. Under a strengthened continuity assumption, assuming that $f_k(x)$ (and hence also $f^a(x)$) has a uniformly in k Lipschitz continuous partial derivative with respect to x, it is possible to show that the approximation error is $O(\delta(\mu))$ rather than $O(\sqrt{\delta(\mu)})$.

Infinite Horizon Result

Again we consider a system of the form

$$x_{k+1} = x_k + \mu f_k(x_k); \qquad k \in \mathrm{Z}, \tag{2.19}$$

the solutions of which we want to approximate by solutions of

$$x_{k+1}^a = x_k^a + \mu f_k^a(x_k^a); \qquad k \in \square. \tag{2.20}$$

In the previous subsection a finite horizon $O(1/\mu)$ averaging approximation result was introduced.

In this section we pursue an averaging approximation result valid on the whole time axis under the additional condition that the averaged difference equation has a stable and uniformly attracting solution within the domain of interest. We discuss one such result.

First we need to strengthen the notion of an average. As infinite horizon results are envisaged, it is natural to expect that the average $f^a(\cdot)$ is in some sense a uniform over time good approximation for the time-varying function $f_k(\cdot)$:

Definition. The function $f : \mathsf{R}^n \times \mathsf{N} \to \mathsf{R}^n$ has a *uniform average* $f^a : \mathsf{R}^n \to \mathsf{R}^n$ on a compact domain $D \subset \mathsf{R}^n$ if, for all $x \in D$, $k_0, \epsilon > 0$ there exists an $N > 0$ independent of k_0 such that for all $M \geq N$

$$\frac{1}{M}\left\| \sum_{i=k_0}^{M+k_0-1} (f_i(x) - f^a(x)) \right\| < \epsilon. \tag{2.21}$$

Remarks.

1. It follows from the above definition that for all integers $L > 0$ the averaging error

$$\delta^*(\mu) := \sup_{k_0>0} \sup_{x\in D} \sup_{k\in[0,L/\mu]} \mu \left\| \sum_{i=k_0}^{k+k_0} (f_i(x) - f^a(x)) \right\| \tag{2.22}$$

is an $o(1)$ order function, i.e. $\lim_{\mu\to 0} \delta^*(\mu) = 0$.

2. The existence of an average is not sufficient to guarantee the existence of a *uniform* average. In the important situation that $\sum_{i=0}^{k}(f_i(x) - g(x))$ is a bounded function of k then g is a uniform average. Notice that there can be at most one such function g. The k-periodic functions belong to this class. Also k-almost periodic functions possess a uniform average, yet do not necessarily satisfy the condition that $\sum_{i=0}^{k}(f_i(x) - f^a(x))$ is a bounded function of k.

Without loss of generality we assume the equilibrium to be the origin. More precisely we assume:

Assumption 2.3. *Consider the difference equation* (2.20). *Let* $f^a(0) = 0$. *Assume that there exist a neighborhood of* 0, $\Gamma \subset \mathsf{R}^n$, *on which a positive definite, Lipschitz continuous* $V : \Gamma \to \mathsf{R}^+$ *and a positive definite continuous* $W : \Gamma \to \mathsf{R}^+$ *are defined; furthermore there exist constants* $\mu_s > 0$ *and* $c > 0$ *such that for all* $x \in U := \{x \mid V(x) \leq c\}$, *and all* $\mu \in [0, \mu_s]$ *there holds:* $V(x + \mu f^a(x)) - V(x) \leq -\mu W(x)$.

Remark. In Assumption 2.3, the set U is a compact subset of the *domain of attraction* of the equilibrium. The domain of attraction of an equilibrium is the set of all initials conditions for which the trajectories converge to this equilibrium.

In order to establish the infinite horizon result we require besides the existence of a uniform average, that the averaged difference equation possesses a uniformly asymptotically stable equilibrium. For definitions of the stability concepts and some related results, we refer to Appendix B.

We have the following result:

Theorem 2.4. *Consider* (2.19) *and the averaged equation* (2.20). *Let* f *satisfy Assumption 2.1 and have a uniform average* f^a. *Let the origin be a uniformly* *asymptotically stable equilibrium for the averaged equation* (2.20) *in the sense of Assumption 2.3.*

Let D be a compact subset of the domain of attraction. Let E be an interior[†] *subset of D such that trajectories of* (2.20) *starting in E reach the set U, specified in Assumption 2.3, in at most $O(1/\mu)$ time.*

There exists a positive constant μ^* such that for all $\mu \in [0, \mu^*)$ the solutions $x_k(x_0, k_0, \mu)$ of the difference equation (2.19) *for any $k_0 \geq 0$ and for any $x_0 \in E$ can be uniformly approximated by $x_k^a(x_0, \mu)$, the solution of the averaged difference equation* (2.20) *on $k \geq k_0$. That is,*

$$\left\| x_k(x_0, k_0, \mu) - x_k^a(x_0, \mu) \right\| = o(1), \quad \textit{for all } k \geq k_0. \tag{2.23}$$

Moreover, if the equilibrium is locally exponentially stable (in that the matrix $Df^a(0)$ has only eigenvalues with negative real part), then the approximation error can be estimated as $O\left(\sqrt{\delta^*(\mu)}\right)$, an $o(1)$ order function.

Essentially the theorem states that provided the averaged equation has a uniformly stable equilibrium then the approximation error between the original and the approximate solution remains small over the complete trajectory for all those trajectories inside a substantial subset of the domain of attraction.

Remark. As observed in a remark following Theorem 2.2, provided f has a Lipschitz continuous partial derivative with respect to x, and provided the origin is locally exponentially stable, the approximation error estimate can be strengthened to

$$\left\| x_k(x_0, k_0) - x_k^a(x_0) \right\| = O(\delta^*(\mu)). \tag{2.24}$$

C.3 Transforming an adaptive system into standard form

Recall the adaptive system (2.3)

$$\begin{aligned} x_{k+1} &= A(\theta_k)x_k + B_k(\theta_k), \qquad & x_0, \\ \theta_{k+1} &= \theta_k + \mu g_k(\theta_k, x_k), \qquad & \theta_0. \end{aligned} \tag{3.1}$$

The system (3.1) is not in a format directly suited for averaging. Obviously θ is a slow variable and it is on this equation we would like to use the averaging ideas. However, a direct application of averaging is not possible as x_k depends on θ_k. In order to be able to apply averaging it is essential that we can express the dependence of x on θ. This is the main aim of this section.

First we introduce some hypotheses concerning the smoothness of the adaptive system (3.1):

[†] E is called an interior subset of D if $E \subset D$ and $\partial D \cap \partial E = \emptyset$

Assumption 3.1. *Let $\Theta \subset \mathsf{R}^m$ be compact. Consider the difference equation* (3.1)*:*

1. *$A : \mathsf{R}^m \to \mathsf{R}^{p\times p}$ is continuously differentiable with respect to $\theta \in \Theta$.*
2. *$B : \mathsf{R}^m \times \mathsf{N} \to \mathsf{R}^p$ is continuously differentiable with respect to $\theta \in \Theta$.*
3. *B is bounded in k.*
4. *$D_\theta B_k(\theta)$ is bounded in k.*
5. *$g : \mathsf{R}^m \times \mathsf{R}^p \times \mathsf{N} \to \mathsf{R}^m$ is locally bounded and locally Lipschitz continuous in $(\theta, x) \in \Theta \times X \subset \mathsf{R}^m \times \mathsf{R}^p$ uniformly in k.*

We also require that there exist parameter values θ such that the transition matrix A is stable. This is not an unreasonable request. If there were no such θ, then it is highly likely that due to the slow adaptation the x component would become extremely large. Also without such an assumption we have no hope that the adaptation could ever stop, if θ would converge and $A(\theta)$ were unstable then x would diverge.

We make this more precise:

Assumption 3.2. *There exists $r > 1$ such that*

$$S := \{\theta \in \Theta \mid A'(\theta)P(\theta)A(\theta) + I = P(\theta) \quad \text{with } I \le P(\theta) < rI\} \tag{3.2}$$

is nonempty.

Assumption 3.2 states that there is a nonempty subset S of Θ such that for $\theta \in S$ the matrix $A(\theta)$ is a stability matrix whose eigenvalues remain bounded away from the unit circle. (See also Section A.10.)

In order to describe how the slow time scale effects of θ are observed in x we introduce *the frozen system*:

$$x^0_{k+1}(\nu) = A(\nu)x^0_k(\nu) + B_k(\nu), \qquad x^0(0,\nu) = x_0, \qquad k = 0, 1, \ldots. \tag{3.3}$$

Here x_0 equals the initial condition of the x state in the adaptive system description (3.1). $\nu \in \Theta$ is a fixed parameter.

It is not difficult to demonstrate that the solutions of the difference equation (3.3) are for all $\nu \in S$ bounded and differentiable with respect to ν. The following lemma makes this precise.

Lemma 3.3. *Consider the frozen system equation* (3.3). *Under Assumptions 3.1 and 3.2 it follows that for all $\nu \in S$:*

1. *$x^0_k(\nu)$ is bounded uniformly in $\nu \in S$; for some $C_0 > 0$:*

$$\left\| x^0_k(\nu) \right\| \le \sqrt{r}(\sigma^k \|x_0\| + \frac{1-\sigma^k}{1-\sigma} C_0), \tag{3.4}$$

with $\sigma = \sqrt{1 - 1/r}$.

2. *$x_k^0(\nu)$ is continuously differentiable with respect to $\nu \in S$.*

3. *$D_\nu x_k^0(\nu)$ is bounded uniformly in $\nu \in S$; for some $C_1, C_2 > 0$:*

$$\left\| D_\nu x_k^0(\nu) \right\| \leq C_1 C_2 \frac{1-\sigma^k}{\sigma} + C_1 \frac{k\sigma^k}{1-\sigma} \|x_0\| . \tag{3.5}$$

Remark. The relevance of (3.3) can be appreciated by viewing it as the description of the adaptive system (3.1) where the adaptation has been switched off, i.e. $\mu = 0$. It has been associated with the names as *frozen system* or *no adaptation approximation*.

The following result establishes how x in (3.1) depends on the slow variable θ up to terms of order of μ.

Theorem 3.4. *Consider the difference equation* (3.1) *under Assumptions 3.1 and 3.2. Consider also the frozen system* (3.3).

Let (x_k, θ_k) denote the solution of the difference equation (3.1) *starting in (x_0, θ_0) at time k_0. Consider $\theta_0 \in S$. Let $x_k^0(\nu)$ denote the solution of the frozen system* (3.3) *starting in x_0 at the same initial time k_0.*

There exists a positive constant $\mu_0 > 0$ such that for all $\mu \in [0, \mu_0)$ on the time interval $\{k : k \geq k_0$ and $\theta_k \in S\}$ we have that

1. *$x_k^0(\theta_k)$ is an $O(\mu)$ approximation of x_k:*

$$\left\| x_k - x_k^0(\theta_k) \right\| \leq C_x \mu; \quad \text{some } C_x > 0. \tag{3.6}$$

2. *θ_k can be approximated by θ_k^0 up to $O(\mu)$ on a time scale $O(1/\mu)$ where θ_k^0 is the solution of the difference equation:*

$$\theta_{k+1}^0 = \theta_k^0 + \mu g_k(\theta_k^0, x_k^0(\theta_k^0)); \qquad \theta_{k_0}^0 = \theta_{k_0}, \tag{3.7}$$

with

$$\left\| \theta_k - \theta_k^0 \right\| \leq C_\theta \mu; \quad \text{some } C_\theta > 0. \tag{3.8}$$

3. *$x_k - x_k^0(\theta_k^0) = O(\mu)$ on a time scale $O(1/\mu)$.*

Remarks.

1. Theorem 3.4 essentially allows us to decouple the x equation from the θ equation in the adaptive system (3.1). It allows us to study separately a family of linear systems (3.3) and a nonlinear time-varying equation (3.7) in order to find an approximate solution to the complete adaptive system. Moreover the nonlinear time-varying equation governing the θ^0 update, equation (3.7), is in standard form for the application of the averaging results. This implies that we can further simplify the equations. This will be pursued in the next section.

2. Theorem 3.4 establishes approximations on a time scale $O(1/\mu)$ and for as long as θ wanders in a domain $\square$ where $A(\theta)$ is a stability matrix. Whenever $\theta(0)$ is such that $A(\theta(0))$ is a stability matrix, this will be the case on at least a time scale of $O(1/\mu)$ because $\theta_{k+1} - \theta_k$ is of $O(\mu)$. In the special circumstance that some stability property can be established, e.g. an average based approximation for θ^0 has some kind of attractor, strictly contained in the stability domain $\square$, then all approximations established in Theorem 3.4 hold on the whole time axis.

3. Summarizing loosely the content of Theorem 3.4 we have that the solutions x_k, θ_k of the adaptive system (3.1)

$$\begin{aligned} x_{k+1} &= A(\theta_k)x_k + B_k(\theta_k), \qquad && x_0, \\ \theta_{k+1} &= \theta_k + \mu g_k(\theta_k, x_k), && \theta_0, \end{aligned} \tag{3.9}$$

   are $O(\mu)$ approximated on a time interval $O(1/\mu)$ by $x_k^0(\theta_k^0), \theta_k^0$ where x^0 is defined via the difference equation (3.3) (the so-called frozen system) and θ^0 is defined in (3.7).

$$\begin{aligned} x_{k+1}^0(\nu) &= A(\nu)x_k^0(\nu) + B_k(\nu), \qquad && x_0^0(\nu) = x_0, \\ \theta_{k+1}^0 &= \theta_k^0 + \mu g_k(\theta_k^0, x_k^0(\theta_k^0)), && \theta_{k_0}^0 = \theta_{k_0}. \end{aligned} \tag{3.10}$$

C.4 Averaging Approximation

Theorem 3.4 establishes a finite time decoupling of the x and θ equations in the adaptive system (2.3), whereby the θ variable is approximated by θ^0 governed by the difference equation (3.7). This is in standard form for the application of the averaging results discussed in Section C.2. Using the results of Theorems 2.2 and 2.4 we can obtain the following characterization of the solutions of the adaptive system (2.3).

Theorem 4.1. *Consider the adaptive system* (2.3), *the frozen system* (3.3) *and the approximate update* (3.7) *under the Assumptions 3.1 and 3.2. Let* $\theta_0 \in \square$. *Let* $\delta_\theta := \inf_{\nu \in \partial(\square)} \|\theta_0 - \nu\|$.

Let $\Xi = \max\left(\|x_0\|, \sup_{\nu\in\square} \sup_k \|B_k(\nu)\|\right)$.

Assume that the function $g_k(\nu, x_k^0(\nu))$ *has a well defined average* $g^a(\nu)$ *for any* $\nu \in \square$ *with associated order function* $\delta_g(\mu)$:

$$\delta_g(\mu) = \sup_{\nu\in\square(r)} \sup_{k\in[0,L/\mu]} \mu \left\| \sum_{i=0}^{k} \left[g_i(\nu, x_i^0(\nu)) - g^a(\nu) \right] \right\|. \tag{4.1}$$

There exists a positive constant $\mu^a(\delta_\theta, \Xi)$ *such that for all* $\mu \in [0, \mu^a)$ *the solution* x_k, θ_k *of the adaptive system* (2.3) *is approximated on a time scale of* $O(1/\mu)$ *by* $x_k^0(\theta_k^a), \theta_k^a$ *up to order* $O(\delta_g(\mu))$ *where* θ^a *is defined via:*

$$\theta_{k+1}^a = \theta_k^a + \mu g^a(\theta_k^a), \qquad \theta_0^a = \theta_0, \qquad k = 0, 1, \ldots. \tag{4.2}$$

The above result can be extended to an infinite time result provided the averaged equation has some extra stability property.

Theorem 4.2. *Consider the adaptive system* (2.3)*, the frozen system* (3.3) *and the approximate update* (3.7) *under Assumptions 3.1 and 3.2. Let* $\theta_0 \in \square$*. Let* $\delta_\theta := \inf_{\nu \in \partial(\square)} \|\theta_0 - \nu\|$.

Let $\Xi = \max\left(\|x_0\|, \sup_{\nu\in\square} \sup_k \|B_k(\nu)\|\right)$.

Assume that the function $g_k(\nu, x^0(k,\nu))$ *has a well defined uniform average* $g^a(\nu)$ *for any* $\nu \in \square$ *with associated order function* $\delta_g^u(\mu)$*:*

$$\delta_g^u(\mu) = \sup_{\nu\in\square} \sup_{k_0} \sup_{k\in[0,L/\mu]} \mu \left\| \sum_{i=k_0}^{k_0+k} \left[g_i(\nu, x_i^0(\nu)) - g^a(\nu) \right] \right\|. \tag{4.3}$$

Let $\theta_\infty \in \square$ *be a uniformly asymptotically stable equilibrium for the averaged equation* (4.2) *such that Assumption 2.3 holds. Denote by* Θ_∞ *the largest domain of attraction of* θ_∞ *fully contained in* $\square$[‡]*. Let* $\theta_0 \in \Theta_\infty$.

There exists a positive constant $\mu^a(\Theta_\infty, \Xi)$ *such that for all* $\mu \in [0, \mu^a)$ *the solution* x_k*,* θ_k *of the adaptive system* (2.3) *is approximated uniformly in* k *by* $x_k^0(\theta_k^a)$*,* θ_k^a *up to order* $o(1)$ *where* θ^a *is defined by equation* (4.2).

Moreover, if the equilibrium θ_∞ *is locally exponentially stable (i.e. all the eigenvalues of* $Dg^a(\theta_\infty)$ *have a negative real part) then the approximation error is* $O(\delta_g^u(\mu))$.

Remark. The statements of Theorems 4.1 and 4.2 lead to the following important conclusion. If the averaged equation (4.2) which captures the essence of the adaptation mechanism has no attractor in the domain $\square$ where the frozen system is well defined, that is the adaptive mechanism forces the adapted variable θ outside $\square$, then unacceptable behavior is to be expected. In this situation averaging can only be applied on a finite time basis and predicts that the adaptive system will have poor performance. Indeed as θ leaves the stability domain $\square$ the x variable will grow exponentially. Whenever there is an attractor in the domain $\square$ where the frozen system behaves well, averaging can be used for the whole trajectory, hence may be used to analyze the asymptotic performance of the overall adaptive system (2.3). In this case good performance may be achieved.

It transpires that adaptive algorithm design may concentrate on providing the average g^a, see (4.2), with the right properties, namely an attractor close to the points for which the frozen system has the behavior we would like to see.

[‡]This is the set of all initial conditions for which the trajectories start in $\square$, remain in $\square$ and converge to θ_∞.

References

Anderson, B. D. O., Bitmead, R. R., Johnson, C. R., Kokotovic, P. V., Kosut, R. L., Mareels, I. M. Y., Praly, L. and Riedle, B. D. (1986). *Stability of Adaptive Systems: Passivity and Averaging Analysis*, MIT Press, Cambridge, MA.

Anderson, B. D. O. and Kosut, R. L. (1991). Adaptive robust control: Online learning, *Proc. IEEE Conf. on Decision and Control*, Brighton, pp. 297–8.

Anderson, B. D. O. and Moore, J. B. (1979). *Optimal Filtering*, Prentice-Hall, Englewood Cliffs, N.J.

Anderson, B. D. O. and Moore, J. B. (1989). *Optimal Control: Linear Quadratic Methods*, Prentice-Hall, Englewood Cliffs, N.J.

Åstrom, K. J. and Wittenmark, B. (1984). *Computer Controlled Systems: Theory and Design*, Prentice-Hall, Englewood Cliffs, N.J.

Barnett, S. (1971). *Matrices in Control Theory*, Van Nostrand Reinhold Company Inc., New York.

Bart, H., Gohberg, I., Kaashoek, M. A. and Dooren, P. V. (1980). Factorizations of transfer functions, *SIAM J. Control Optim.* **18**: 675–96.

Bellman, R. E. (1970). *Introduction to Matrices*, McGraw-Hill, New York.

Benveniste, A., Metivier, M. and Priouret, P. (1991). *Adaptive Algoritms and Stochastic Approximations*, Vol. 22 of *Applications of Mathematics*, Springer-Verlag, Berlin.

Blackmore, P. (1995). *Discretization Methods for Control Systems Design*, PhD thesis, Australian National University.

Boyd, S. P. and Barratt, C. H. (1991). *Linear Controller Design: Limit Of performance*, Prentice-Hall, Englewood Cliffs, N.J.

Chakravarty, A. and Moore, J. B. (1985). Aircraft flutter suppression via adaptive LQG control, *Proc. American Control Conf.*, Boston, pp. 488–93.

Chakravarty, A. and Moore, J. B. (1986). Flutter suppression using central tendency adaptive pole assignment, *Proc. Control Engineering Conf.*, Sydney, pp. 78–80.

Chen, C. T. (1984). *Linear Systems Theory and Design*, Holt, Rinehart and Winston, New York.

Chen, C. T. (1987). *Linear Control System Design and Analysis*, Holt, Rinehart and Winston, New York.

Chew, K. K. and Tomizuka, M. (1990). Digital control of repetitive errors in a disk drive system, *IEEE Control System Mag.* pp. 16–19.

Cybenko, G. (1989). Approximation by superposition of a sigmoidal function, *J. Math. Control, Signal and Systems* **2**: 302–4.

Dahleh, M. A. and Pearson, J. B. (1987). ℓ_1 optimal controllers for MIMO discrete-time systems, *IEEE Trans. on Automatic Control* **32**.

Dahleh, M. A. and Pearson, J. B. (1988). Optimal rejection of persistent disturbances, robust stability and mixed sensitivity minimization, *IEEE Trans. on Automatic Control* **33**.

Davison, E. J. and Wang, S. H. (1989). Properties of linear time-invariant multivariable systems subject to arbitrary output and state feedback, *IEEE Trans. on Automatic Control* **18**: 24–32.

DeSilva, C. (1989). *Control Sensors and Actuators*, Prentice-Hall, Englewood Cliffs, N.J.

Desoer, C. A., Liu, R. W., Murray, J. and Saeks, R. (1980). Feedback system design: The fractional representation approach to analysis and synthesis, *IEEE Trans. on Automatic Control* **25**(6): 399–412.

Dooren, P. V. and Dewilde, P. (1981). Minimal cascade factorization of real and complex rational transfer matrices, *IEEE Trans. on Circuits Systems* **28**: 390–400.

Doyle, J. C. (1974). Guaranteed margins in LQG regulators, *IEEE Trans. on Automatic Control* **23**(4): 664–5.

Doyle, J. C., Francis, B. A. and Tannenbaum, A. (1992). *Feedback Control Theory*, MacMillan, New York.

Doyle, J. C. and Stein, J. G. (1979). Robustness with observers, *IEEE Trans. on Automatic Control* **24**(4): 607–11.

Elliott, R. E., Aggoun, L. and Moore, J. B. (1994). *Hidden Markov models: Estimation and control,* Springer-Verlag, Berlin.

Feuer, A. and Goodwin, G. (1996). *Sampling in Digital Signal Processing and Control,* Birkhäuser Verlag, Basel.

Francis, B. A. (1987). *A Course in H_∞ Control Theory,* Springer-Verlag, Berlin.

Franklin, G. F. and Powell, J. D. (1980). *Digital Control of Dynamic Systems,* Addison-Wesley, Reading, MA, USA.

Gevers, M. R. (1993). Towards a joint design of identification and control. Presented at a semi-plenary session of Proc. European Control Conf., Groningen, The Netherlands.

Gevers, M. R. and Li, G. (1993). *Parametrizations in Control, Estimation and Filtering Problems,* Springer-Verlag, Berlin.

Goodwin, G. and Sin, K. (1984). *Adaptive Filtering, Prediction, and Control,* Prentice-Hall, Englewood Cliffs, N.J.

Green, M. and Limebeer, D. J. N. (1994). *Linear Robust Control,* Prentice-Hall, Englewood Cliffs, N.J.

Hansen, F. R. (1989). *Fractional Representation Approach to Closed Loop System Indentifcation and Experiment Design*, PhD thesis, Stanford University.

Hara, S. and Sugie, T. (1988). Independent parametrization of two-degrees-of-freedom compensators in general robust tracking, *IEEE Trans. on Automatic Control* **33**(1): 59–68.

Hara, S., Yamamoto, Y., Omata, T. and Nakano, M. (1988). Repetitive control systems: A new type servo system for periodic exogenous systems, *IEEE Trans. on Automatic Control* **33**: 659–68.

Helmke, U. and Moore, J. B. (1994). *Optimization and Dynamical Systems,* Springer-Verlag, Berlin.

Hirsch, M. W. and Smale, S. (1974). *Differential Equations, Dynamical Systems, and Linear Algebra*, Academic Press, New York.

Horowitz, R. and Li, B. (1995). Adaptive control for disk file actuators, *Proc. IEEE Conf. on Decision and Control,* New Orleans, pp. 655–60.

Imae, J. and Hakomori, K. (1987). A second order algorithm for optimal control assuring the existence of Riccati solutions, *J. Society for Instrument and Control Engineers* **23**(4): 410–12.

Imae, J., Irlicht, L. S., Obinata, G. and Moore, J. B. (1992). Enhancing optimal controllers via techniques from robust and adaptive control, *International J. Adaptive Control and Signal Proc.* **6**: 413–29.

Irlicht, L. S., Mareels, I. M. Y. and Moore, J. B. (1993). Switched controller design for resonances suppression, *Proc. IFAC World Congress*, Sydney, pp. 79–82.

Irlicht, L. S. and Moore, J. B. (1991). Functional learning in optimal non-linear control, *Proc. American Control Conf.*, Chicago, pp. 2137–42.

Irwin, M. C. (1980). *Smooth Dynamical Systems*, Academic Press, New York.

Isidori, A. (1989). *Nonlinear Control Systems*, Springer-Verlag, Berlin.

Kailath, T. (1980). *Linear Systems*, Prentice-Hall, Englewood Cliffs, N.J.

Keller, J. P. and Anderson, B. D. O. (1992). A new approach to the discretization of continuous-time controller, *IEEE Trans. on Automatic Control* **37**(2): 214–23.

Kučera, V. (1979). *Discrete Linear Control: The Polynomial Equation Approach*, John Wiley & Sons, New York, London, Sydney.

Kwakernaak and Sivan (1972). *Linear Optimal Control Systems*, John Wiley & Sons, New York, London, Sydney.

Lee, W. S. (1994). *Iterative Identification and Control Design for Robust Performance*, PhD thesis, Australian National University.

Lee, W. S., Anderson, B. D. O., Kosut, R. L. and Mareels, I. M. Y. (1993). A new approach to adaptive robust control, *International J. Adaptive Control and Signal Proc.* **7**: 183–211.

Lehtomaki, N. A., Sandell, Jr., N. R. and Athans, M. (1981). Robustness results in linear quadratic Gaussian based multivariable control design, *IEEE Trans. on Automatic Control* **26**: 75–92.

Li, B. (1995). *Wiener Filter Based Adaptive Control with Applications to the Design of Disk File Servers*, PhD thesis, University of Berkeley.

Ljung, L. (1987). *System Identification: Theory for the User*, Prentice-Hall, Englewood Cliffs, N.J.

Ljung, L. and Söderström, T. (1983). *Theory and Practice of Recursive Identification*, MIT Press, Cambridge, MA.

McFarlane, D. C. and Glover, K. (1989). *Robust Controller Design using Normalized Coprime Factor Plant Descriptions*, Lecture Notes in Control and Information Sciences, Springer-Verlag, Berlin.

Madievski, A., Anderson, B. D. O. and Gevers, M. R. (1993). Optimum realization of sampled-data controllers for FWL sensitivity minimization, *Automatica* **31**: 367–79.

Mareels, I. M. Y. and Polderman, J. W. (1996). *Adaptive control systems: An introduction*, Birkhäuser Verlag, Basel.

Middleton, R. H. and Goodwin, G. (1990). *Digital Control and Estimation*, Prentice-Hall, Englewood Cliffs, N.J.

Moore, J. B., Gangsaas, D. and Blight, J. D. (1982). Adaptive flutter suppression as a complement to LQG based aircraft control, *Proc. IEEE Conf. on Decision and Control*, San Diego, pp. 1191–200.

Moore, J. B., Glover, K. and Telford, A. J. (1990). All stabilizing controllers as frequency shaped state estimate feedback, *IEEE Trans. on Automatic Control* **35**: 203–8.

Moore, J. B., Hotz, A. F. and Gangsaas, D. (1982). Adaptive flutter suppression as a complement to LQG based aircraft control, *Proc. IFAC Identification Conf.*, Boston.

Moore, J. B. and Irlicht, L. S. (1992). Coprime factorization over a class of nonlinear systems, *International J. Robust and Nonlinear Control* **2**: 261–90.

Moore, J. B. and Tay, T. T. (1989a). Adaptive control within the class of stabilizing controllers for a time-varying nominal plant, *IEEE Trans. on Automatic Control* **34**: 367–71.

Moore, J. B. and Tay, T. T. (1989b). Adaptive control within the class of stabilizing controllers for a time-varying nominal plant, *International J. on Control* **50**(1): 33–53.

Moore, J. B. and Tay, T. T. (1989c). Loop recover via H_2/H_∞ sensitivity recovery, *International J. on Control* **49**(4): 1249–71.

Moore, J. B. and Tomizuka, M. (1989). On the class of all stabilizing regulators, *IEEE Trans. on Automatic Control* **34**: 1115–20.

Moore, J. B. and Xia, L. (1987). Loop recovery and robust state estimate feedback designs, *IEEE Trans. on Automatic Control* **32**(6): 512–17.

Moore, J. B. and Xia, L. (1989). On a class of all stabilizing partially decentralized controllers, *Automatica* **25**: 1941–60.

Moore, J. B., Xia, L. and Glover, K. (1986). On improving control loop robustness subject to model matching controllers, *System Control Letters* **7**: 83–7.

Moore, J. B., Xia, Y. and Xia, L. (1989). On active resonance and flutter suppression techniques, *Proc. Australian Aero. Conf.*, Melbourne, pp. 181–5.

Morari, M. and Zafiriou, E. (1989). *Robust Process Control*, Prentice-Hall, Englewood Cliffs, N.J.

Narendra, K. and Annaswamy, A. (1989). *Stable adaptive systems*, Prentice-Hall, Englewood Cliffs, N.J.

Nemhauser, G. L. and Wolsey, L. A. (1988). *Integer and Cominatorial Optimization*, John Wiley & Sons, New York, London, Sydney.

Nett, C. N. (1986). Algebraic aspects of linear control system stability, *IEEE Trans. on Automatic Control* **31**(10): 941–9.

Nett, C. N., Jacobson, C. A. and Balas, M. J. (1984). A connection between state-space and doubly coprime fractional representation, *IEEE Trans. on Automatic Control* **29**(9): 831–2.

Obinata, G. and Moore, J. B. (1988). Characterization of controller in simultaneous stabilization, *System Control Letters* **10**: 333–40.

Ogata, K. (1987). *Discrete Time Control Systems*, Prentice-Hall, Englewood Cliffs, N.J.

Ogata, K. (1990). *Modern Control Engineering*, 2nd edn, Prentice-Hall, Englewood Cliffs, N.J.

Paice, A. D. B. and Moore, J. B. (1990a). On the Youla-Kučera parameterization for nonlinear systems, *System Control Letters* **14**: 121–9.

Paice, A. D. B. and Moore, J. B. (1990b). Robust stabilization of nonlinear plants via left coprime factorizations, *System Control Letters* **15**: 125–35.

Paice, A. D. B., Moore, J. B. and Horowitz, R. (1992). Nonlinear feedback system stability via coprime factorization analysis, *J. Math. Systems, Estimation and Control* **2**: 293–321.

Partanen, A. (1995). *Controller Refinement with Application to a Sugar Cane Crushing Mill*, PhD thesis, Australian National University.

Perkins, J. E., Mareels, I. M. Y. and Moore, J. B. (1992). Functional learning in signal processing via least squares, *International J. Adaptive Control and Signal Proc.* **6**: 481–98.

Polderman, J. W. (1989). *Adaptive Control and Identification: Conflict or Conflux, CWI tract 67*, Centrum voor Wiskunde en Informatica, Amsterdam.

Rohrs, R., Valavani, L. S., Athans, M. and Stein, G. (1985). Robustness of continuous time adaptive control algorithms in the presence of unmodeled dynamcis, *IEEE Trans. on Automatic Control* **30**: 881–9.

Sage, A. P. and White, C. C. (1977). *Optimum Systems Control*, Prentice-Hall, Englewood Cliffs, N.J.

Sanders, J. A. and Verhulst, F. (1985). *Averaging Methods in Nonlinear Dynamical Systems*, Springer-Verlag, Berlin.

Sastry, S. and Bodson, M. (1989). *Adaptive Control*, Prentice-Hall, Englewood Cliffs, N.J.

Schrama, R. J. P. (1992a). Accurate models for control design: The necessity of an iterative scheme, *IEEE Trans. on Automatic Control* .

Schrama, R. J. P. (1992b). *Approximate Identification and Control Design*, PhD thesis, Delft University of Technology.

Solo, V. and Kong, X. (1995). *Adaptive signal processing algorithms: stability and performance*, Prentice-Hall, Englewood Cliffs, N.J.

Sontag, E. D. (1990). *Mathematical Control Theory*, Springer-Verlag, Berlin.

Stein, G. and Athans, M. (1987). The LQG/LTR procedure for multivariable feedback control, *IEEE Trans. on Automatic Control* **32**(2): 105–14.

Tay, T. T. (1989). *Enhancing robust controllers with adaptive techniques*, PhD thesis, Australian National University.

Tay, T. T. and Moore, J. B. (1990). Performance enhancement of two-degree-of-freedom controllers via adaptive techniques, *International J. Adaptive Control and Signal Proc.* **4**: 69–84.

Tay, T. T. and Moore, J. B. (1991). Enhancement of fixed controller via adaptive-Q disturbance estimate feedback, *Automatica* **27**(1): 39–53.

Tay, T. T., Moore, J. B. and Horowitz, R. (1989). Indirect adaptive techniques for fixed controller performance enhancement, *International J. on Control* **50**(5): 1941–59.

Telford, A. J. and Moore, J. B. (1989). Doubly coprime factorization reduced order observers, and dynamic state estimate feedback, *International J. on Control* **50**: 2583–97.

Telford, A. J. and Moore, J. B. (1990). Adaptive stabilization and resonance suppression, *International J. on Control* **52**: 725–36.

Teo, K. L., Goh, C. J. and Wong, K. H. (1991). *A Unified Computational Approach to Optimal Control Problems*, Longman Scientific and Technical, Harlow, Essex.

Teo, Y. T. and Tay, T. T. (1995). Design of an ℓ_1 optimal regulator: The limits-of-performance approach, *IEEE Trans. on Automatic Control* **40**(12).

Vidyasagar, M. (1985). *Control System Synthesis: A Factorization Approach*, MIT Press, Cambridge, MA.

Vidyasagar, M. (1986). Optimal rejection of persistent bounded disturbances, *IEEE Trans. on Automatic Control* **31**(6): 527–34.

Vidyasagar, M. (1991). Further results on the optimal rejection of persistent bounded disturbances, *IEEE Trans. on Automatic Control* **36**(6): 642–52.

Wang, L. and Mareels, I. M. Y. (1991). Adaptive disturbance rejection, *Proc. IEEE Conf. on Decision and Control*, Brighton.

Wang, Z. (1991). *Performance Issues in Adaptive Control*, PhD thesis, University of Newcastle.

Williamson, D. (1991). *Digital Control and Implementation, Finite Wordlength Consideration*, Prentice-Hall, Englewood Cliffs, N.J.

Wolovich, W. A. (1977). *Linear Multivariable Systems*, Springer-Verlag, Berlin.

Wonham, W. M. (1985). *Linear Multivariable Control: A Geometric Approach*, Springer-Verlag, Berlin.

Yan, W. Y. and Moore, J. B. (1992). A multiple controller structure and design strategy with stability analysis, *Automatica* **28**: 1239–44.

Yan, W. Y. and Moore, J. B. (1994). Stable linear matrix fractional transformations with applications to stabilization and multistage H_∞ control design, *International J. Robust and Nonlinear Control* **65**.

Youla, D. C., Bongiorno, Jr., J. J. and Jabr, H. A. (1976a). A modern Wiener-Hopf design of optimal controllers. Part I, *IEEE Trans. on Automatic Control* **21**(1): 3–14.

Youla, D. C., Bongiorno, Jr., J. J. and Jabr, H. A. (1976b). A modern Wiener-Hopf design of optimal controllers. Part II, *IEEE Trans. on Automatic Control* **21**(6): 319–30.

Zang, Z., Bitmead, R. R. and Gevers, M. R. (1991). H_2 iterative model refinement and control rebustness enhancement, *Proc. IEEE Conf. on Decision and Control*, Brighton, pp. 279–84.

Zhang, Z. and Freudenberg, J. S. (1987). Loop transfer recovery with nonminimum phase zeros, *Proc. IEEE Conf. on Decision and Control*, Los Angeles, pp. 956–7.

Author Index

Subject Index