Continuous-Time Self-Tuning Control
Volume I – Design

Peter Gawthrop

Originally published in 1987

doi:10.5281/zenodo.7004513

Foreword

Self-tuning control has traditionally developed in a discrete-time context. In contrast, industrial control systems (whether electronically analogue or digital) appear to the user to be continuous-time devices. This dichotomy has hindered the application of self-tuning controllers. This monograph attempts to bridge this gap by considering self-tuning control in a continuous-time context. This reorientation of self-tuning research is not merely cosmetic. There is a good reason for designing industrial control systems in a continuous-time setting: the real-world is made up of continuous-time objects. This fundamental advantage of continuous-time design will, I hope, become apparent on reading this monograph.

There are a number of apparently competing approaches to self-tuning control to be found in the literature. An objective of this monograph is to provide a unified approach to the design and analysis of such algorithms.

This volume concentrates on the design of continuous-time self-tuning controllers; a companion volume will give details of digital implementation, including Pascal algorithms.

Any research monograph builds upon the work of many people too numerous to mention. However I must acknowledge the long and fruitful collaboration with Dr. David Clarke of the University of Oxford which led directly to many of the ideas to be found in this monograph. Also I must acknowledge the influence of Dr. K.W. Lim of the University of Singapore who, as a research student, made many contributions to the robustness ideas to be found here. I wish to thank Chris Barclay, Ahmad Besharati-Rad, Mohamed Kharbouch, Xiaofeng Liu, Coorous Mohtadi, Markku Nihtila and Panos Nomikos who read though many badly written drafts, made helpful suggestions and eliminated some (but undoubtedly not all) of the errors.

The University of Sussex — P.J. Gawthrop

July 1986

Notation

Numbering

The chapters are numbered from 0 to 10. The sections within each chapter are numbered sequentially using decimal notation; thus section 5 of chapter 2 is numbered as 2.5. Within each section, equations are numbered sequentially from 1. References to equations within a section just give the equation number. References to equations without a section are prefixed by the full section number; thus equation 3 of section 2 of chapter 1 is denoted by equation 1.2.3.

Pages are numbered within each chapter; thus the 5th page of chapter 4 is denoted by 4-5. Left-hand pages also display the chapter number and title; right-hand pages also display the section number and title. It is hoped that the reader will find this system beneficial when searching the book.

Each chapter is followed by a list of references in order of appearance in the chapter. An index to keywords is given at the end of the book.

Symbols

In general, functions of time are written in lower case followed by the time argument (t); thus the system output is symbolised by y(t). The corresponding Laplace transforms are denoted by a $\bar{}$ and followed by the Laplace argument (s); thus the Laplace transformed system output is symbolised by $\bar{y}(s)$. System transfer function polynomials are written in upper case followed by the Laplace argument (s); thus the system transfer function denominator is symbolised by A(s).

Quantities associated with an emulator output are denoted by **; quantities associated with an approximate emulator output (ignoring initial conditions) are denoted by *. Estimated quantities are denoted by $\hat{}$.

An index to the more important symbols appears at the end of the book.

Contents

CHAPTER 2: EMULATORS

CHAPTER 3: EMULATOR-BASED CONTROL

CHAPTER 5: LEAST-SQUARES IDENTIFICATION

CHAPTER 6: SELF-TUNING CONTROL

CHAPTER 9: CASCADE CONTROL

CHAPTER 10: TWO-INPUT TWO-OUTPUT SYSTEMS

CHAPTER 0

Continuous-Time Self-Tuning Control

0.1. INTRODUCTION

Self-tuning control has largely developed within a discrete-time framework; presumably because of the digital technology necessary for the implementation of adaptive control. However, although technology dictates implementation, it need not dictate design. As the world outside the computer is essentially continuous-time, it seems appropriate to design self-tuning controllers in a continuous-time setting although the implementation is digital.

A continuous-time approach to self-tuning control was given by Young in 1965[1]; more recently, and with the benefit of the large amount of work in discrete-time self-tuning, a continuous-time approach has been revived by Egardt[2,3].

In my own research, I tentatively discussed the idea of continuous-time self-tuning in my thesis[4]. Choosing discrete-time transfer functions for self-tuning control based on continuous-time models was explored in reference[5], and a hybrid approach was discussed in references[6,7]. An argument for a fully continuous-time design approach was given in reference[8]. This book brings together some thoughts on the subject of continuous-time self-tuning control arising from the ideas appearing in reference[8].

Of course, most work in model-reference adaptive control has been conducted in a continuous-time setting; but such algorithms are usually of a rather simple form due to the constraints of analogue implementation. However, model-reference adaptive controllers and self-tuning controllers have been shown[2,3] to be closely related.

There have also been a number of attempts to link continuous-time and discrete-time approaches, for example[5,6,9,10].

Within the continuous-time context, Egardt was able to unify a number of apparently diverse algorithms[2,3]. More recently[8], a number of algorithms including model-reference, pole-placement and predictive have been considered within a unified continuous-time context. In this book, these ideas are extended and refined. The notion of an <u>emulator</u> is introduced and is used to unify a number of old algorithms and to generate some new ones. This is introduced by way of the celebrated Smith predictor[11].

The design approach presented in this book is more closely related to control engineering practice than is usual in this field; in particular, the method is motivated by Smith's predictor[11]. It is to be expected that such an approach is likely to lead to <u>robust</u> control algorithms, and this has been proved in certain cases (see chapter 7).

In short, three main ideas are explored in this book:

- Design of self-tuning controllers in a continuous-time (as opposed to a discrete-time) context.

- The use of an *emulator*, an extension of Smith's predictor, to unify and illuminate the design of self-tuning controllers.

- The use of *control weighting* to give self-tuning controllers which are *robust* in the face of neglected system dynamics.

These three ideas are introduced in the following sections.

0.2. THE CONTINUOUS-TIME APPROACH

Most systems of interest to the control engineer exist in a continuous-time setting - they are described by *differential equations*. In contrast, most controllers which are sophisticated enough to have a self-tuning capability are implemented using digital microprocessor technology and as such exist in a discrete-time setting - they are described by *difference* equations. It follows that controllers must often be designed by starting off with a continuous-time system and ending up with a discrete-time controller. We contrast two approaches to such design: *continuous-time design* as in Figure 1; and *discrete-time design* as in Figure 2.

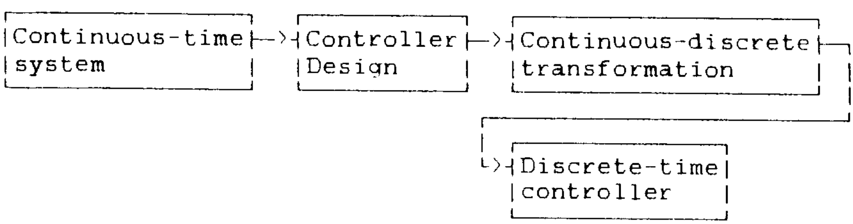


Figure 0.2.1 Continuous-time design

Each design method starts with a continuous-time system and

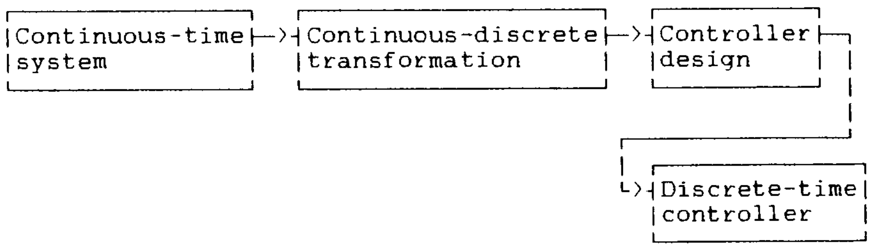


Figure 0.2.2 Discrete-time design

ends with a discrete-time controller; but the design and continuous-discrete transformation steps are transposed between the two methods.

Some advantages of the continuous-time, as opposed to the discrete-time approach are as follows:

- The design method is matched to the actual system to be controlled. Thus system characteristics such as relative degree and zero location can be directly addressed.

- Artefacts of sampling such as sampled minimum phase systems having zeros outside the unit disc[12,13] are avoided.

- The controller coefficients arising from the self-tuning controller correspond to continuous-time (Laplace domain) transfer functions. Most control engineers find these easier to interpret than coefficients of discrete-time (z-domain) transfer functions. An example of this is that the self-tuning PI (proportional plus integral) controller discussed in this book and elsewhere[14] directly estimates the integral time-constant of the controller.

- The controller sample interval is chosen after the design stage, not before.

0.3. EMULATORS

The control of systems with time-delay can be simplified by making use of a predictor. This idea was suggested by Smith in the late '50s[11,15].

Smith's predictor can be regarded as a method of realising the unrealisable transfer function e^{sT}. In particular, it generates the quantity $\bar{y}_T^{\star}(s)$ (Fig 1) given by

$$\bar{y}_T^{\star}(s) = \bar{y}(s) + [1 - e^{-sT}]\frac{B(s)}{A(s)}\bar{u}(s) \qquad (1)$$

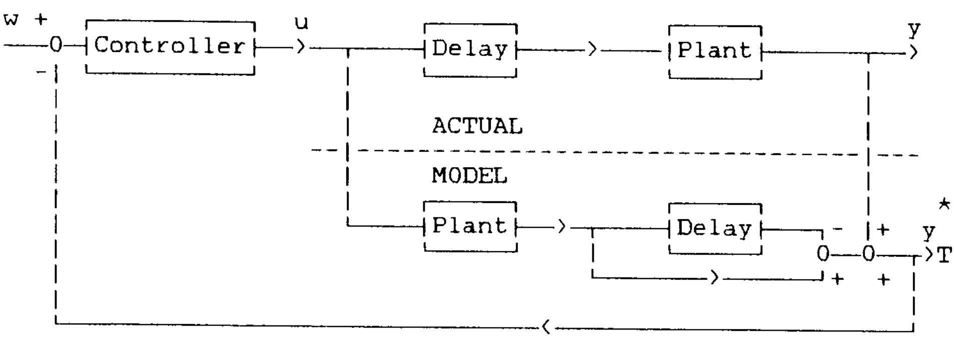


Figure 0.3.1 Smith's Predictor

In the absence of disturbances, substitution of the system equation gives

$$\bar{y}_T^{\star}(s) = \frac{B(s)}{A(s)}\bar{u}(s) = e^{sT}\,\bar{y}(s) = \bar{y}_T(s) \qquad (2)$$

where $\bar{y}_T(s)$ is the Laplace transform of $y_T(t)=y(t+T)$. That is, in the absence of disturbances, the effect of the Smith predictor is the same as including an inverse time delay (e^{sT}) in series with the system output.

The significant thing about this result is that the time delay (e^{-sT}) is cancelled from the system loop-gain by the inverse delay (e^{sT}). That is the closed-loop characteristic equation does not have a time delay factor. This is brought out by drawing the feedback loop as in Figure 2, where the explicit predictor equation 1 is replaced by equation 2.

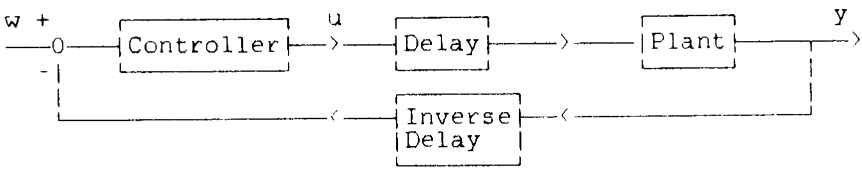


Figure 0.3.2 The equivalent feedback loop

The main points of this discussion are now summarised:

1 A nasty component of the system, a delay, can be removed from the loop gain using an unrealisable component, an inverse delay.

2 An unrealisable component can be emulated using realisable transfer functions operating on both the system input and the system output.

3 Such emulation is only possible if the system transfer function is known.

A particular design method, based on Smith's predictor, can be used to overcome the effect of a nasty system component, a time delay. The method can be interpreted as using an <u>emulator</u> to emulate an unrealisable component (in this case e^{sT}) which cancelled out the nasty component (in this case e^{-sT}).

However, a time delay is not the only bothersome component of a transfer function; there are at least three:

1 A time delay.

2 A high relative degree ρ (a lot more poles than zeros).

3 Zeros with positive real parts (unstable numerator B(s)).

Why not cancel all these out?

The corresponding unrealisable transfer function generates the quantity $\bar{\phi}(s)$ from $\bar{y}(s)$ as

$$\bar{\phi}(s) = e^{sT} \frac{P(s)}{Z(s)}\bar{y}(s) \tag{3}$$

1 e^{sT} cancels out the delay; the net delay is reduced to zero.

2 If degree(P(s))-degree(Z(s)) = ρ (the relative degree of the system), the net relative degree is reduced to zero.

3 If Z(s) contains all the unwanted factors of B(s) then such factors are cancelled; the net number of unstable zeros is reduced to zero.

In this book, the design of such emulators, together with the corresponding fixed and self-tuning controllers, is discussed in some detail.

As the seminal self-tuning regulator of Astrom and Wittenmark[16] was based on a discrete-time predictor, it is not surprising that the emulator, as a generalisation of a predictor, also forms the basis of a self-tuning algorithm.

Finally, we note that the concept of an emulator is not restricted to the continuous-time approach; a discrete-time development is given in reference[17]. However, the notion of an emulator is much more meaningful in a continuous-time setting as it relates directly to the actual continuous-time system.

0.4. ROBUSTNESS

In this book, a controller is said to be robust if it remains stable in the presence of neglected system dynamics. There are two categories of neglected dynamics considered here:

- Neglected dynamics arising from underestimating the order of a single-input single-output system.
- Neglected dynamics arising from neglecting the interaction between loops in a two-input two-output system.

These two situations can be generalised[18], but this generalisation is beyond the scope of this book.

Robustness has received considerable attention in the past few years; see the references for chapter 7. Indeed a book on the subject has recently appeared[19]. Roughly speaking, robustness research can be divided into local robustness meaning stability for sufficiently small initial parameter error and sufficiently small estimation rate and global robustness meaning stability for any initial parameter error and parameter update rate. It is the latter that is discussed in this book.

Much theoretical research was stimulated by the work of Rohrs[20] who showed, by means of simulation, that model reference adaptive control was not robust, in the sense that it could be rendered unstable by quite small neglected dynamics. His two examples[20] have now become standard for illustrating robustness results, and we use his second example in chapters 4 and 7.

The key idea used in this book to give robust control is <u>control weighting</u>. Roughly speaking, the reason why model-reference is not robust is that it tries to match a reference model at <u>all</u> frequencies. This is both unnecessary and dangerous: unnecessary because we are not interested in closed-loop setpoint response at high frequencies; dangerous because it is not usually possible to match a reference model at high frequencies. In chapter 7 it is shown that adaptive robustness is intimately connected with a <u>notional feedback loop</u> which must be stable for robustness. It is found that the notional feedback loop has infinite gain in the absence of control weighting, and this leads to non-robust algorithms.

The conclusion reached in this book is that control weighting at high frequencies is essential for robustness. This conclusion is in accord with my practical experience (for example[21,22]) where control weighting (using the generalised minimum variance algorithm[23,24] has always been used to achieve satisfactory practical control.

The approach used in this book is based on some earlier work on stability and convergence[25,26] utilising the input-output stability approach[27] and also some work on discrete-time robustness[28,29].

0.5. ORGANISATION OF THE BOOK

Apart from this chapter, the book contains a further 10 chapters. The arrangement of material is such that the reader should not need to refer forward to understand a particular topic. The reader may, of course, wish to look forwards for the purposes of motivation. The <u>index</u> is designed in such a way that any topics referred to in the index are underlined unless they actually form part of a section heading.

The chapters in the book are as follows:

1 Continuous-time systems

2 Emulators

3 Emulator-based control

4 Non-adaptive robustness

5 Least-squares identification

6 Self-tuning control

7 Robustness of self-tuning controllers

8 Non-adaptive and adaptive robustness

9 Cascade control

10 Two-input two-output systems

These chapters are outlined in the following subsections.

Continuous-time systems

The background required for this book is that of an undergraduate course in classical continuous-time control from the transfer-function point of view. The book by Dorf[30] would exemplify the sort of material required. This chapter provides the basic ideas and notation used in the rest of the book and could be skimmed through on a first reading. A small amount of material on state-space filters is included as background to the implementation of the self-tuning algorithms.

Emulators

This chapter provides design equations for a number of emulators; including those for reducing relative order, reducing the number of non-mimimum phase zeros and reducing time delay. Algorithms are given in detail. Some care is taken to incorporate system initial conditions into the emulator design, as it is known that these are important in parameter identification[31,32].

Emulator-based control

A number of fixed parameter controllers arise from putting an emulator into a feedback loop. These include: model-reference control, predictive control and pole-placement control. All these controllers may have control weighting giving detuned versions, which, as shown in chapter 7, have desirable robustness properties.

The ideal of a notional feedback system is introduced in this chapter.

Non-adaptive robustness

The robustness of fixed parameter, emulator-based controllers to neglected dynamics is considered in this chapter. As well as being of interest in its own right, this provides a basis for the adaptive robustness properties considered in chapter 7. Rohrs second example[20] is used to illustrate the results.

Least-squares identification

As it is less well known than its discrete-time counterpart, a continuous-time least-squares algorithm is derived in full. It is shown that the algorithm may be regarded as a single-input single-output system with gain (in a special sense[27]) of less than one. This result is central to the robustness analysis of chapter 7.

Discrete-time least-squares is outlined and compared with the continuous-time version. It is shown how continuous-time parameters can be estimated via this method.

Self-tuning control

Putting together emulators, feedback and least-squares identification gives self-tuning control. In particular, we regard a self-tuning controller as a self-tuning emulator within a feedback loop. We distinguish between implicit and explicit algorithms as well as between on and off-line emulator design. The algorithms include implicit versions of model-reference and pole-placement algorithms.

A number of illustrative simulations are given.

Robustness of self-tuning controllers

An error feedback system for the self-tuning controller, in the presence of neglected dynamics, is derived in this chapter and is shown to comprise a linear time-invariant system $M(s)$ in feedback with the single-input single-output system Ω representing the least-squares estimator. It follows that the properties of $M(s)$, in particular the M-locus $M(j\omega)$, are crucial in determining robustness. Some results are proved for a particular version of the self-tuning controller.

The results are illustrated by simulation based on Rohrs's example[20].

This chapter is based on an internal report[33].

Non-adaptive and adaptive robustness

When is adaptive control better than non-adaptive control? This is an unanswered question. This chapter attempts to illuminate this question and its possible answers by comparing the non-adaptive design method of Horowitz[34,35] with a particular self-tuning controller. It is suggested that the adaptive controller has an advantage for slowly varying systems in that an extra degree of design freedom may relieve the sensor noise problem associated with high-gain two degree-of-freedom design.

This chapter is based on a conference paper[36].

Cascade control

Cascade control is a common multi-loop control configuration. This chapter compares and contrasts a number of approaches to this problem in a self-tuning context.

This chapter is based on a conference paper[37].

Two-input two-output systems

The final chapter of the book considers another common control system configuration: an interacting two-loop system. The single-loop self-tuning algorithm is extended to account for loop interaction and the robustness of the resulting scheme is analysed.

This chapter is based on an internal report[38].

References

1. Young, P.C., "The determination of the parameters of a dynamic process," Radio and Electronic Engineer, vol. 29, pp. 345-361, 1965.

2. Egardt, B., "Unification of some continuous-time adaptive control schemes," IEEE Trans. Autom. Control, vol. AC-24, p. 588, 1979.

3. Egardt, B., Stability of adaptive controllers, Springer, 1979.

4. Gawthrop, P.J., Studies in identification and control, 1976. D.Phil. Thesis, Oxford University

5. Gawthrop, P.J., "Some interpretations of the self-tuning controller," Proceedings IEE, vol. 124, no. 10, pp. 889-894, 1977.

6. Gawthrop, P.J., "Hybrid self-tuning control," Proc. IEE, vol. 127, no. 5, pp. 229-236, 1980.

7. Gawthrop, P.J. and Clarke, D.W., "Hybrid self-tuning control and its interpretation," Proceedings of 3rd IMA Conference on Control Theory, Academic Press., 1980.

8. Gawthrop, P.J., "A continuous-time approach to discrete-time self-tuning control," Optimal Control: Applications & Methods, vol. 3, no. 4, pp. 399-414, 1982.

9. Goodwin, G.C., "Some observations on robust estimation and control," in Preprints of the 7th IFAC/IFORS symposium on identification and system parameter estimation, York, U.K., 1985.

10. Goodwin, G.C., Leal, R.L., Mayne, D.Q., and Middleton, R.H., "Rapprochement between continuous and discrete

model-reference adaptive control," Automatica, vol. 22, no. 2, pp. 199-208, 1986.

11. Smith, O.J.M., "A controller to overcome dead-time," ISA transactions, vol. 6, no. 2, pp. 28-33, 1959.

12. Astrom, K.J., Hagander, P., and Sternby, J., "Zeros of sampled systems," Automatica, vol. 20, no. 1, pp. 31-38, 1980.

13. Wellstead, P.E., Zanker, P., and Edmunds, J.M., Wellstead, P.E., Edmunds, J.E., Prager, D., and Zanker, P., "Self-tuning pole/zero assignment regulators," Int. J. Control., vol. 30, no. 1, pp. 1-26, 1979.

14. Gawthrop, P.J., "Self-tuning PID controllers: Algorithms and implementation," IEEE Transactions on Automatic Control., vol. AC-31, no. 3, 1986.

15. Marshall, J.E., Control of time-delay systems, Peter Peregrinus Ltd., 1979.

16. Astrom, K.J. and Wittenmark, B., "On self-tuning regulators," Automatica, vol. 9, pp. 185-199, 1973.

17. Gawthrop, P.J., "An introduction to discrete-time self-tuning control," in Signal processing for control, ed. Jones, R.P., Springer, 1986.

18. Gawthrop, P.J., "Robust self-tuning control of n-input n-output systems," in Preprints of the 7th IFAC Symposium on Identification and System Parameter Estimation, York, U.K., 1985.

19. Anderson, B.D.O., Bitmead, R.R., Johnson, C.R., Kokotovic, P.V., Kosut, R.L., Mareels, I.M.Y., Praly, L., and Reidle, B.D., Stability of adaptive systems: Passivity and averaging analysis, MIT Press, 1986.

20. Rohrs, C.E., Valavani, L., Athans, M., and Stein, G., "Robustness of continuous-time adaptive control in the presence of unmodeled dynamics," Trans. IEEE, vol. AC-30, pp. 881-889, 1985.

21. Clarke, D.W and Gawthrop, P.J., "Implementation and application of microprocessor-based self-tuners," Automatica, vol. 17, no. 1, pp. 233-244, 1981.

22. Proudfoot, C.G., Gawthrop, P.J., and Jacobs, O.L.R., "Self-tuning control of a pH neutralisation process," Proc. IEE, vol. 5 pt D., no. 5, pp. 267-272, 1983.

23. Clarke, D.W and Gawthrop, P.J., "Self-tuning controller," Proceedings IEE, vol. 122, no. 9, pp. 929-934, 1975.

24. Clarke, D.W and Gawthrop, P.J., "Self-tuning control," Proceedings IEE, vol. 126, no. 6, pp. 633-640, 1979.

25. Gawthrop, P.J., "On the stability and convergence of self-tuning controllers," in Proceedings of the IMA conference on 'The Analysis and Optimisation of Stochastic Systems',, Academic Press, 1978.

26. Gawthrop, P.J., "On the stability and convergence of a self-tuning controller," Int. J. Control, vol. 31, no. 5, pp. 973-998, 1980.

27. Desoer, C.A. and Vidyasagar, M., Feedback systems : Input-output properties, Academic Press, London, 1975.

28. Gawthrop, P.J and Lim, K.W., "On the robustness of self-tuning controllers," Proc. IEE, vol. 129 ptD, pp. 21-29, 1982.

29. Lim, K.W., Robustness of self-tuning controllers, D.Phil. Thesis, Oxford University, 1982.

30. Dorf, R.G., Modern control systems, Addison-Wesley, 1980.

31. Gawthrop, P.J., "Parametric identification of transient signals," IMA Journal of Mathematical Control and Information, vol. 1, pp. 117-128, 1984.

32. Gawthrop, P.J., "Parameter identification from non-contiguous data," Proceedings IEE, vol. 131 pt. D, no. 6, pp. 261-265, 1984.

33. Gawthrop, P.J., "Robustness of self-tuning controllers PartI: Single-input single-output systems.," Report CE/T/13, School of Engineering and Applied Sciences, Univ. of Sussex., 1985.

34. Horowitz, I., Synthesis of feedback systems, Academic Press, 1963.

35. Horowitz, I. and Sidi, M., "Synthesis of feedback systems with large plant ignorance for prescribed time-domain tolerances," International Journal of Control, vol. 16, pp. 287-309, 1972.

36. Gawthrop, P.J., "Comparative robustness of non-adaptive and adaptive control," in Proceedings of the IEE conference "Control '85", Cambridge, U.K., 1985.

37. Gawthrop, P.J., "Multi-loop self-tuning control: Cascade systems," in Preprints of the 9th IFAC triennial world congress., ed. K.J. Astrom, vol. VII, pp. 127-132, Budapest, 1984.

38. Gawthrop, P.J., "Robustness of self-tuning controllers PartII: Two-input two-output systems.," Report CE/T/12, School of Engineering and Applied Sciences, Univ. of Sussex., 1985.

CHAPTER 1

Continuous-Time Systems

Aims. To review the system theory required as a background for the rest of the book.

1.1. INTRODUCTION

For most of this book, we shall be concerned with the control of single-input single-output linear time-invariant systems. Multivariable systems will also be considered, but will be built up from the single-input single-output systems examined in this chapter. The assumption of linearity is, as always, more for convenience than for realism.

The assumption of time invariance is to simplify the the description of the systems and the analysis of the algorithms. In must be admitted that with this assumption the current of view of self-tuning methods is inconsistent: part of the motivation for using such methods is that practical systems change with time. Nevertheless, simulation results indicate that slowly time-varying systems can be successfully controlled by self-tuning algorithms.

We shall model systems using the differential equation and Laplace transform transfer function approach. Of course computers see the world in terms of difference equations and z-transforms because they are blinkered by the analogue-digital interface; but it is argued in this book that this is no reason for us to take such a computercentric view of systems.

Systems in this book are formed from three components:

1. The controlled system forced by the control signal.
2. Transient disturbances modelled as the transient response of an input-free dynamic system.
3. Forced disturbances modelled as the output of a dynamic system forced by a signal which cannot be controlled. A special case of a forced disturbance is a stochastic process where the system input is white noise.

These components are treated in the following subsections. They are combined into a standard form in section 1.9.

We shall only cover those topics from system theory which are relevant to this book. Those who are not familiar with basic system and control theory are advised to consult a standard textbook such as[1,2,3].

1.2. TRANSFER FUNCTIONS

The simplifying assumptions of linearity and time-invariance allow dynamic systems to be written as linear differential equations with constant coefficients. The time variable is denoted by t and is assumed to start at $t=0$. We shall take the view that complex systems can be built up by interconnecting elementary subsystems of the form

$$\sum_{i=0}^{n} a'_i \frac{d^{n-i}}{dt^{n-i}} y'(t) = \sum_{i=0}^{n} b'_i \frac{d^{n-i}}{dt^{n-i}} u'(t) \tag{1}$$

y' is the <u>system output</u>, u' the <u>system input</u> and a'_i and b'_i $(0 \leq i \leq n)$ are <u>system coefficients</u>.

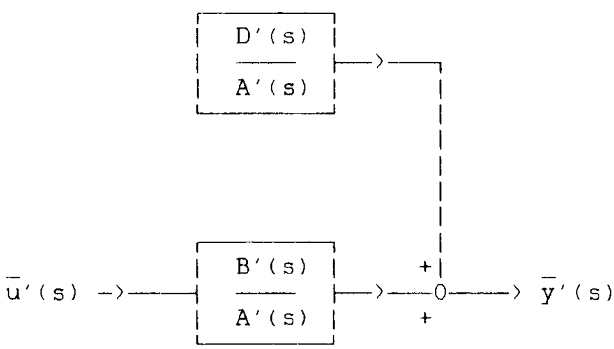
We will assume without loss of generality that

$$a'_0 \neq 0 \tag{2}$$

and thus the <u>system order</u> is n. Let m be the highest value of j for which $b_{n-j} \neq 0$. Then

$$\rho \triangleq n - m \tag{3}$$

is the <u>relative order</u> of the system.



<u>Figure 1.2.1 Laplace transform of subsystem</u>

This equation may be rewritten in terms of Laplace transforms (see Figure 1.2.1) as

$$\bar{y}'(s) = \frac{B'(s)}{A'(s)}\bar{u}'(s) + \frac{D'(s)}{A'(s)} \tag{4}$$

where

$$A'(s) = a'_0 s^n + a'_1 s^{n-1} + \dots + a'_n \tag{5}$$

$$B'(s) = b'_0 s^m + b'_1 s^{m-1} + \dots + b'_m \tag{6}$$

$\bar{y}'(s)$ is the Laplace transform of $y'(t)$, $\bar{u}'(s)$ is the Laplace transform of $u'(t)$, and $D'(s)$ is a n-1th order polynomial dependent on the n <u>initial conditions</u>

$$\frac{d^j y'}{dt^j}(0) \quad i=0..n-1. \tag{7}$$

The <u>system transfer-function</u> is the ratio of the two polynomials

$$H'(s) = \frac{B'(s)}{A'(s)} \tag{8}$$

The transfer function is said to be <u>strictly proper</u> if the relative order $\rho = n-m > 0$, and <u>proper</u> if the relative order $\rho = n-m \geq 0$.

The n system <u>poles</u> are the n roots of $A'(s)=0$; the m (finite) system <u>zeros</u> are the m roots of $B'(s)=0$. If none of the poles has the same value as any of the zeros then the polynomials $A'(s)$ and $B'(s)$ are said to be <u>relatively prime</u> and the transfer function $B'(s)/A'(s)$ is said to have no <u>cancelling factors</u>.

The system <u>frequency-response</u> is defined as

$$H'(j\omega) = \frac{B'(j\omega)}{A'(j\omega)} \tag{9}$$

this complex function of frequency can be interpreted as the ratio of the steady-state system output to the system input when the system input is the unit exponential $e^{-j\omega t}$.

1.3. MARKOV PARAMETERS AND IMPULSE RESPONSE

Equation 1.2.4 reveals that the solution to the differential equation 1.2.1 has two parts: a forced response with Laplace transform

$$H'(s)\bar{u}(s) = \frac{B'(s)}{A'(s)}\bar{u}(s) \tag{1}$$

and a transient response with Laplace transform

$$\frac{D'(s)}{A'(s)} \tag{2}$$

The forced component, involving the transfer function $H'(s)$, determines the effect of the system input on the output and hence is of particular interest in the design of feedback control systems.

A useful notion is the impulse response $h'(t)$ of a system defined as the forced system response when the input is a dirac δ function. As the Laplace transform of a δ function is unity, it follows that

$$\text{Lap}\{h'(t)\} = \frac{B'(s)}{A'(s)} \tag{3}$$

That is, the system transfer function $H'(s)$ is the Laplace transform of the system impulse response $h'(t)$.

The system transfer function can be reexpressed in terms of s^{-1} and the relative order ρ as

$$\frac{B'(s)}{A'(s)} = s^{-\rho}\,\frac{b_0 + b_1 s^{-1} + \ldots + b_m s^{-m}}{a_0 + a_1 s^{-1} + \ldots + a_n s^{-n}} \tag{4}$$

Using repeated algebraic long division, this transfer function can be expressed as a polynomial in s^{-1} as

$$\frac{B'(s)}{A'(s)} = \sum_{i=0}^{\infty} h_i s^{-i} \tag{5}$$

The coefficients h_i are the system <u>Markov</u> <u>parameters</u>[3]. From equation 4 it follows that

$$h_i = 0 \text{ for } i < \rho \tag{6}$$

Multiplying by 1/s (the Laplace transform of a unit step) and taking the inverse Laplace transform, the unit <u>step</u> <u>response</u> of a proper system is given by the <u>Taylor</u> <u>series</u> about t=0

$$h'(t) = h_0 \sum_{i=1}^{\infty} h_i \frac{t^i}{i!} \tag{7}$$

Thus the Markov parameters h_i $i>0$ are the ith derivatives of the unit step response at time t=0+.

The Markov parameter representation is useful for dividing the Laplace transform of derivatives of the impulse response of the system into <u>proper</u> and <u>improper</u> parts. In particular, the transfer function H'(s) multiplied by s^k can be decomposed into a strictly proper transfer function and the rest as

$$s^k H'(s) = s^k \frac{B'(s)}{A'(s)} = E_k(s) + \frac{F_k(s)}{A'(s)} \tag{8}$$

where

$$\deg(F) < \deg(A) \tag{9}$$

(It is shown in standard algebra textbooks, e.g.[4], that this decomposition is unique iff B'(s) and A'(s) are <u>relatively</u> <u>prime</u>).

Equation 8 corresponds to the operation of long division using integers where $F_k(s)$ corresponds to the quotient and $E_k(s)$ the remainder.

The first term represents the non strictly proper part and is given by

$$E_k(s) = \sum_{i=\rho}^{k} h_i s^{k-i} = h_\rho s^{k-\rho} + \ldots + h_k \tag{10}$$

and the second term represents the strictly proper part given by

$$\frac{F_k(s)}{A'(s)} = \sum_{i=k+1}^{\infty} h_i s^{-i} = h_{k+1} s^{-1} + \ldots \tag{11}$$

Denoting the coefficient of s^{n-1} in $F_k(s)$ by f_{k0}, it follows that

$$h_{k+1} = \frac{f_{k0}}{a_0} \tag{12}$$

Those familiar with the discrete-time predictor of Astrom[5] will recognise this decomposition with z replacing s. This is because Markov parameters in discrete-time are the coefficients of the weighting sequence expansion of a z-transfer function[3].

1.4. THE MARKOV RECURSION ALGORITHM

A Markov recursion algorithm giving the Markov parameters h_k, together with the polynomials $E_k(s)$ and $F_k(s)$, can be derived as follows[6]:

Multiplying equation 1.3.8 by s

$$s^{k+1}\frac{B'(s)}{A'(s)} = sE_k(s) + s\frac{F_k(s)}{A'(s)} \tag{1}$$

$$= [sE_k(s) + h_{k+1}] + \frac{[sF_k(s) - h_{k+1}A'(s)]}{A'(s)}$$

where the second equality is obtained by adding h_{k+1} to the first term and subtracting it from the second. Using equation 3.11, the second term of the second equality is proper. Together with equation 3.12 this yields the following recursive algorithm:

$$h_{k+1} = \frac{f_{k0}}{a_0} \tag{2}$$

$$E_{k+1}(s) = sE_k(s) + h_{k+1}$$

$$F_{k+1}(s) = sF_k(s) - h_{k+1}A'(s)$$

The initial polynomials are

$$E_0 = 0; \; F_0 = B'(s) \tag{3}$$

Note that if $k < \rho$ then

$$F_k(s) = s^k B'(s) \tag{4}$$

1.5. STABILITY AND GAIN

Stability

We list some standard stability results for linear time invariant systems described by the transfer function $H'(s) = B'(s)/A'(s)$. These results are intuitively obvious; a deeper treatment is found , for example, in[7,8].

1. The system is stable if the poles of $H'(s)$ (roots of $A'(s)$) have negative real parts.

2. The transient response decays to zero at least as fast as $\kappa e^{-\alpha t}$ for a finite constant κ if the poles of H(s -

α) have negative real parts.

Gain

The <u>gain</u> of a system can be defined in various ways[7,8]. For a linear time invariant system H'(s) the gain γ may be defined as the maximum steady-state sinusoidal gain at any frequency ω

$$\gamma = \sup_{\omega} H'(j\omega) \tag{1}$$

The root mean square of the system output y(t) may be shown[7,8] to be bounded in terms of the system input u(t) by

$$\sqrt{\int_0^t y^2(\tau)d\tau} \leq \gamma \sqrt{\int_0^t u^2(\tau)d\tau} + \kappa \text{ for all } t \tag{2}$$

where κ is a finite positive constant.

The scalar quantity

$$\sqrt{\int_0^t u^2(\tau)d\tau} \tag{3}$$

Is also called the truncated L_2 norm of the signal u(t)[7,8].

Exponential weighting

The <u>exponentially weighted</u> function $y_\alpha(t)$ corresponding to a signal y(t) is defined as

$$y_\alpha(t) \triangleq e^{\alpha t} y(t) \tag{4}$$

Suppose that the impulse response of H'(s) is h'(t). Using the convolution integral, it follows that

$$y(t) = \int_0^t h'(t-\tau)u(\tau)d\tau \tag{5}$$

Substituting for the exponentially multiplied variables

$$y_\alpha(t) = \int_0^t h'_\alpha(t-\tau)u_\alpha(\tau)d\tau \tag{6}$$

where

$$h'_\alpha(t) \triangleq e^{\alpha t}h(t) \tag{7}$$

The transfer function of this exponentially multiplied system is then

$$H'_\alpha(s) = \int_0^\infty e^{-st}h'_\alpha(t)dt = \int_0^\infty e^{-(s-\alpha)t}h'(t)dt = H'(s-\alpha) \tag{8}$$

In other words, the exponentially multiplied signals are related by the same transfer function as the original signals except that 's' is replaced by 's-α' This corresponds to the well known 'shifting on the s axis' theorem of Laplace transforms[9].

1.6. CONTROLLABLE STATE-SPACE REPRESENTATION

The differential equation for a strictly proper subsystem may be written in controllable state-space form as:

$$\frac{d}{dt}\underline{X}^c = \underline{AX}^c + \underline{U}u \tag{1}$$

$$y(t) = \underline{B}^T\underline{X}^c(t) \tag{2}$$

where the companion matrix $\underline{A}$ is given by

$$\underline{A} = \begin{vmatrix} -a'_1 & -a'_2 & -a'_3 & . & -a'_n \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ . & . & . & . & . \\ 0 & 0 & 0 & 1 & 0 \end{vmatrix} \tag{3}$$

$$\underline{B}^T = [b_1', b_2', .., b_n'] \tag{4}$$

$$\underline{U}^T = [1,0,0,\ldots,0] \qquad (5)$$

If the subsystem is not strictly proper ($b_0 \neq 0$) the system has a direct feedthrough term. For the purposes of this book, we handle this in a rather unconventional way by using an <u>extended state vector</u>. The single nth order differential equation 1.6.1 is recast as n first order differential equations and an algebraic equation

$$\frac{d}{dt}x^c_i = x^c_{i-1} \quad i=1..n \qquad (6)$$

$$a_0 x^c_0 = u - a_1 x^c_1 - a_2 x^c_2 - \ldots - a_n x^c_n \qquad (7)$$

The <u>extended state vector</u> is then defined as

$$\underline{X}^c = [x^c_0, x^c_1, \ldots, x^c_n]^T \qquad (8)$$

($x^c_1 - x^c_n$ forms the state; x^c_0 is the extension)

Taking Laplace transforms (with zero initial conditions) of equations 6 and 7 gives

$$s\bar{x}^c_i = \bar{x}^c_{i-1} \qquad (9)$$

$$a_0 \bar{x}^c_0 = \bar{u}(s) - \sum_{i=1}^{n} s^{-i} \bar{x}^c_0 \qquad (10)$$

and so

$$\bar{x}^c_0 = \frac{s^n}{A'(s)} \qquad (11)$$

It follows that (with zero initial conditions) the Laplace transform of the extended state vector is

$$\bar{\underline{X}}^c(s) = \frac{1}{A'(s)} \begin{vmatrix} s^n \\ s^{n-1} \\ \cdot \\ 1 \end{vmatrix} \bar{u}(s) \qquad (12)$$

In this formulation, the states are all derivatives of x^c_n. For this reason, x^c_n is sometimes called the partial state ξ of the system[3]. With zero initial conditions, the partial state ξ can be written in terms of the system input and output as

$$\xi = \bar{x}^c_n = \frac{1}{A'(s)}\bar{u}(s) = \frac{1}{B'(s)}\bar{y}(s) \qquad (13)$$

1.7. OBSERVABLE STATE-SPACE REPRESENTATION

An alternative state-space representation is:

$$\frac{d}{dt}\underline{X}^o = \underline{A}\underline{X}^o + \underline{H}u \qquad (1)$$

$$y = \underline{U}^T\underline{X}^o = x^o_n \qquad (2)$$

where $\underline{A}$ and $\underline{U}$ are as before and

$$\underline{H}^T = [h_n, h_{n-1}, \ldots, h_1] \qquad (3)$$

where h_i is the ith Markov parameter of the system. As in the controllable representation, this may be rewritten in terms of n first order differential equations and one algebraic equation as:

$$x^o_0 = h_n u - a_1 x^o_1 - a_2 x^o_2 - .. - a_n x^o_n \qquad (4)$$

$$\frac{d}{dt}x^o_i = x^o_{i-1} \quad i=1..n + h_{n-i}u \; i=1..n \qquad (5)$$

The extended state vector is then defined as

$$\underline{X}^o = [x^o_0, x^o_1, .., x^o_n]^T \qquad (6)$$

Taking the Laplace transforms (with zero initial conditions)

$$\bar{x}^o_n = \bar{y}(s) = \frac{B'(s)}{A'(s)}\bar{u}(s) \tag{7}$$

hence, taking the Laplace transforms

$$\bar{x}^0_{n-1} = [s\frac{B'(s)}{A'(s)} - h_1]\bar{u}(s) = \frac{F'_1(s)}{A'(s)}\bar{u}(s) \tag{8}$$

Proceeding in this fashion, it follows that

$$\bar{x}^0_{n-k} = [s\frac{F'_{k-1}(s)}{A'(s)} - h_k]\bar{u}(s) = \frac{F'_k(s)}{A'(s)}\bar{u}(s) \tag{9}$$

that is

$$\bar{\underline{X}}^o(s) = \frac{1}{A'(s)}\begin{vmatrix} F'_n(s) \\ F'_{n-1}(s) \\ \cdot \\ B'(s) \end{vmatrix}\bar{u}(s) \tag{10}$$

Thus the observable form is closely related to the <u>Markov recursion algorithm</u> of section 1.4.

1.8. TIME DELAYS

Many practical systems include a pure time delay. One class of subsystems with a pure input delay can be modelled as

$$\sum_{i=0}^{n} a'_i \frac{d^{n-i}}{dt^{n-i}} y'(t) = \sum_{i=0}^{n} b'_i \frac{d^{n-i}}{dt^{n-i}} u'(t-T) \tag{1}$$

where T is the duration of the delay. If <u>the initial conditions corresponding to the time delay are zero</u> the corresponding Laplace transformed system is

$$\bar{y}'(s) = e^{-sT}\frac{B'(s)}{A'(s)}\bar{u}'(s) + \frac{D'(s)}{A'(s)} \tag{2}$$

The modelling of systems with non-zero initial conditions corresponding to the delay is more difficult[10,11]. We shall not consider it in this book.

1.9. THE SYSTEM EQUATION

The systems considered in this book are composed of a number of subsystems representing the effect of the control signal and the disturbances affecting the process. These subsystems are considered in turn and then combined to form the overall system model of the form

$$\bar{y}(s) = e^{-sT} \frac{B(s)}{A(s)}\bar{u}(s) + \frac{C(s)}{A(s)}\bar{v}(s) + \frac{D(s)}{A(s)} \tag{1}$$

The issues involved in modelling the disturbances are then considered.

The controlled system equation

The controlled system is modelled by the equations of section 1.2 with y' replaced by y^c and with a time delay included

$$\bar{y}^c(s) = e^{-sT} \frac{B^c(s)}{A^c(s)}\bar{u}(s) + \frac{D^c(s)}{A^c(s)} \tag{2}$$

Transient disturbances

Some disturbances may be modelled as the transient response of a dynamic system. In Laplace transform form such a disturbance $y^t(t)$ can be written in the form of 1.2.4 with $u' = 0$ as

$$\bar{y}^t(s) = \frac{B^t(s)}{A^t(s)} \tag{3}$$

Example: Constant

The constant disturbance

$$y^t(t) = k \tag{4}$$

can be modelled as

$$\bar{y}^t(s) = \frac{k}{s} \tag{5}$$

Example: Sinusoid

The sinusoidal disturbance

$$y^t(t) = \cos \omega_0 t \tag{6}$$

can be modelled as

$$\bar{y}^t(s) = \frac{s}{s^2 + \omega_0^2} \tag{7}$$

Forced disturbances

Practical disturbances are often too irregular to be modelled as transient disturbances but are nevertheless smooth enough to be predicted over a limited time horizon. Such disturbances can be usefully modelled as a high bandwidth random signal v(t) passed through a transfer function

$$\bar{y}^f(s) = \frac{B^f(s)}{A^f(s)}\bar{v}(s) \tag{8}$$

Example: Random jumps

A piecewise constant signal with jumps of random amplitude at random times can be modelled as

$$\bar{y}^f(s) = \frac{1}{s}\bar{v}(s) \tag{9}$$

$$\bar{v}(s) = \sum_{i=0}^{\infty} k_i e^{-sT_i} \tag{10}$$

where k_i is a sequence of random amplitudes and T_i a sequence of random times.

Example: Random process

A stochastic process with rational spectral density

$$\frac{B^f(-s)B^f(s)}{A^f(-s)A^f(s)} \tag{11}$$

may be modelled by passing white noise through a rational transfer function; see[5,12] for a detailed discussion. To avoid the mathematical details of stochastic process, we will consider a model of the form

$$\bar{y}^f(s) = \frac{B^f(s)}{A^f(s)}\bar{v}(s) \tag{12}$$

where $\bar{v}(s)$ is a finite variance, high bandwidth stochastic process.

The system model

The disturbed single-input single-output system (Figure 1.9.1) considered here is of the form

$$\bar{y}(s) = e^{-sT}\frac{B(s)}{A(s)}\bar{u}(s) + \frac{C(s)}{A(s)}\bar{v}(s) + \frac{D(s)}{A(s)} \tag{13}$$

This can arise from the three types of subsystems in

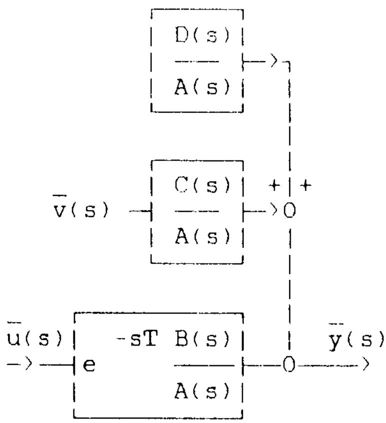


Figure 1.9.1 The system model

various ways. In particular, if

$$y(t) = y^c(t) + y^t(t) + y^f(t) \tag{14}$$

then

$$\bar{y}(s) = e^{-sT} \frac{B^c(s)}{A^c(s)}\bar{u}(s) + \frac{D^c(s)}{A^c(s)} + \frac{B^t(s)}{A^t(s)} + \frac{B^f(s)}{A^f(s)}\bar{v}(s) \tag{15}$$

This is identical to equation 1.9.1 if

$$A(s) = A^c(s)\ A^t(s)\ A^f(s) \tag{16}$$

$$B(s) = B^c(s)\ A^t(s)\ A^f(s) \tag{17}$$

$$C(s) = B^f(s)\ A^c(s)\ A^f(s) \tag{18}$$

$$D(s) = D^c(s)\ A^t(s)\ A^f(s) + B^t(s)\ A^c(s)\ A^f(s) \tag{19}$$

This example makes it clear that systems written in the form of 1.9.1 will usually contain common factors in the numerator and denominator of the various terms. This implies that when each term is written in controllable state-space form it will be unobservable, and when written in observable state-space form it will be uncontrollable. See, for example[3].

Assumptions about the disturbance

In many cases, the disturbance component of the system is such that we would not wish to differentiate it. Given that $\bar{v}(s)$ contains white noise or impulsive components, this can be modelled by making

Disturbance assumption 1

$$\deg(C) = \deg(A) - 1$$

An even worse case would be when we would not wish even to use the system output directly. This can be modelled by making

Disturbance assumption 2

$$\deg(C) = \deg(A)$$

Throughout this book we will assume that C(s) is known, or rather available as a controller design parameter for us to choose.

Disturbance assumption 3

C(s) known.

This seems at first sight to be a rather sweeping assumption. But let us suppose for a moment that the system is "really" given by

$$\bar{y}(s) = e^{-sT}\frac{B(s)}{A(s)}\bar{u}(s) + \frac{C'(s)}{A(s)}\bar{v}'(s) + \frac{D(s)}{A(s)} \tag{20}$$

then this can be written in the form of 1.9.1 if

$$C(s)\bar{v}(s) = C'(s)\bar{v}'(s); \quad \text{that is } \bar{v}(s) = \frac{C'(s)}{C(s)}\bar{v}'(s) \tag{21}$$

As the precise details of the disturbance $\bar{v}(s)$ do not concern us here, the fact that $\bar{v}(s)$ is different from $\bar{v}'(s)$ is not important.

A state-space representation

The system equation can be written in observable state-space form as

$$\frac{d}{dt}\underline{X}^o = \underline{AX}^o + \underline{H}_b u + \underline{H}_c v \tag{22}$$

$$y = \underline{U}^t\underline{X}^o = x^o_n \tag{23}$$

Taking Laplace transforms

$$\underline{X}^o = \frac{1}{A(s)}\begin{vmatrix} F^b_n \\ F^b_{n-1} \\ . \\ B \end{vmatrix}\bar{u}(s) + \frac{1}{A(s)}\begin{vmatrix} F^c_n \\ F^c_{n-1} \\ . \\ C \end{vmatrix}\bar{v}(s) \tag{24}$$

Recalling that $F^b_i = s^i B$ for $i<\rho$ it follows that

$$x^o_{n-k} = s^k\frac{B(s)}{A(s)}\bar{u}(s) + \frac{F^c_k}{A(s)}\bar{v}(s) \text{ for } k<\rho \tag{25}$$

References

1. Nagrath, I.J. and Gopal, M., Systems modelling and anaysis, Tata McGraw-Hill, 1982.

2. Dorf, R.G., Modern control systems, Addison-Wesley, 1980.

3. Kailath, T., Linear Systems, Prentice-Hall, 1980.

4. MacLane, S. and Birkhoff, G., Algebra, Macmillan, New York, 1967.

5. Astrom, K.J., Introduction to stochastic control theory, Academic Press, New York, 1970.

6. Gawthrop, P.J. and Clarke, D.W., "Hybrid self-tuning control and its interpretation," Proceedings of 3rd IMA Conference on Control Theory, Academic Press., 1980.

7. Desoer, C.A. and Vidyasagar, M., Feedback systems : Input-output properties, Academic Press, London, 1975.

8. Vidyasagar, M., Input-output analysis of large-scale interconnected systems, Springer, Berlin, 1981.

9. Kreysig, E., Advanced Engineering Mathematics, Wiley, New York.

10. A.T. Fuller, "Optimal nonlinear control systems with pure delay," Int. J. Control., vol. 8, pp. 145-168, 1968.

11. P.J. Reeve, "Optimal control for systems with pure time delay," Int. J. Control, vol. 11, pp. 659-681, 1970.

12. Melsa, J.L. and Sage, A.P., An introduction to probability and stochastic processes, Prentice-Hall, 1973.

CHAPTER 2
Emulators

Aims. To introduce the concept of an emulator as the generalisation of a predictor. To describe particular emulators providing emulation of improper transfer functions and derivatives, zero cancellation and prediction. To present design methods for a variety of emulators.

2.1. INTRODUCTION

In 1959, Smith introduced the idea of using a predictor to overcome the problems encountered in controlling a system with dead-time[1]. The Kalman-Bucy filter was developed around the same time[2], followed by the state observer[3]. (See[4] for a tutorial account of such state space methods).

These are all examples of using a model of the system, together with input and output measurements, to deduce signals which cannot be directly measured. The Smith predictor deduces future values of the system output; the Kalman filter and state observer deduce system states. The term inferential control has been used to describe control systems containing elements which infer unmeasured variables[5,6].

All these examples illustrate an approach to control systems design where physically unrealisable operations such as prediction or taking derivatives can be emulated by making use of a parametric system model. We shall call the dynamic systems which emulate unrealisable operations emulators.

> [The Concise Oxford Dictionary defines the verb 'emulate' as "Try to equal or excel; rival; imitate zealously". We use the last meaning in this book. 'Emulator' is the corresponding noun.]

In this chapter we shall consider three classes of such unrealisable operations and their corresponding emulators; those corresponding to:

1. Derivatives

2. Zero cancellation

   and

3. Prediction.

Why are such emulators useful? Derivatives are useful to reduce the relative degree ρ of a system, zero cancellation is useful to reduce the number of non-minimum phase system zeros, and predictors are useful to reduce system time delay. These aspects are considered further in chapter 3, where the emulator is put into a feedback loop.

The difficulty with emulators (as with predictors) is that an accurate system model is required before the emulator can be designed. Self-tuning emulators, and the corresponding self-tuning controllers, are introduced in chapter 6 to overcome this problem.

2.2. OUTPUT DERIVATIVES

In the presence of noise, it is usually not feasible to take derivatives of the system output. This is reflected in our model by the disturbance assumptions 1 and 2 of section 1.9 that the relative order of the disturbance transfer function $C(s)/A(s)$ is either 0 or 1. The former case implies that we would not wish to use y directly without low-pass filtering, the latter that we could use y but could not take any derivatives.

In this section we show that it is possible to emulate the operation of taking a derivative without introducing white noise and its derivative. The method is closely related to the state-space observable form of section 1.7, and hence to state observers[3,4].

As most of the development is in the Laplace domain, it is convenient to consider s-multiplied signals in the Laplace domain rather than signal derivatives in the time domain. The two approaches are the same if initial conditions are zero; and in any case the resultant stability properties are the same.

The s^k multiplied system output (equation 1.9.1) is

$$\bar{y}_k(s) = s^k\bar{y}(s) = s^k\frac{B(s)}{A(s)}\bar{u}(s) + s^k\frac{C(s)}{A(s)}\bar{v}(s) + s^k\frac{D(s)}{A(s)} \qquad (1)$$

Using the Markov parameter expansion of section 1.4, the s^k multiplied disturbance transfer function may be decomposed into two parts

$$s^k\frac{C(s)}{A(s)} = E_{1k}(s) + \frac{F_{1k}(s)}{A(s)} \qquad (2)$$

where

$$E_{1k}(s) = h_0 s^k + h_1 s^{k-1} + \ldots + h_k \tag{3}$$

h_i (i = 0..k) are the first k <u>Markov</u> <u>parameters</u> of C(s)/A(s) and

$$\deg(F) < \deg(A) \tag{4}$$

The transfer function $F_{1k}(s)/A(s)$ represents the strictly proper part of $s^k\frac{C(s)}{A(s)}$ and $E_{1k}(s)$ the improper remainder. Such a decomposition is unique (if C(s) and A(s) have no common factors)[7].

In a similar fashion, the s^k multiplied initial condition term can be decomposed as:

$$s^k\frac{D(s)}{A(s)} = E^D_{1k}(s) + \frac{F^D_{1k}(s)}{A(s)} \tag{5}$$

The first term $E^D_{1k}(s)$ is a polynomial in s; the corresponding time domain function contains impulse functions and their derivatives; this term is thus not realisable. On the other hand, the second term $\frac{F^D_{1k}(s)}{A(s)}$ is a <u>proper</u> transfer function.

Using this <u>realisability</u> <u>decomposition</u>, $\bar{y}_k(s)$ may be written as the sum of an emulated value $\bar{y}_k^{**}(s)$ and the corresponding error $\bar{e}^{**}_{1k}(s)$:

$$\bar{y}_k(s) = \bar{y}_k^{**}(s) + \bar{e}^{**}_{1k}(s) \tag{6}$$

where

$$\bar{y}_k^{**}(s) = s^k\frac{B(s)}{A(s)}\bar{u}(s) + \frac{F_{1k}(s)}{A(s)}\bar{v}(s) + \frac{F^D_{1k}(s)}{A(s)} \tag{7}$$

and

$$\bar{e}^{**}_{1k}(s) = E_{1k}(s)\bar{v}(s) + E^{D}_{1k}(s) \tag{8}$$

Equation 7 cannot be implemented as it stands as $\bar{v}(s)$ is unknown. But from the system equation 1.9.1

$$\bar{v}(s) = \frac{A(s)}{C(s)}\bar{y}(s) - \frac{B(s)}{C(s)}e^{-sT}\bar{u}(s) - \frac{D(s)}{C(s)} \tag{9}$$

Hence

$$\bar{y}_k^{**}(s) = [s^k\frac{B(s)}{A(s)} - \frac{F_{1k}(s)}{A(s)}\frac{B(s)}{C(s)}]e^{-sT}\bar{u}(s) + \frac{F_{1k}(s)}{C(s)}\bar{y}(s) \tag{10}$$

$$+ [\frac{F^{D}_{1k}(s)}{A(s)} - \frac{F_{1k}(s)}{A(s)}\frac{D(s)}{C(s)}]$$

Using the decomposition identity 2

$$[s^k\frac{B(s)}{A(s)} - \frac{F_{1k}(s)}{A(s)}\frac{B(s)}{C(s)}] = \frac{E_{1k}(s)B(s)}{C(s)} \tag{11}$$

Using the decomposition identity 5

$$\frac{F^{D}_{1k}(s)}{A(s)} - \frac{F_{1k}(s)}{A(s)}\frac{D(s)}{C(s)} = \frac{E_{1k}(s)D(s) - E^{D}_{1k}(s)C(s)}{C(s)} \tag{12}$$

Hence

$$\bar{y}_k^{**}(s) = \frac{F_{1k}(s)}{C(s)}\bar{y}(s) + \frac{E_{1k}(s)B(s)}{C(s)}e^{-sT}\bar{u}(s) + \frac{I_{1k}(s)}{C(s)} \tag{13}$$

where

$$I_{1k}(s) = E^{D}_{1k}(s)C(s) - E_{1k}(s)D(s) \tag{14}$$

Remarks

1. $\frac{F_{1k}(s)}{C(s)}$ is, by definition, proper.

2. The relative degree of $\frac{E_{1k}(s)B(s)}{C(s)}$ is $\rho - k$ where ρ is the relative degree of the controlled system transfer function $\frac{B(s)}{A(s)}$. For this term to be realisable, we must have $k \leq \rho$.

3. The emulator is constructed in such a way that the initial condition term $F^D_{1k}(s)/A(s)$ is strictly proper. It follows that in its final form, the corresponding emulator term $I_{1k}(s)/C(s)$ is also strictly proper.

As, by definition, C(s) is stable, the initial condition term D(s)/A(s) corresponds to a decaying transient term which becomes small after a time somewhat greater than the time constants associated with C(s). For this reason, the term may be omitted from the predictor to give the approximate predictor (see Figure 2.2.1):

$$\bar{y}_k^{\star}(s) = \frac{F_{1k}(s)}{C(s)}\bar{y}(s) + \frac{E_{1k}(s)B(s)}{C(s)}e^{-sT}\bar{u}(s) \tag{15}$$

with associated error

$$\bar{e}^{\star}_{1k}(s) = \bar{e}^{\star\star}_{1k}(s) + \frac{I_{1k}(s)}{C(s)} \tag{16}$$

The auxiliary output and the emulator

Linear combinations of output derivatives can be readily emulated using such methods. In particular, if the auxiliary output $\bar{\phi}_1(s)$ is defined as

$$\bar{\phi}_1(s) = P(s)\bar{y}(s) \tag{17}$$

$$= p_0 s^n \bar{y}(s) + p_1 s^{n-1}\bar{y}(s) + \ldots + p_n \bar{y}(s)$$

The corresponding emulated auxiliary output can be written as

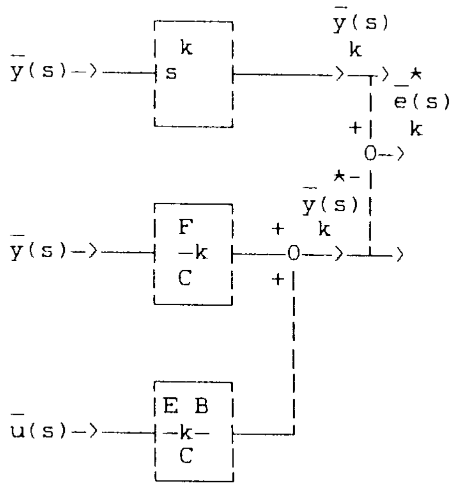


Figure 2.2.1 Emulating output derivatives

$$\bar{\phi}^{**}_{1}(s) = \sum_{k=0}^{n} p_{n-k}\bar{y}_k^{**}(s) \tag{18}$$

with corresponding error

$$\bar{e}_1^{**}(s) = \sum_{k=0}^{n} p_{n-k}\bar{e}^{**}_{1k}(s) \tag{19}$$

Using the explicit expression 2.2.13 for $y_k^{**}(t)$, it follows that

$$\bar{\phi}^{**}_{1}(s) = \frac{F_1(s)}{C(s)}\bar{y}(s) + \frac{E_1(s)B(s)}{C(s)}e^{-sT}\bar{u}(s) + \frac{I_1(s)}{C(s)} \tag{20}$$

where

$$I_1(s) = E_1(s)D(s) - E^D_{1}(s)C(s) \tag{21}$$

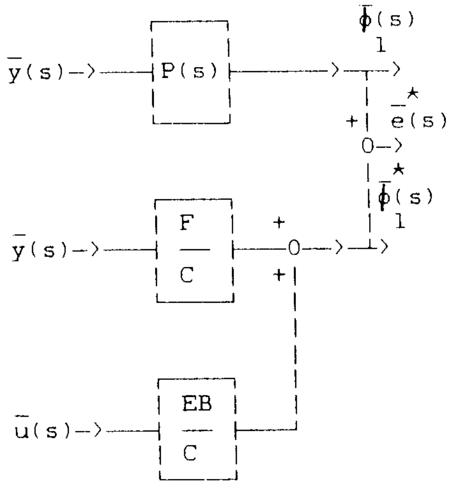


Figure 2.2.2 Emulating the auxiliary output

with associated error

$$\bar{e}^{**}_{1k}(s) = E_1(s)\bar{v}(s) + E^D_1(s) \tag{22}$$

$E_1(s)$, $E^D_1(s)$ and $F_1(s)$ are obtained from

$$E_1(s) = \sum_{k=0}^{n} p_{n-k} E_{1k}(s) \tag{23}$$

$$E^D_1(s) = \sum_{k=0}^{n} p_{n-k} E^D_{1k}(s) \tag{24}$$

$$F_1(s) = \sum_{k=0}^{n} p_{n-k} F_{1k}(s) \tag{25}$$

Alternatively, taking a weighted sum of the Markov decomposition 2, $E_1(s)$ and $F_1(s)$ may be obtained from

$$P(s)\frac{C(s)}{A(s)} = E_1(s) + \frac{F_1(s)}{A(s)} \tag{26}$$

This is the algebraic (s replaces z) continuous-time analogue of the discrete-time generalised minimum variance method in[8,9].

State Space Considerations

Comparison with section 1.9 shows that (assuming zero initial conditions) $y_k^*(t)$ is the kth component of the observable state space form for all $k \leq \rho$. That is

$$\bar{\phi}^*_1(s) = \underline{pX}^o \tag{27}$$

where

$$\underline{p} = [0,0,\ldots,p_{n_p}, \ldots, p_o] \tag{28}$$

See[10] for further details.

Example

Consider the second order system described by

$$A(s) = s(s+1);\; B(s) = 1+bs;\; T = 0 \tag{29}$$

$$C(s) = 1+sc;\; D(s) = 1+ds$$

Applying the Markov recursion formula to $\frac{C(s)}{A(s)}$ we have:

Initial values

$E_{10} = 0$ and $F_{10} = 1+sc$

First Markov parameter

$h_1 = c$

Step 1

$$E_{11} = h_1 = c;$$

$$F_{11} = sF_{10} - h_1 h_1 = s(1+cs) - cs(1+s) = s(1-c)$$

Defining an auxiliary output with $P(s) = 1+ps$

$$E_1 = 1.E_{10} + p.E_{11} = pc \tag{30}$$

$$F_1 = 1.F_{10} + p.F_{11} \tag{31}$$

$$= 1+cs + ps(1-c) = 1 + (p+c-pc)s$$

In a similar fashion:

$$E^D{}_1(s) = pd \tag{32}$$

and so

$$E_1(s)D(s) - E^D{}_1(s)C(s) = pc(1+ds) - pd(1+cs) = p(c-d) \tag{33}$$

Thus

$$\bar{\phi}^{\star}{}_1(s) = \frac{pc(1+bs)}{1+cs}\bar{u}(s) + \frac{1 + (p+c-pc)}{1+cs}\bar{y}(s) + \frac{p(c-d)}{1+cs} \tag{34}$$

Note that all three transfer functions are proper.

In the particular case that

$$p=0.5;\ c=0.5;\ b=0.1;\ d = 0.1 \tag{35}$$

it follows that

$$\bar{\phi}^{*}_{1}(s) = \frac{0.25(1+0.1s)}{1+0.5s}\bar{u}(s) + \frac{1+0.75s}{1+0.5s}\bar{y}(s) + \frac{0.2}{1+0.5s} \qquad (36)$$

□

2.3. ZERO CANCELLING AND OTHER FILTERS

The previous section considers an auxiliary output $\bar{\phi}_1(s)$ which is a polynomial P(s) times the system output; the all zero filter P(s) is not physically realisable due to the implied derivative action. In this section we consider the emulation of a different sort of non-realisable transfer function: multiple derivative action filtered by a possibly unstable polynomial Z(s). Such an emulator can be used to effectively cancel right half plane zeros.

To include the derivative (or, more correctly, s multiplied) emulators of the previous section as a special case, we include derivatives in this section as well. Thus we define the signal $\bar{\xi}_k(s)$ (Figure 2.3.1) by

$$\bar{\xi}_k(s) = \frac{s^k}{Z(s)}\bar{y}(s) \qquad (1)$$

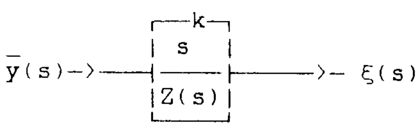


Figure 2.3.1 Zero cancelling filter

Using the system equation 1.9.1,

$$\bar{\xi}_k(s) = s^k \frac{B}{Z(s)A(s)} e^{-sT} \bar{u}(s) + s^k \frac{C}{Z(s)A(s)} \bar{v}(s) + \frac{s^k D}{Z(s)A(s)} \quad (2)$$

As in section 2.2, $s^k C/ZA$ and $s^k D/ZA$ are divided into <u>realisable</u> and non-<u>realisable</u> parts. But first we divide $1/Z$ into notionally realisable and non-realisable parts by defining the polynomials $Z^+(s)$ and $Z^-(s)$ as two factors of $Z(s)$:

$$Z(s) = Z^+(s)Z^-(s) \quad (3)$$

This decomposition is not unique, and particular choices of $Z^+(s)$ and $Z^-(s)$ will depend on the application. $Z^+(s)$ is regarded as the realisable part and $Z^-(s)$ the non-realisable part. The following <u>design rules</u> are imposed:

<u>Z design rule 1</u>

$Z^+(s)$ contains no zeros with positive real part.

<u>Z design rule 2</u>

$Z(s)$ contains no zero at $s=0$.

Note that the first rule implies that $Z^-(s)$ contains all the factors of Z having roots with positive real parts, but may also have roots with negative real part.

With this notation, we can define the polynomials $E_{2k}(s)$ and $F_{2k}(s)$ by

$$s^k \frac{C(s)}{A(s)Z(s)} = \frac{E_{2k}(s)}{Z^-(s)} + \frac{F_{2k}(s)}{A(s)Z^+(s)} \quad (4)$$

where

$$\deg(F_{2k}(s)) < \deg(A(s)Z^+(s)) \quad (5)$$

In terms of polynomials, this equation becomes

$$s^k C(s) = E_{2k}(s) Z^+(s) A(s) + F_{2k}(s) Z^-(s) \tag{6}$$

Note that when $Z=1$, we have $E_{2k}(s) = E_{1k}(s)$ and $F_{2k}(s) = F_{1k}(s)$.

In a similar fashion, the initial condition term can be decomposed as

$$s^k \frac{D(s)}{A(s)Z(s)} = \frac{E^D{}_{2k}(s)}{Z^-(s)} + \frac{F^D{}_{2k}(s)}{A(s)Z^+(s)} \tag{7}$$

where

$$\deg(F^D{}_{2k}(s)) < \deg(A(s)Z^+(s)) \tag{8}$$

We shall defer the solution of these equations for a moment and assume that $E_{2k}(s)$, $E^D{}_{2k}(s)$, $F_{2k}(s)$ and $F^D{}_{2k}(s)$ have been found. Substituting into equation 2

$$\bar{\xi}_k(s) = s^k \frac{B}{Z(s)A(s)} e^{-sT} \bar{u}(s) \tag{9}$$

$$+ \frac{F_{2k}(s)}{A(s)Z^+(s)} \bar{v}(s) + \frac{E_{2k}(s)}{Z^-(s)} \bar{v}(s)$$

$$+ \frac{F^D{}_{2k}(s)}{A(s)Z^+(s)} + \frac{E^D{}_{2k}(s)}{Z^-(s)}$$

As in the previous section, this may be divided into realisable and unrealisable parts as

$$\bar{\xi}_k(s) = \bar{\xi}_k^{**}(s) + \bar{e}_{1k}^{**}(s) \tag{10}$$

and the system equation 1.9.1 used to eliminate $\bar{v}(s)$ to give

$$\bar{\xi}_k^{**}(s) = + \frac{F_{2k}(s)}{Z^+(s)C(s)} \bar{y}(s) + \frac{E_{2k}(s)B(s)}{Z^-(s)C(s)} e^{-sT} \bar{u}(s) \tag{11}$$

$$+ \frac{I_2(s)}{C(s)}$$

where

$$I_2(s) = \frac{E^D{}_{2k}(s)C(s) - E_{2k}(s)D(s)}{Z^-(s)} \tag{12}$$

and

$$\bar{e}_2^{**}(s) = \frac{E_{2k}(s)}{Z^-(s)}\bar{v}(s) + \frac{E^D{}_{2k}(s)}{Z^-(s)} \tag{13}$$

This error signal is never actually generated, so the fact that it is not realisable is not a difficulty.

A particularly important case is when

$$B(s) = B^+(s).B^-(s);\ Z^-(s) = B^-(s) \tag{14}$$

and $B^-(s)$ contains all zeros of $B(s)$ with positive real part. Equation 11 then becomes

$$\bar{\xi}_k^{**}(s) = \frac{F_{2k}(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{E_{2k}(s)B^+(s)}{C(s)}e^{-sT}\ \bar{u}(s) \tag{15}$$

$$+ \frac{I_2(s)}{C(s)}$$

<u>The auxiliary output and the emulator</u>

Linear combinations of filtered output derivatives can be readily emulated using such methods. In particular, if <u>auxiliary output</u> $\bar{\phi}_2(s)$ is defined as

$$\bar{\phi}_2(s) = \frac{P(s)}{Z(s)}\bar{y}(s) \tag{16}$$

the corresponding emulated auxiliary output can be written as

$$\bar{\phi}^{**}_{2}(s) = \sum_{k=0}^{n} p_{n-k}\bar{\xi}_{k}^{**}(s) \tag{17}$$

Note that if $Z(s) = 1$ then $\bar{\phi}_2(s) = \bar{\phi}_1(s)$.

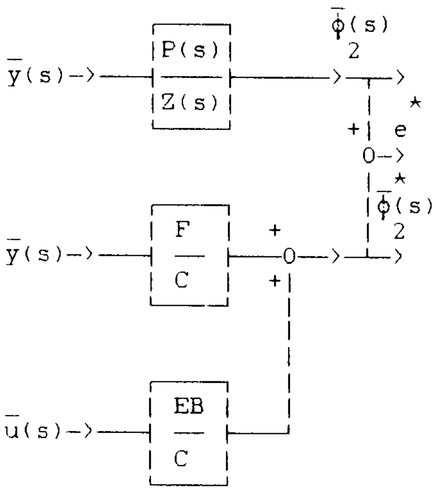


Figure 2.3.2 Emulating the auxiliary output

Using the explicit expression for $\bar{\xi}_k^{**}(s)$, it follows that

$$\bar{\phi}^{**}_{2}(s) = \frac{F_2(s)}{C(s)Z^{+}(s)}\bar{y}(s) + \frac{E_2(s)B(s)}{C(s)Z^{-}(s)}e^{-sT}\,\bar{u}(s) + \frac{I_2(s)}{C(s)} \tag{18}$$

(Figure 2.3.2 shows approximate version) with associated error

$$\bar{e}_2^{**}(s) = \frac{E_2(s)}{Z^{-}(s)}\bar{v}(s) + \frac{E^{D}_{2}(s)}{Z^{-}(s)} \tag{19}$$

$E_2(s)$, $I_2(s)$ and $F_2(s)$ are obtained from

$$E_2(s) = \sum_{k=0}^{n} p_{n-k} E_{2k}(s) \tag{20}$$

$$I_2(s) = \sum_{k=0}^{n} p_{n-k} I_{2k}(s) \tag{21}$$

$$F_2(s) = \sum_{k=0}^{n} p_{n-k} F_{2k}(s) \tag{22}$$

Alternatively, taking a weighted sum of the the Markov decomposition, $E_2(s)$ and $F_2(s)$ may be obtained from

$$\frac{P(s)C(s)}{Z(s)A(s)} = \frac{E_2(s)}{Z^-(s)} + \frac{F_2(s)}{Z^+(s)A(s)} \tag{23}$$

or in polynomial form

$$P(s)C(s) = E_2(s)A(s)Z^+(s) + F_2(s)Z^-(s) \tag{24}$$

and $I_2(s)$ is obtained from

$$\frac{P(s)D(s)}{Z(s)A(s)} = \frac{E^D{}_2(s)}{Z^-(s)} + \frac{F^D{}_2(s)}{Z^+(s)A(s)} \tag{25}$$

and

$$I_2(s) = \frac{E_2(s)D(s) - E^D{}_2(s)C(s)}{Z^-(s)} \tag{26}$$

State Space Considerations

If $Z(s) = B(s)$, then ξ_0 corresponds to the partial state of the system.

Thus, in this case,

$$\bar{\Phi}_2^{\star}(s) = \underline{p}X^c \tag{27}$$

where

$$\underline{p} = [0,0,\ldots,p_{n_p}, \ldots, p_0] \tag{28}$$

It follows that this special case is related to the controllable form of section 1.6.

2.4. SOLVING DIOPHANTINE EQUATIONS

The emulator of the previous section requires the solution of the polynomial equation 2.3.24

$$P(s)C(s) = E_2(s)A(s)Z^+(s) + F_2(s)Z^-(s) \tag{1}$$

This equation is an example of a linear Diophantine equation[11,12,13,7]. This section is devoted to methods of solving such equations.

This Diophantine equation has a solution if, and only if, the greatest common divisor (GCD) of $Z^-(s)$ and $(Z^+(s)A(s))$ is also a factor of $C(s)$[11,12,13,7]. However, we will avoid this problem by arguing as follows. Firstly, we will choose $Z^+(s)$ and $Z^-(s)$ so that they have no common factors. Secondly, the purpose of the filter is to cancel zeros of $\frac{B(s)}{A(s)}$ using the polynomial $Z(s)$. There is no point in cancelling zeros of $B(s)$ which are already cancelled by $A(s)$, so we choose $Z(s)$ so that it has no factors in common with $A(s)$. Hence we would never wish to choose $Z(s)$, $Z^-(s)$ and $Z^+(s)$ in such a way that $Z^-(s)$ and $(Z^+(s)A(s))$ had common factors. Nevertheless, we require a method of checking that this is so, preferably without needing to factorise the polynomials.

This leads to the following three step algorithm for solving equation 1 (that is, equation 2.3.24) for $E_2(s)$ and $F_2(s)$ (this approach is essentially that of[7], page 159; alternative approaches appear in[11,12,13]):

A Use Euclid's algorithm to calculate the GCD $(g(s))$ of $Z^-(s)$ and $(Z^+(s)A(s))$. Compute $Z^{--}(s) = \frac{Z^-(s)}{g(s)}$

B Use Euclid's algorithm to solve the polynomial equation

$$e(s)a(s) + f(s)b(s) = 1 \quad (2)$$

where

$$a(s) = A(s)Z^+(s); \; b(s) = Z^{--}(s) \quad (3)$$

C Use e(s) and f(s) to solve

$$E_{2k}(s)a(s) + F_{2k}(s)b(s) = C(s) \quad (4)$$

The three steps A-C are considered in the following sub sections.

A. Finding the GCD

The classical Euclidian algorithm[7] for finding the GCD of two polynomials is to be found in many textbooks on algebra, for example[7]. Euclid applied the algorithm to integers; it also applies to polynomials, as integers and polynomials possess a similar algebraic structure[7,13].

The algorithm is as follows:

1. Set $\alpha_0 = a(s) = A(s)Z^+(s)$ and set $\alpha_1 = b(s) = Z^-(s)$.
2. Recursively compute the remainder r_i and the quotient q_i from

$$\alpha_{i-1} = q_i\alpha_i + r_i \tag{5}$$

and set

$$\alpha_{i+1} = r_i \tag{6}$$

3. The degree of α_i decreases as i increases, so eventually for some i=n+1, $r_{n+1} = 0$, and so

$$\alpha_n = q_{n+1}\alpha_{n+1} \tag{7}$$

It follows that $\alpha_{n+1} = r_n$ is a factor of α_n. From equation 4 with i = n it follows that r_n is also a factor of α_{n-1}. Repeating this argument, r_n is a factor of both α_0 and α_1.

Thus the GCD g(s) of a(s) and b(s) is the last non-zero remainder r_n of the above algorithm. That is,

$$g(s) = r_n \tag{8}$$

B. Solving the Diophantine Equation

Having found the GCD g(s), we are in a position to compute $Z^{--}(s) = \frac{Z^{-}(s)}{g(s)}$.

1. If degree(g(s))>0 then the previous algorithm is executed but with $Z^{-}(s)$ replaced by $Z^{--}(s)$.

2. Equation 4 with i=n can be rewritten as:

$$\beta_n\alpha_n + \gamma_n\alpha_{n-1} = 1 \tag{9}$$

where

$$\beta_n = -q_n; \; \gamma_n = 1$$

Using equations 4&5 with i=n-1, it follows that

$$\alpha_n = r_{n-1} = \alpha_{n-2} - q_{n-1}\alpha_{n-1} \tag{10}$$

Hence we can write

$$\beta_{n-1}\alpha_{n-1} + \gamma_{n-1}\alpha_{n-2} = 1 \tag{11}$$

where

$$\beta_{n-1} = \gamma_n - \beta_n q_{n-1}; \ \gamma_{n-1} = \beta_n \tag{12}$$

Proceeding in this way the following equations for β_i and γ_i are recursively computed from the following:

<u>Recursive algorithm</u>

$$\beta_{i-1}\alpha_{i-1} + \gamma_{i-1}\alpha_{i-2} = 1 \tag{13}$$

$$\beta_{i-1} = \gamma_i - \beta_i q_{i-1}; \ \gamma_{i-1} = \beta_i \tag{14}$$

$\beta_1 = f(s)$ and $\gamma_1 = e(s)$ then solve

$$e(s)a(s) + f(s)b(s) = 1 \tag{15}$$

C. <u>Diophantine recursion</u>

From the previous equation, we have

$$\frac{e(s)}{b(s)} + \frac{f(s)}{a(s)} = \frac{1}{b(s)a(s)} \tag{16}$$

In other words

$$\frac{e(s)}{Z^{--}(s)} + \frac{f(s)}{A(s)Z^{+}(s)} = \frac{1}{Z^{--}(s)Z^{+}(s)A(s)} = \frac{q(s)}{A(s)Z(s)} \tag{17}$$

and multiplying by s^k

$$s^k \frac{e(s)}{Z^{--}(s)} + s^k \frac{f(s)}{A(s)Z^+(s)} = s^k \frac{g(s)}{A(s)Z(s)} \tag{18}$$

Following the arguments in section 1.3, we can use the <u>Diophantine</u> <u>recursion</u> <u>algorithm</u> to divide the transfer function

$$s^k \frac{f(s)}{A(s)Z^+(s)} \tag{19}$$

into a realisable (derivative free) part, $F'(s)/A(s)Z^+(s)$, and unrealisable $E'(s)$ parts as

$$s^k \frac{f(s)}{A(s)Z^+(s)} = E'(s) + \frac{F'(s)}{A(s)Z^+(s)} \tag{20}$$

Substituting into equation 1 (or 2.3.24) then gives

$$s^k \frac{g(s)}{A(s)Z(s)} = \frac{E_{2k}(s)}{Z^-(s)} + \frac{F_{2k}(s)}{A(s)Z^+(s)} \tag{21}$$

where

$$E_{2k}(s) \triangleq s^k e(s)g(s) + E'(s)Z^-(s) \tag{22}$$

$$F_{2k}(s) \triangleq F'(s) \tag{23}$$

Finally, following the arguments of sections 2.2 & 2.3:

$$\bar{\xi}_k^{\star}(s) = \frac{F_{2k}(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{E_{2k}(s)B(s)}{C(s)Z^-(s)}e^{-sT}\bar{u}(s) \tag{24}$$

<u>Remark</u>

Common factors of $B(s)$ and $Z^-(s)$ should be cancelled <u>before</u> implementation of equation 23.

Example

As in section 2.2, consider the second order system described by

$$A(s) = s(s+1);\ B(s) = 1+bs;\ T = 0 \tag{25}$$

$$C(s) = 1+sc;\ D(s) = 1+ds$$

We wish to derive a zero-cancelling emulator for:

$$\bar{\phi}_2(s) = \frac{P(s)}{Z(s)}\bar{y}(s) \tag{26}$$

where Z(s) is given by:

$$Z(s) = Z^-(s) = 1+zs \tag{27}$$

We shall not specify P(s) at the moment.

As discussed in section 2.4, the corresponding Diophantine equation may be solved in three steps as follows:

A. Find the GCD of $Z^-(s)$ and $A(s)$

Using the algorithm of section 2.4, subsection A, the following equations result:

$$\alpha_0 = A(s)Z^+(s) = s(1+s);\ \alpha_1 = Z^-(s) = 1+zs \tag{28}$$

Using the recursive formula

$$\alpha_{i-1}(s) = q_i(s)\alpha_i(s) + r_i(s) \tag{29}$$

and setting

$$\alpha_{i+1}(s) = r_i(s) \tag{30}$$

the following sequence of polynomials results:

i	α_{i-1}	q_i	α_i	r_i
1	$s(1+s)$	s/z	$1+zs$	$s(z-1)/z$
2	$1+zs$	$z^2/(z-1)$	$s(z-1)/z$	1
3	$s(z-1)/z$	$s(z-1)/z$	1	0

B. Solving the Diophantine equation

Following the algorithm in section 2.4, we have

$$\beta_2 = -q_2 = \frac{z^2}{z-1}; \quad \gamma_2 = 1 \tag{31}$$

Using the recursion equations

$$\beta_{i-1} = \gamma_i - \beta_i q_{i-1}; \quad \gamma_{i-1} = \beta_i \tag{32}$$

with i=2 gives

$$f(s) = \beta_1 = \gamma_2 - \beta_2 q_1 \tag{33}$$

$$= 1 - \frac{z^2}{1-z}\frac{s}{z} = 1 - \frac{zs}{1-z}$$

$$e(s) = \gamma_1 = \beta_2 = \frac{z^2}{1-z} \tag{34}$$

It can be verified by partial fraction expansion that indeed

$$\frac{e(s)}{b(s)} + \frac{f(s)}{a(s)} = \frac{1}{1-z}\left[\frac{z^2}{1+zs} + \frac{1-z-zs}{s(s+1)}\right] = \frac{1}{b(s)a(s)} \tag{35}$$

C. Diophantine recursion

Using the recursive equations of section 2.4,

$$E_{2k}(s) = sE_{2k-1} + h_{1k}Z^{-}(s) \quad (36)$$

$$F_{2k}(s) = sF_{2k-1} - h_{1k}A(s)Z^{+}(s) \quad (37)$$

where

$$h_{1k} = \text{first Markov parameter of } \frac{F_{2k-1}}{A(s)Z^{+}(s)} \quad (38)$$

we get the following sequence of polynomials:

k	h_{1k}	$E_{2k}(s)$	$F_{2k}(s)$
0	-1	$z^2/(1-z)$	$1 - sz/(1-z)$
1	$-z/(1-z)$	$-z/(1-z)$	$s/(1-z)$
2	$1/(1-z)$	$1/(1-z)$	$-s/(1-z)$
3	$-1/(1-z)$	$s - 1/(1-z)$	$s/(1-z)$

In is now possible to compute emulators of various choices of P(s) and C(s) <u>without</u> having to recompute solutions to Diophantine equations. For example

$$P(s) = (1 + 0.5s)^2 = 1 + s + 0.25s^2; \; C(s) = 1+0.5s \quad (39)$$

so

$$P(s)C(s) = (1 + 0.5s)^3 = 1 + 1.5s + 0.75s^2 + 0.125s^3 \quad (40)$$

Using equations 2.3.20&22 and the entries in the Table, $E_2(s)$ and $F_2(s)$ are given by

$$E_2(s) = \frac{1}{1-z}[z^2 - 1.5z + 0.75 - 0.125] + 0.125s \quad (41)$$

$$= \frac{1}{1-z}[z^2 - 1.5z + 0.625) + 0.125s$$

$$F_2(s) = 1 + \frac{s}{1-z}[-z + 1.5 - 0.75 + 0.125] \qquad (42)$$

$$= 1 + \frac{s}{1+z}[0.875 - z]$$

Example: $B(s) = 1+0.1s$

In this case

$$E_2(s) = 0.125s + 0.539; \; F_2(s) = 0.861s + 1 \qquad (43)$$

giving

$$\bar{\phi}^{\star}_2(s) = \frac{0.125s+0.539}{0.5s+1}\tilde{u}(s) + \frac{0.861s+1}{0.5s+1}\tilde{y}(s) \qquad (44)$$

Note that the factor

$$Z^-(s) = B^-(s) = B(s) = 1+0.1s \qquad (45)$$

has been cancelled from the $\tilde{u}(s)$ term of the emulator equation.

This example can be compared with the example of section 2.2.

Example: $B(s) = 1-s$

In this case

$$E_2(s) = 0.125s + 1.562; \; F_2(s) = 0.938s + 1 \qquad (46)$$

giving

$$\bar{\phi}^{\star}_2(s) = \frac{0.125s+1.562}{0.5s+1}\tilde{u}(s) + \frac{0.938s+1}{0.5s+1}\tilde{y}(s) \qquad (47)$$

Note that the factor

$$Z^-(s) = B^-(s) = B(s) = 1-s \qquad (48)$$

has been cancelled from the $\bar{u}(s)$ term of the emulator equation.

□

2.5. PREDICTORS

We now turn to systems with pure time delay which can be written as equation 1.9.1, repeated here as

$$\bar{y}(s) = e^{-sT}\frac{B(s)}{A(s)}\bar{u}(s) + \frac{C(s)}{A(s)}\bar{v}(s) \tag{1}$$

The question of initial conditions becomes difficult in the presence of time delays; so, for simplicity, we will assume zero initial conditions (D(s)=0) in this case.

As pointed out by Smith[1] one approach to designing feedback controllers for such systems is to incorporate a <u>predictor</u> into the feedback loop. This method has been discussed in detail by Marshall[14].

The use of predictors in discrete-time minimum variance control was considered by Astrom in his book[15]; in particular he pioneered the polynomial approach to designing predictors. The presentation in this book is a continuous-time analogue of this method.

Prediction of random functions has a long history. The Weiner filter has a predictive version (see, for example, the book[16] by Kailath). Other relevant books are[17,15,18,19]. The statistical approach is not used in this book.

The purpose of a predictor is to deduce the system output a time T (the system delay) into the future. Putting this together with the previous section suggests an auxiliary function $\bar{\phi}_3(s)$ of the form

$$\bar{\phi}_3(s) = e^{sT}\frac{P(s)}{Z(s)} \tag{2}$$

But firstly, we consider the predictor alone and consider

$$y_T(t) = y(t+T) \tag{3}$$

or in Laplace transform terms

$$\bar{y}_T(s) = e^{sT}\bar{y}(s) \tag{4}$$

Using equation 2.5.1

$$\bar{y}_T(s) = e^{sT}\bar{y}(s) = \frac{B(s)}{A(s)}\bar{u}(s) + e^{sT}\frac{C(s)}{A(s)}\bar{v}(s) \tag{5}$$

The first term of the right-hand side is known. This could, by itself, form a predictor giving

$$\bar{\phi}^{*}(s) = \frac{B(s)}{A(s)}\bar{u}(s) \tag{6}$$

$$\bar{e}^{*}(s) = e^{sT}\frac{C(s)}{A(s)}\bar{v}(s) \tag{7}$$

Due to its open-loop nature this would not usually make a satisfactory predictor.

To obtain a closed-loop predictor we must somehow include the disturbance term $e^{sT}\frac{C(s)}{A(s)}$ in the predictor. But, due to the exponential factor, this term is not causal and hence not realisable. In the same way as in previous sections, this disturbance term is divided into realisable and non-realisable parts; but in this case realisability is associated with causality rather than with properness.

Let the impulse response of $e^{sT}\frac{C(s)}{A(s)}$ be denoted by $h_0(t)$, that is

$$\text{Lap}\{h_0(t)\} = H_0(s) = e^{sT}\frac{C(s)}{A(s)} \tag{8}$$

As $\frac{C(s)}{A(s)}$ is causal

$$h_0(t) = 0 \quad t<-T \tag{9}$$

It follows that $h_0(t)$ can be written as the sum of two functions

$$h_0(t) = h_1(t) + h_2(t) \tag{10}$$

where

$$h_1(t) = 0;\ t\geq 0 \text{ and } h_2(t) = 0;\ t < 0 \tag{11}$$

Thus setting

$$H_1(s) \triangleq \text{Lap}\{h_1(t)\};\ H_2(s) \triangleq \text{Lap}\{h_2(t)\} \tag{12}$$

the disturbance transfer function can be decomposed as

$$e^{sT}\frac{C(s)}{A(s)} = H_0(s) = H_1(s) + H_2(s) \tag{13}$$

<u>Example</u> (<u>Unit integrator</u>)

Suppose that

$$\frac{C(s)}{A(s)} = \frac{1}{s} \tag{14}$$

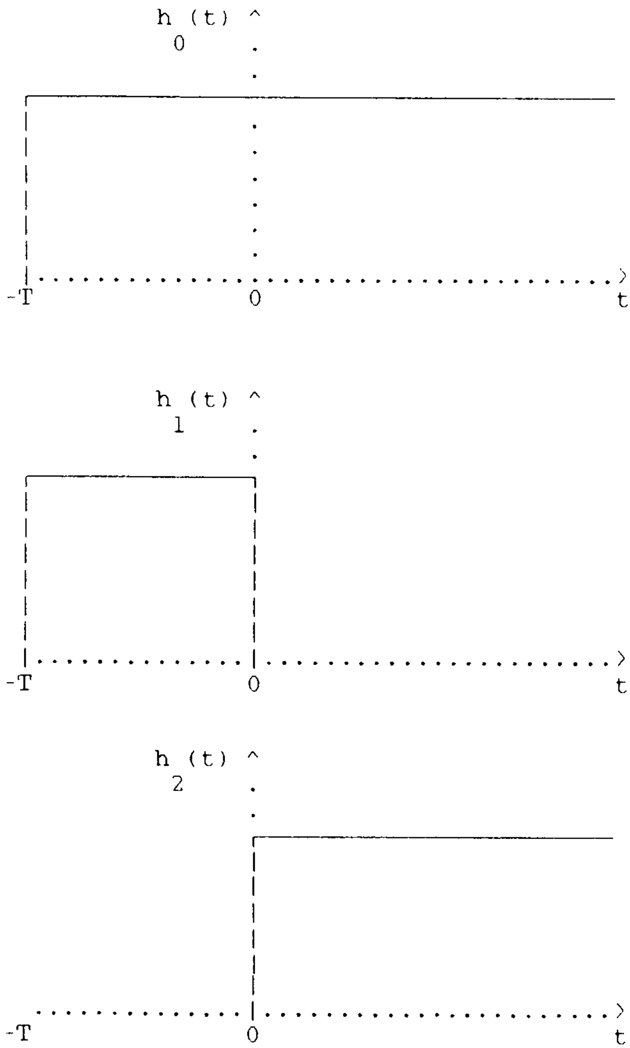
Then

$$h_0(t) = \begin{cases} 1 & t > -T \\ 0 & \text{elsewhere} \end{cases} \tag{15}$$

$$h_1(t) = \begin{cases} 1 & -T < t < 0 \\ 0 & \text{elsewhere} \end{cases}$$

$$h_2(t) = \begin{cases} 1 & t > T \\ 0 & \text{elsewhere} \end{cases}$$

These functions of time are displayed in Figure 2.5.1.



<u>Figure 2.5.1 Realisability decomposition</u> - <u>unit integrator</u>

The corresponding Laplace transforms are:

$$H_0(s) = e^{sT}\frac{1}{s}H_1(s) = \frac{e^{sT} - 1}{s};\; H_2(s) = \frac{1}{s} \tag{16}$$

Note that both transfer functions are proper.

□

Example (Rational transfer function)

Suppose that $\frac{C(s)}{A(s)}$ is rational and A(s) has n distinct roots α_i. Then a partial fraction decomposition is:

$$H_0(s) = e^{sT}\frac{C(s)}{A(s)} = \sum_{i=1}^{n} e^{sT}\frac{r_i}{s - \alpha_i} \tag{17}$$

The corresponding impulse response is

$$h_0(t) = \sum_{i=1}^{n} r_i e^{\alpha_i t+T} \; ; \; t > -T \tag{18}$$

Hence

$$H_1(s) = \sum_{i=1}^{n} e^{sT} r_i \frac{1-e^{-(s - \alpha_i)T}}{s - \alpha_i} \tag{19}$$

and

$$H_2(s) = \sum_{i=1}^{n} r_i \frac{e^{\alpha_i T}}{s - \alpha_i} \tag{20}$$

□

Continuous-time FIR Transfer Functions

In each of the above examples, the realisability decomposition is of the form

$$\frac{C(s)}{A(s)} = E_T(s) + e^{-sT}\frac{F_T(s)}{A(s)} \tag{21}$$

where

$$H_1(s) = e^{sT} E_T(s); \; H_2(s) = \frac{F_T(s)}{A(s)} \tag{22}$$

Having performed the decomposition, the unrealisable quantity $\bar{y}_T(s)$ can be rewritten as

$$\bar{y}_T(s) = \bar{y}_T^{\star}(s) + \bar{e}^{\star}(s) \tag{23}$$

where

$$\bar{y}_T^{\star}(s) = \frac{B(s)}{A(s)}\bar{u}(s) + \frac{F_T(s)}{A(s)}\bar{v}(s) \tag{24}$$

$$\bar{e}^{\star}(s) = H_1(s)\bar{v}(s) = e^{sT} E_T(s)\bar{v}(s) \tag{25}$$

Finally, substituting for $\bar{v}(s)$ in equation 24 and using the decomposition 21, the predictor can be written as

$$\bar{y}_T^{\star}(s) = \frac{E_T(s)B(s)}{C(s)}\bar{u}(s) + \frac{F_T(s)}{C(s)}\bar{y}(s) \tag{26}$$

The transfer function $E_T(s) = e^{-sT} H_1(s)$ has an impulse response which is zero for all time t>T. For this reason it will be called a <u>CFIR</u> or <u>continuous-time</u> <u>finite</u> <u>impulse</u> <u>response</u> system. CFIR transfer functions based on rational transfer functions with <u>distinct</u> <u>poles</u> have the following properties:

1. The impulse response is zero for all time greater than a finite value T.

2. The transfer function has no poles.

3. The transfer function is not rational.

Properties 1 and 3 are obvious; property 2 may be derived as follows:

Property 2

$E_T(s) = e^{-sT} H_1(s)$ comprises n terms of the form

$$r_i \frac{1 - e^{-(s-\alpha_i)T}}{s - \alpha_i} \tag{27}$$

At first sight, this term has a pole at $s=\alpha_i$. But substituting $s=\alpha_i$ into the numerator gives $1 - e^0 = 0$. Thus each of the n apparent poles has zero residue; that is, the function has no poles.

Property 3 is important as it means that $H_1(s)$ cannot be realised using a rational transfer function; however, $H_1(s)$ can be approximated by a rational transfer function. One way of doing this is described in a following section.

The auxiliary output and the emulator

Based on the results of the previous sections, we are in a position to define an auxiliary output $\bar{\Phi}_3(s)$ as

$$\bar{\Phi}_3(s) = e^{sT}\,\bar{\Phi}_2(s) = e^{sT}\,\frac{P(s)}{Z(s)}\bar{y}(s) \tag{28}$$

From the results of section 2.3, it follows that in the presence of a pure time delay (and zero initial conditions):

$$\bar{\Phi}_2^*(s) = \frac{F_2(s)}{A(s)Z^+(s)}\bar{v}(s) + \frac{P(s)B(s)}{A(s)Z^-(s)}e^{-sT}\,\bar{u}(s) \tag{29}$$

hence

$$\bar{\Phi}_3^*(s) = e^{sT}\,\frac{F_2(s)}{A(s)Z^+(s)}\bar{v}(s) + \frac{P(s)B(s)}{A(s)Z^-(s)}\bar{u}(s) \tag{30}$$

The first term is unrealisable, so decompose it into realisable and unrealisable parts as

$$e^{sT} \frac{F_2(s)}{A(s)Z^+(s)} = e^{sT}E_F(s) + \frac{F_3(s)}{A(s)Z^+(s)} \tag{31}$$

We can then define $\bar{\phi}_3^\star(s)$ as the realisable part of $\bar{\phi}_2^\star(s)$

$$\bar{\phi}_3^\star(s) = \frac{F_3(s)}{A(s)Z^+(s)}\bar{v}(s) + \frac{P(s)B(s)}{A(s)Z^-(s)}\bar{u}(s) \tag{32}$$

Finally, combining the system equation 1.9.1 with the two identities 21 and 31

$$\bar{\phi}_3^\star(s) = \frac{F_3(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{E_3(s)B(s)}{C(s)Z^-(s)}\bar{u}(s) \tag{33}$$

where

$$E_3(s) = E_F(s) + Z^-(s)E_2(s) \tag{34}$$

Alternatively, $E_3(s)$ and $F_3(s)$ can be directly expressed as:

$$e^{sT} \frac{P(s)C(s)}{Z(s)A(s)} = e^{sT} \frac{E_3(s)}{Z^-(s)} + \frac{F_3(s)}{Z^+(s)A(s)} \tag{35}$$

2.6. APPROXIMATE TIME DELAYS

The problem with designing controllers for systems with a pure time delay is that the resultant controller is not rational and thus cannot be realised using rational transfer functions. One approach to this problem is to design a controller for a rational system which contains a rational approximation to a time delay.

Time-delay approximation

One class of approximations to time delays have the all-pass transfer function

$$e^{-sT} \cong \frac{T(-s)}{T(s)} \tag{1}$$

where T(s) is a finite order polynomial in s. A particular choice of T(s) is the Pade polynomial of order n_T given by by

$$T(s) = t_0 s^{n_T} + t_1 s^{n_T - 1} + \ldots + t_{n_T} \tag{2}$$

where

$$t_{n_T} = 1 \tag{3}$$

and

$$t_{n_T - i} = \frac{T}{i(n_T - i+1)(2n_T - i+1)} t_{n_T - i+1} \tag{4}$$

See[14] for details.

System approximation

Using this approximation for the time delay, the system can be approximately written as

$$\bar{y}(s) = \frac{T(-s)}{T(s)} \frac{B(s)}{A(s)} \bar{u}(s) + \frac{C(s)}{A(s)} \bar{v}(s) + \frac{D(s)}{A(s)} \tag{5}$$

$$= \frac{B_T(s)}{A_T(s)} \bar{u}(s) + \frac{C_T(s)}{A_T(s)} \bar{v}(s) + \frac{D_T(s)}{A_T(s)}$$

where

$$A_T(s) = T(s)A(s); \; B_T(s) = T(-s)B(s) \tag{6}$$

$$C_T(s) = T(s)C(s);\ D_T(s) = T(s)D(s)$$

<u>The auxiliary output and the emulator</u>

In a similar fashion, we define the auxiliary function $\bar{\phi}_4(s)$ by:

$$\bar{\phi}_4(s) = \frac{P_T(s)}{Z_T(s)}\bar{y}(s) \cong e^{sT}\frac{P(s)}{Z(s)}\bar{y}(s) \tag{7}$$

where

$$P_T(s) \triangleq T(s)P(s);\ Z_T(s) = T(-s)Z(s) \tag{8}$$

The rational system is now of the form considered in section 2.3. Noting that the Pade polynomial T(s) has all roots within the stability, the polynomial T(-s) has all roots without the stability region. Thus the polynomial $Z_T(s)$ is decomposed as:

$$Z_T(s) = Z_T^+(s)Z_T^-(s);\ Z_T^+(s) = Z^+(s);\ Z_T^-(s) = T(-s)Z^-(s) \tag{9}$$

With the above approximations, the polynomial identity 2.4.1 (or 2.3.24) becomes

$$\frac{T(s)P(s)C(s)}{T(-s)Z(s)A(s)} = \frac{E_4(s)}{T(-s)Z^-(s)} + \frac{F_4(s)}{Z^+(s)A(s)} \tag{10}$$

where $\deg(F_T(s)) < \deg(Z^+(s)A(s))$. The corresponding emulator equation then becomes:

$$\bar{\phi}^{**}{}_4(s) = \frac{F_4(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{E_4(s)B(s)}{T(s)C(s)Z^-(s)}\bar{u}(s) + \frac{I_4(s)}{T(s)C(s)} \tag{11}$$

where

$$I_4(s) = \frac{E^D{}_4(s)C(s) - E_4(s)D(s)}{Z_T^-(s)} \tag{12}$$

with corresponding error

$$\bar{e}^{\star\star}_4(s) = \frac{E_4(s)}{Z^-(s)}\bar{v}(s) + \frac{E^D_4(s)}{Z^-(s)} \tag{13}$$

2.7. LINEAR-IN-THE-PARAMETERS FORM

One particular structure which can be used for realising the emulators of this section is the linear-in-the-parameters form. In transfer function form, each emulator can be written as

$$\bar{\phi}^{\star\star}(s) = \frac{G(s)}{C_T(s)}\bar{u}_z(s) + \frac{F(s)}{C(s)}\bar{y}_z(s) + \frac{I(s)}{C_T(s)} \tag{1}$$

where

$$\bar{\phi}^{\star\star}(s) = \begin{matrix} \bar{\phi}^{\star\star}_1(s) \\ \bar{\phi}^{\star\star}_2(s) \\ \bar{\phi}^{\star\star}_3(s) \\ \bar{\phi}^{\star\star}_4(s) \end{matrix} \quad \text{according to context} \tag{2}$$

and

$$\frac{G(s)}{C(s)Z^{-+}(s)} \triangleq \frac{E(s)B(s)}{C(s)Z^-(s)} \tag{3}$$

with common factors of $Z^-(s)$ and $B(s)$ cancelled out. In the case of $\bar{\phi}^{\star\star}_4(s)$, using equations 2.6.6,

$$C_T(s) = T(s)C(s) \tag{4}$$

otherwise

$$C_T(s) = C(s) \tag{5}$$

The filtered signals $\bar{u}_z(s)$ and $\bar{y}_z(s)$ are given, in the case of $\bar{\phi}^{**}{}_1(s)$, by

$$\bar{u}_z(s) \triangleq e^{-sT}\,\bar{u}(s);\quad \bar{y}_z(s) \triangleq \bar{y}(s) \tag{6}$$

in the case of $\bar{\phi}^{**}{}_2(s)$ by

$$\bar{u}_z(s) \triangleq \frac{e^{-sT}}{Z^{-+}(s)}\bar{u}(s);\quad \bar{y}_z(s) \triangleq \frac{1}{Z^{+}(s)}\bar{y}(s) \tag{7}$$

and in the case of $\bar{\phi}^{**}{}_3(s)$ and $\bar{\phi}^{**}{}_4(s)$ by

$$\bar{u}_z(s) \triangleq \frac{1}{Z^{-+}(s)}\bar{u}(s);\quad \bar{y}_z(s) \triangleq \frac{1}{Z^{+}(s)}\bar{y}(s) \tag{8}$$

This emulator equation may be rewritten as

$$\phi^{**}(t) = \underline{X}_e^{T}(t)\underline{\theta}_e \tag{9}$$

where the <u>data vector</u> $\underline{X}_e(t)$ and the <u>parameter vector</u> $\underline{\theta}_e$ are given, in Laplace transform terms, by

$$\bar{\underline{X}}_e(s) \triangleq \begin{vmatrix} \bar{\underline{X}}_u(s) \\ \bar{\underline{X}}_y(s) \\ \bar{\underline{X}}_i(s) \end{vmatrix};\quad \underline{\theta}_e = \begin{vmatrix} \underline{\theta}_u \\ \underline{\theta}_y \\ \underline{\theta}_i \end{vmatrix} \tag{10}$$

Where

$$\bar{\underline{X}}_u(s) = \frac{1}{C(s)}\begin{vmatrix} s^n \\ s^{n-1} \\ . \\ 1 \end{vmatrix}\bar{u}_z(s);\quad \bar{\underline{X}}_y(s) = \frac{1}{C(s)}\begin{vmatrix} s^n \\ s^{n-1} \\ . \\ 1 \end{vmatrix}\bar{y}_z(s) \tag{11}$$

$$\bar{\underline{X}}_i(s) = \frac{1}{C(s)}\begin{vmatrix} s^n \\ s^{n-1} \\ . \\ 1 \end{vmatrix} \tag{12}$$

and θ_e is given by

$$\theta_u = \begin{vmatrix} q_0 \\ q_1 \\ . \\ q_n \end{vmatrix}; \quad \underline{\theta}_y = \begin{vmatrix} f_0 \\ f_1 \\ . \\ f_n \end{vmatrix}; \quad \underline{\theta}_i = \begin{vmatrix} i_0 \\ i_1 \\ . \\ i_n \end{vmatrix} \tag{13}$$

The vectors $\bar{\underline{X}}_u(s)$, $\bar{\underline{X}}_y(s)$ and $\bar{\underline{X}}_i(s)$ are the Laplace transforms of vectors in controllable form (see section 1.6). The time-domain versions may therefore be computed from the differential equations 1.6.1.

This particular form provides a convenient means for implementing an emulator. In particular, the data vector $\underline{X}_e(t)$ is clearly distinguished from the parameter vector $\underline{\theta}_e$. This form will used in chapter 6 when self-tuning emulators are discussed.

References

1. Smith, O.J.M., "A controller to overcome dead-time," ISA transactions, vol. 6, no. 2, pp. 28-33, 1959.

2. Kalman, R.E., "A new approach to linear filtering and prediction problems," ASME Journal of Basic Engineering, vol. 82, pp. 35-45, 1960.

3. Luenberger, D.G., "Observing the state of a linear system," IEEE Trans., vol. MIL-*, 1964.

4. Kwakernaak, H, and Sivan, R.,, Linear optimal control systems, Wiley, 1972.

5. Joseph, B., Brosilow, C.B., and Tong, M., "Inferential control of processes: Parts I-III," AIChE Journal, vol. 25, pp. 485-509, 1978.

6. Parrish,J.R. and Brosilow, C.B., "Inferential control algorithms," Automatica, vol. 21, no. 5, pp. 527-538, 1985.

7. MacLane, S. and Birkhoff, G., Algebra, Macmillan, New York, 1967.

8. Clarke, D.W. and Gawthrop, P.J., "Self-tuning controller," Proceedings IEE, vol. 122, no. 9, pp. 929-934, 1975.

9. Clarke, D.W. and Gawthrop, P.J., "Self-tuning control," Proceedings IEE, vol. 126, no. 6, pp. 633-640, 1979.

10. Gawthrop, P.J. and Clarke, D.W., "Hybrid self-tuning control and its interpretation," Proceedings of 3rd IMA Conference on Control Theory, Academic Press., 1980.

11. Kucera, V., Discrete linear control: The polynomial equation approach., Wiley, Prague, 1979.

12. Kailath, T., Linear Systems, Prentice-Hall, 1980.

13. Astrom, K.J. and Wittenmark, B., Computer controlled systems, Prentice Hall, 1984.

14. Marshall, J.E., Control of time-delay systems, Peter Peregrinus Ltd., 1979.

15. Astrom, K.J., Introduction to stochastic control theory, Academic Press, New York, 1970.

16. Kailath, T., Lectures on Weiner and Kalman Filtering, Springer, 1981.

17. Whittle, P., Prediction and regulation, English Universities Press, 1963.

18. Yaglom, A.M., Stationary random functions, Dover, 1973.

19. Solodovnikov, V.V., Introduction to the statistical dynamics of automatic control systems, Dover, 1960.

CHAPTER 3
Emulator-Based Control

Aims. To introduce and illustrate the use of emulators in a feedback loop. To introduce the notional feedback loop and its use in investigating the closed-loop properties of the emulator-based control. To show that well-known control strategies such as model-reference, pole-placement and predictive control are limiting cases of particular emulators in a feedback loop. To discuss the choice of emulator-based control design parameters.

3.1. INTRODUCTION

Self-tuning controllers are based on many different non-adaptive control design techniques. The purpose of this chapter is to present a selection of such design approaches in a unified fashion. The unifying concept is the emulator considered in the previous chapter. We shall see that, by incorporating such an emulator in the feedback path of an otherwise classical control scheme, many types of algorithms can be considered in a common framework. The classes of algorithms include: model-reference (pole/zero placement), pole placement, steady-state linear-quadratic, and predictive control.

An important concept to be covered is that of <u>control weighting</u> or <u>detuning</u> of control algorithms. This will be shown in a later chapter to be crucial in giving a robust adaptive algorithm.

3.2. THE CONTROL LAW

The single-input single-output feedback controllers considered in this book can all be written in a common form; as classical feedback controllers but with an <u>emulator</u> in the feedback path. The control law can be written in two equivalent forms:

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \bar{\phi}^{\star}(s)] \qquad (1)$$

and

$$\bar{\phi}^{\star}(s) + Q(s)\hat{u}(s) - R\bar{w}(s) = 0 \qquad (2)$$

where

Symbol	Quantity
$\hat{u}(s)$	Control signal
$\bar{\phi}^{\star}(s)$	emulator output
$\bar{w}(s)$	setpoint
$Q(s)$	control weighting
$R(s)$	setpoint filter

$1/Q(s)$ and $R(s)$ are proper transfer functions. $\bar{\phi}^{\star}(s)$ is the <u>emulator output</u> corresponding to one of the emulators described in chapter 2. That is

$$\bar{\phi}^{\star}(s) = \begin{cases} \bar{\phi}^{\star}_{1}(s) \\ \bar{\phi}^{\star}_{2}(s) \\ \bar{\phi}^{\star}_{3}(s) \\ \bar{\phi}^{\star}_{4}(s) \end{cases} \quad \text{according to context} \tag{3}$$

and can be written in transfer function form as in section 2.7 as

$$\bar{\phi}^{\star}(s) = \frac{G(s)}{C(s)}\bar{u}_{z}(s) + \frac{F(s)}{C(s)}\bar{y}_{z}(s) \tag{4}$$

This would typically be implemented in linear-in-the-parameters form as in section 2.7.

Alternatively, we could use $\bar{\phi}^{\star\star}(s)$ in place of $\bar{\phi}^{\star}(s)$. However, in this chapter, we shall ignore the effect of initial conditions; that is, we concentrate on the system <u>setpoint response</u> and the system <u>disturbance response</u>. This emulator-based control law is given in Fig 3.2.1.

<u>Limiting the control signal</u>

In many contexts, it is appropriate to limit the control action of a feedback controller, typically to avoid actuator saturation. This can readily be done here by interposing a suitable non-linearity between the 1/Q(s) transfer function and the control signal as follows:

$$\bar{u}^{\star}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \bar{\phi}^{\star}(s)] \tag{5}$$

$$\hat{u}(t) = \text{Sat}\{u^{\star}(t)\} \tag{6}$$

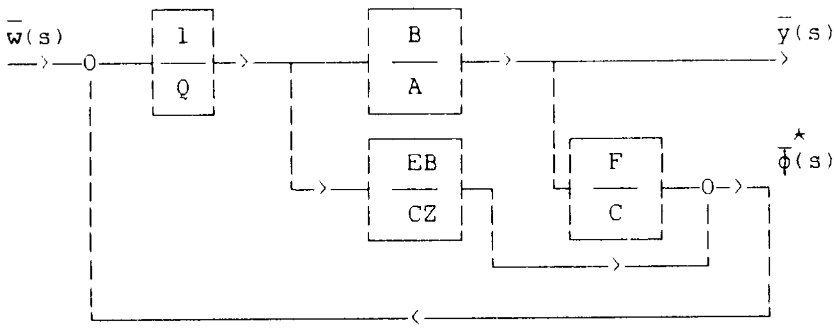


Figure 3.2.1 The Emulator in the Feedback Loop.

where "Sat" indicates the appropriate non-linear saturation function.

The crucial point here is that the emulator should operate on the signal $\hat{u}(t)$ reaching the plant, not the signal $\bar{u}^*(s)$ before the saturation. See[1,2] for a discussion in the discrete-time context.

3.3. THE NOTIONAL FEEDBACK LOOP

To obtain the properties of emulator-based feedback control laws, the idea of a notional feedback loop is introduced in this section. To obtain general equations, we consider the emulator for $\bar{\phi}_3(s)$ which includes all the other emulators as special cases. Recall that:

$$\bar{\phi}_3(s) = \bar{\phi}_3^*(s) + \bar{e}_3^*(s) \tag{1}$$

and that

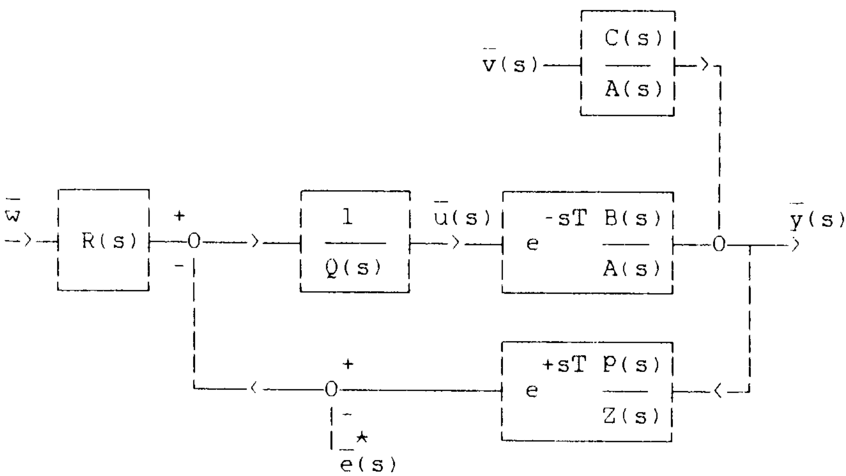
$$\bar{\phi}_3(s) = e^{sT}\frac{P(s)}{Z(s)} \tag{2}$$

In this chapter, the controller output is assumed to be the nominal system input:

$$\bar{u}(s) = \hat{u}(s) \qquad (3)$$

The consequences of this assumption being false are examined in chapter 4.

Combining these equations gives the block diagram of Figure 3.3.1. This <u>notional feedback system</u> provides an easier way of deriving system equations than using Figure 3.2.1.



<u>Figure 3.3.1 The notional feedback system</u>

This block diagram is a correct representation of the preceding equations; and is useful for giving insight into the control laws and their relationships. However, it does not, by itself, give any information about sensitivity to modelling error, as the error equation 3.3.1 assumes no modelling error. We will return to the study of sensitivity in the next chapter, but for the moment we assume no modelling error.

As discussed by Horowitz[3,4], controllers for single-input single-output systems have <u>two degrees of freedom</u> available to the designer: a transfer function multiplying the setpoint w and one multiplying the measured system output y. The controllers considered in this chapter are no exception to this rule: Figure 3.3.1 is one of the many ways of representing such a controller. In later chapters, the non adaptive emulator generating $\bar{\phi}^{*}(s)$ will be replaced by a self-tuning version. In such circumstances, the transfer function $\frac{P(s)}{Z(s)}$ becomes a <u>third degree of freedom</u> available to the designer. This idea is pursued further in chapter 8.

Combining the equations displayed in Figure 3.3.1, the following expressions for closed-loop system quantities are obtained:

<u>Notional loop-gain</u>

$$L(s) \triangleq \frac{1}{Q(s)} \frac{P(s)B(s)}{Z(s)A(s)} \tag{4}$$

This is the product of all the transfer functions within the loop displayed in Figure 3.3.1.

<u>Closed-loop system output</u>

$$\bar{y}(s) = \frac{L(s)}{1+L(s)} e^{-sT} \frac{Z(s)}{P(s)}[R(s)\bar{w}(s) + \bar{e}^{*}(s)] \tag{5}$$

$$+ \frac{1}{1+L(s)} \frac{C(s)}{A(s)}\bar{v}(s)$$

$$= e^{-sT} \frac{B(s)Z(s)}{P(s)B(s) + Q(s)Z(s)A(s)}[R(s)\bar{w}(s) + \bar{e}^{\star}(s)] \quad (6)$$

$$+ Q(s) \frac{Z(s)C(s)}{P(s)B(s) + Q(s)Z(s)A(s)}\bar{v}(s)$$

Closed-loop system input

$$\bar{u}(s) = \frac{L(s)}{1+L(s)} \frac{Z(s)A(s)}{P(s)B(s)} \bar{z}(s) \quad (7)$$

where the equivalent setpoint $\bar{z}(s)$ is given by

$$\bar{z}(s) = R(s)\bar{w}(s) - e^{sT} \frac{P(s)C(s)}{Z(s)A(s)}\bar{v}(s) + \bar{e}^{\star}(s) \quad (8)$$

$$= R(s)\bar{w}(s) + [\frac{E(s)}{Z^{-}(s)} - e^{sT} \frac{P(s)C(s)}{Z(s)A(s)}]\bar{v}(s)$$

$$= R(s)\bar{w}(s) - \frac{F(s)}{AZ^{+}(s)}\bar{v}(s) \quad (9)$$

This equivalent setpoint may be regarded as the net influence of disturbances and setpoint on the control signal referred to the same point on the block diagram as the filtered setpoint $R(s)\bar{w}(s)$.

It will sometimes be convenient to decompose this equivalent setpoint into the part $\bar{e}^{\star}(s)$ due to the emulation error and the rest as

$$\bar{z}(s) = \tilde{z}(s) + \bar{e}^{\star}(s) \quad (10)$$

where

$$\tilde{z}(s) = R(s)\bar{w}(s) - e^{sT} \frac{P(s)C(s)}{Z(s)A(s)}\bar{v}(s) \quad (11)$$

The closed-loop characteristic equation

Before taking a detailed look at the various controller options available, these two equations can be used to give an overview of the aims and characteristics of the emulator-based control laws. The following commments can be made:

1. As discussed in section 2.3, an important special case is to choose

$$B(s) = B^{+}(s)B^{-}(s); \; Z(s) = Z^{+}(s)Z^{-}(s); \; Z^{-}(s) = B^{-}(s) \tag{12}$$

In this case, the nominal loop-gain L(s) is

$$L(s) = \frac{P(s)B^{+}(s)}{Q(s)Z^{+}(s)A(s)} \tag{13}$$

2. The stability of the closed-loop system is dependent on the zeros of the transfer function 1+L(s); thus the equation

$$P(s)B^{+}(s) + Q(s)A(s)Z^{+}(s) = 0 \tag{14}$$

must have no zeros with positive real parts.

Parallel transfer functions

An alternative viewpoint, based on[5], is to regard Q(s) as a transfer function in parallel with the system. Define

$$\bar{\phi}_Q(s) \triangleq \bar{\phi}(s) + Q(s)\bar{u}(s) \tag{15}$$

as the auxiliary output corresponding to the system in Figure 3.3.2 comprising Q(s) in parallel with P(s) cascaded with the system. The transfer function of the augmented plant relating $\bar{\phi}_Q(s)$ to $\bar{u}(s)$ is

$$\frac{P(s)B^{+}(s) + Q(s)A(s)Z^{+}(s)}{A(s)Z^{+}(s)} \tag{16}$$

The zeros of this augmented plant are precisely the roots of the characteristic equation (3.3.14).

The control law 3.2.1 may be rewritten as:

$$\bar{\Phi}_Q(s) = R(s)\bar{w}(s) + \bar{e}^{\star}(s) \tag{17}$$

In the absence of any disturbance ($\bar{e}^{\star}(s)=0$), this control law sets the auxiliary output $\bar{\Phi}_Q(s)$ exactly equal to the filtered setpoint $R(s)\bar{w}(s)$; this is only possible if the augmented plant is invertible. In particular, the augmented system must have stable zeros.

Thus Q(s) may be reinterpreted as a means of moving plant zeros to give an invertible augmented plant. A discussion along these lines (but in the discrete-time context) appears in[5,6] and[7].

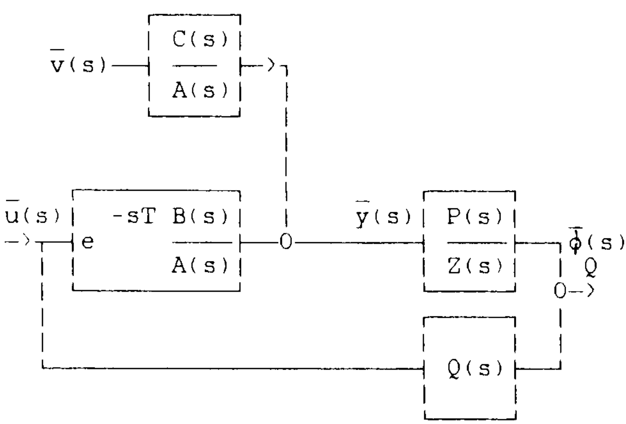


Figure 3.3.2 The auxiliary output

3.4. CHOOSING P(s) AND Z(s)

Let us first of all consider the case with no time delay (T=0), no control weighting (Q(s)=0) and no setpoint filter R(s):

$$Q(s) = 0;\ R(s) = 1;\ T = 0;\ B(s) = B^{+}(s)B^{-}(s); \qquad (1)$$

In addition Z(s) is chosen as

$$Z(s) = Z^{+}(s)Z^{-}(s);\ Z^{-}(s) = B^{-}(s) \qquad (2)$$

The closed loop equations then become:

Notional loop-gain

$$L(s) = \infty \qquad (3)$$

Closed-loop system output

$$\bar{y}(s) = \frac{Z(s)}{P(s)}[\bar{w}(s) + \bar{e}^{\star}(s)] \qquad (4)$$

The closed loop system output $\bar{y}(s)$ has two terms: the setpoint response $\frac{Z(s)}{P(s)}\bar{w}(s)$ and the disturbance response $\frac{Z(s)}{P(s)}\bar{e}^{\star}(s)$. Both terms are of the form of a $\frac{Z(s)}{P(s)}$ multiplied by a signal. Thus the closed-loop system output is determined by the reference-model $\frac{Z(s)}{P(s)}$. The reference model zeros are the roots of Z(s); the reference model poles are the roots of P(s).

The closed-loop transfer function generating the system output is stable iff P(s) has all zeros within the left-half s-plane.

As we would usually require that there be no steady-state offset due to the setpoint, we shall choose P(s) and Z(s) such that

P(s) design rule

$$P(0) = 1 \tag{5}$$

Z(s) design rule

$$Z^+(0) = Z^-(0) = 1 \tag{6}$$

Closed-loop system input

$$\bar{u}(s) = \frac{Z(s)A(s)}{P(s)B(s)}\,\bar{z}(s) = \frac{Z^+(s)A(s)}{P(s)B^+(s)}\,\bar{z}(s) \tag{7}$$

where the equivalent setpoint $\bar{z}(s)$ is given by

$$\bar{z}(s) = \bar{w}(s) - \frac{F(s)}{A(s)Z^+(s)}\,\bar{v}(s) \tag{8}$$

The closed-loop transfer function generating the system input is stable iff $P(s)B^+(s)$ has all zeros within the left-half s-plane.

Three special cases of this control strategy are

- Model-reference control
- Pole-placement control
- Steady-state linear-quadratic control

These will be treated in turn.

Model-reference control

Model-reference control is a special case of the above algorithm defined by

$$B^{-}(s) = Z^{-}(s) = 1 \tag{9}$$

thus the closed-loop system model is not related to the open-loop system. It is clear that the control signal will only be stable if

$$B(s) \text{ is stable} \tag{10}$$

Example

Consider the example of section 2.2 where the system is given by

$$A(s) = s(s+1); B(s) = 1+0.1s \tag{11}$$

and the design polynomials by

$$P(s) = 1+0.5s;\; Z(s) = 1;\; C(s) = 1+0.5s \tag{12}$$

As in section 2.2, the corresponding emulator (without initial conditions) is:

$$\bar{\phi}^{\star}(s) = \bar{\phi}^{\star}_{1}(s) = \frac{0.25(1+0.1s)}{1+0.5s}\bar{u}(s) + \frac{1+0.75s}{1+0.5s}\bar{y}(s) \tag{13}$$

Combining this with the control law 3.2.1 with $Q(s)=0$ and $R(s)=1$,

$$\bar{\phi}^{\star}(s) = R\bar{w}(s) \tag{14}$$

gives:

$$\hat{u}(s) = -4\,\frac{1+0.75s}{1+0.1s}\bar{y}(s) + \frac{1+0.5s}{1+0.1s}\bar{w}(s) \tag{15}$$

This is of the classical two degree of freedom form[3] and the transfer function relating $\hat{u}(s)$ to $\bar{y}(s)$ is of the standard phase-advance form of classical control to be found in any elementary textbook, for example[8].

Note that the system zero at s=-10 is cancelled by the controller. This is an inevitable result of specifying a reference model with different zeros to those of the open loop system.

□

Pole-placement control

Pole-placement control is a special case of the above algorithm defined by

$$B^{-}(s) = Z^{-}(s) = B(s);\ Z^{+}(s) = 1 \quad (16)$$

thus the closed-loop system model is related to the open-loop system; the zeros of the open-loop system $\frac{B(s)}{A(s)}$ are identical to those of the closed-loop system $\frac{Z(s)}{P(s)}$. It is clear that the control signal will be stable even if B(s) is not.

Example

Consider the example of section 2.4 where the system is given by

$$A(s) = s(s+1); B(s) = 1-s \quad (17)$$

Note that the system has a zero at s=1 with positive real part. This can be regarded as an integrator in series with a time delay of 2 units represented by the (very crude) first order Pade approximation (section 2.6):

$$e^{-2s} \simeq \frac{1-s}{1+s} \quad (18)$$

The design polynomials in the second example of section 2.4 are

$$P(s) = 1+s+0.25s^2; \; Z(s) = Z^-(s) = 1-s; \; C(s) = 1+0.5s \tag{19}$$

Note that $Z^-(s) = B(s)$ in this case to remove the offending zero. As in section 2.4, the corresponding emulator (without initial conditions) is:

$$\bar{\phi}^*(s) = \frac{0.125s+1.562}{0.5s+1}\bar{u}(s) + \frac{0.938s+1}{0.5s+1}\bar{y}(s) \tag{20}$$

Combining this with the control law 3.2.1 with $Q(s)=0$ and $R(s)=1$,

$$\bar{\phi}^*(s) = \bar{w}(s) \tag{21}$$

gives:

$$\hat{u}(s) = -0.6402\,\frac{1+0.938s}{1+0.0800s}\bar{y}(s) + \frac{1+0.5s}{1+0.1s}\bar{w}(s) \tag{22}$$

This is of the classical two degree of freedom form[3] and the transfer function relating $\hat{u}(s)$ to $\bar{y}(s)$ is of the standard phase-advance form of classical control to be found in any elementary textbook, for example[8].

Note that the system zero at s=1 is <u>not</u> cancelled by the controller. The controller has lower steady-state gain and larger phase advance than the model-reference controller designed in section 2.2 for the system with a zero at -0.1.

□

<u>Steady-state linear-quadratic control</u>

This is not the place to go into a full discussion of linear quadratic control[9,10,11]. Roughly speaking, the essential result is that linear quadratic control is a special case of pole-placement control where P(s) is obtained as the stable spectral factor of

$$P(s)P(-s) = B(s)B(-s) + \lambda A(s)A(-s) \tag{23}$$

with the restriction that B(s) and A(s) must have no common factors[12,9].

3.5. CHOOSING R(s)

From equation 3.3.5 or 3.3.6 it follows that R(s) merely acts as a setpoint filter. Thus if $R \neq 1$, we can replace $\bar{w}(s)$ by $\bar{w}_R(s)$ in the previous section where

$$\bar{w}_R(s) = R(s)\bar{w}(s) \tag{1}$$

R(s) has no effect on the feedback loop itself; it merely acts as another degree of freedom for manipulating the setpoint response without affecting the system loop-gain or response to disturbances.

The importance of R(s) lies in the second degree of freedom it gives in manipulating closed-loop performance.

Model-reference control

If the model-reference controller of section 3.3 is extended so that $R \neq 1$, then the resultant closed-loop setpoint response is determined by

$$\bar{y}(s) = \frac{R(s)}{P(s)}\bar{w}(s) \tag{2}$$

In this equation, R(s) and $\frac{1}{P(s)}$ play identical roles, and as far as the setpoint response is concerned the following design choices are equivalent:

$$\frac{1}{P(s)} = \text{desired model};\ R(s) = 1 \tag{3}$$

and

$$\frac{1}{P(s)} = 1; \; R(s) = \text{desired model} \tag{4}$$

However, when disturbances and sensitivity to parameter variation are considered, these two approaches are very different. Indeed the latter approach leads to an infinite gain controller; thus choosing $P(s) = 1$ is not practical. (See[13] for a discussion of this point in a discrete-time context).

In practice then, both $P(s)$ and $R(s)$ have their uses; in particular $\frac{R(s)}{P(s)}$ specifies the setpoint response, whereas $P(s)$ alters the disturbance response and closed-loop sensitivity.

As we normally require a unity steady-state system gain from setpoint to output we impose the

$$R(s) = 1 \tag{5}$$

3.6. CHOOSING Q(s)

It seems intuitively obvious (and we shall prove this later) that it is not a good idea to have a system with loop gain $L(s) = \infty$. Of course, this is only a notional loop gain and the system is not implemented in this form. But nevertheless, the implication of $L(s) = \infty$ is that we ask for exact matching of our desired closed loop-system at all frequencies. It is clearly unnecessary to specify system performance precisely at high frequencies; we shall see later, in the self-tuning context, that it is also very unwise.

We have already noted (equation 3.3.14) that the stability of the closed-loop control system is dependent on the roots of the characteristic equation:

$$P(s)B^{+}(s) + Q(s)A(s)Z^{+}(s) = 0 \tag{1}$$

We emphasise that this equation does <u>not</u> necessarily give rise to a stable closed-loop system. It has been suggested[5,14,2] in the discrete-time context and in the special case where $B^{+}(s) = B(s)$ that $Q(s)\neq 0$ can be used to give stability when B(s) is not stable. In this book, we do not regard this as being a very useful approach to stabilise a nominal system with unstable zeros: the zero cancelling (pole-placement) approach is more appropriate. We believe that the role of Q(s) is make a feedback controller more robust in the face of neglected dynamics.

If the notional feedback system is stable, then for those frequencies ω where L(jω) is large the ratio of the closed-loop output y to the set point w is:

$$\frac{\bar{y}(j\omega)}{\bar{w}(j\omega)} \cong e^{-j\omega T}\frac{Z(j\omega)}{P(j\omega)}R(jw) \tag{2}$$

Under such circumstances, the closed-loop setpoint frequency response approximates that of the <u>reference model</u>:

$$e^{-sT}\frac{Z(s)}{P(s)}R(s) \tag{3}$$

In particular, if Q(s)=0 (for all s), exact model matching is achieved for all frequencies; and if Q(0)=0 this is achieved at zero frequency.

To give zero weighting at zero frequency we impose the

<u>Q(s) design rule</u>

$$Q(0) = 0 \tag{4}$$

Thus Q(s) will be regarded as a device for reducing the exact matching requirement at high frequency. The use of

$Q(s) \neq 0$ leads to detuned or control-weighted versions of the control laws derived with $Q(s) = 0$. In particular, we now have three control-weighted algorithms:

- Control weighted model-reference control
- Control weighted pole-placement control
- Control weighted linear-quadratic control

In practice, we would usually require exact model matching at zero frequency to avoid steady-state offset. In such circumstances we would choose $Q(s)$ such that

$$Q(0) = 0 \tag{5}$$

3.7. CHOOSING T

In the above discussion, we have implicitly equated the "T" appearing in the emulator with "T" corresponding to the assumed system time-delay. This is in fact quite general as in a later chapter we shall discuss the effect of incorrect system modelling.

The crucial result of the predictive (e^{sT}) component of the emulator is to eliminate the system time-delay from both the nominal loop-gain and the closed-loop characteristic equation. This idea was proposed by Smith[15] and is discussed in detail in the following section. The purely predictive emulator of section 2.5 is in fact a generalised version of that proposed by Smith.

3.8. SMITH'S PREDICTOR

The idea that control of systems with time-delay can be simplified by making use of a predictor was suggested by Smith in the late '50s[15,16]. His predictor can be described by the following Figure. Like the emulator

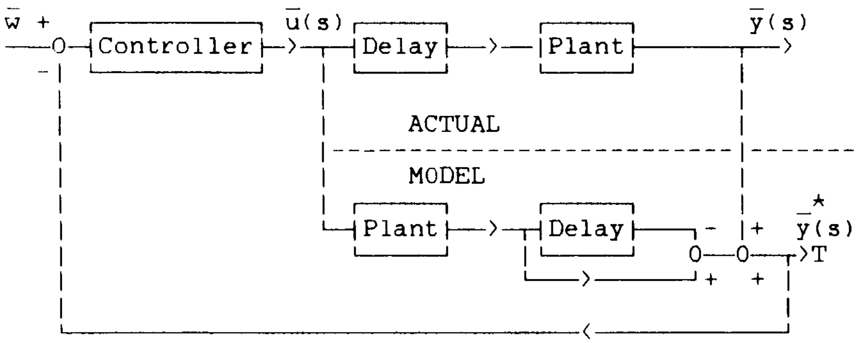


Figure 3.8.1 Smith's Predictor

discussed in the previous section, Smith's predictor can be regarded as a method of realising the unrealisable transfer function e^{sT}. In particular, it generates the quantity $\bar{y}^*_T(s)$ given by

$$\bar{y}^*_T(s) = \bar{y}(s) + [1 - e^{-sT}]\frac{B(s)}{A(s)}\bar{u}(s) \tag{1}$$

In the absence of disturbances, substitution of the system equation gives

$$\bar{y}^*_T(s) = \frac{B(s)}{A(s)}\bar{u}(s) = e^{sT}\,\bar{y}(s) = \bar{y}_T(s) \tag{2}$$

where $\bar{y}_T(s)$ is the Laplace transform of $y_T(t)=y(t+T)$. That is, in the absence of disturbances, the effect of the Smith predictor is the same as including an inverse time delay (e^{sT}) in series with the system output.

How does this relate to the emulators derived here? The purely predictive emulator of section 2.5 is in fact a generalised version of that proposed by Smith. To see this we take the special case

$$P(s) = 1;\ Z(s) = 1;\ C(s) = A(s) \tag{3}$$

and

$$Q(s) = \text{inverse cascade compensator} \tag{4}$$

The decomposition identity can be written as

$$1 = E_T(s) + e^{-sT}\ \frac{F_T(s)}{A(s)} \tag{5}$$

If, in addition, we break the rule that $\frac{F(s)}{A(s)}$ is strictly proper, this my be solved by

$$E_T(s) = 1 - e^{-sT}\ ;\ F_T(s) = A(s) \tag{6}$$

giving the Smith predictor.

Smith's predictor has the advantage that it can be implemented with rational transfer functions and a pure delay; it has the disadvantage that the predictor poles are identical to the system poles, giving poor transient response unless the open-loop system poles have fast time constants.

3.9. CHOOSING C(s)

At first sight, the polynomial C(s) is part of the system; but, as discussed in section 1.8, this is not so as $\bar{v}$ (s) is not specified in detail. To see this, set

$$\bar{v}(s) = \frac{C'(s)}{C(s)}\bar{v}'(s) \tag{1}$$

where C'(s) is a polynomial of the same degree as C(s). An alternative system equation to 1.9.1 is then given by replacing C(s) by C'(s) and $\bar{v}(s)$ by $\bar{v}'(s)$ to give

$$\bar{y}(s) = \frac{B(s)}{A(s)}\bar{u}(s) + \frac{C'(s)}{A(s)}\bar{v}'(s) + \frac{D(s)}{A(s)} \tag{2}$$

Using this equation to deduce the closed-loop system equations gives

Closed-loop system output

$$\bar{y}(s) = \frac{L(s)}{1+L(s)} [e^{-sT} \frac{Z(s)}{P(s)}(R(s)\bar{w}(s) + \frac{C'(s)}{C(s)}\bar{e}^{*}(s))] \quad (3)$$

$$+ \frac{1}{1+L(s)} \frac{C'(s)}{A(s)}\bar{v}'(s)$$

$$= \frac{B(s)Z(s)}{P(s)B(s) + Q(s)Z(s)A(s)}[e^{-sT} R(s)\bar{w}(s) + \frac{C'(s)}{C(s)}\bar{e}^{*}(s)] \quad (4)$$

$$+ Q(s) \frac{Z(s)C'(s)}{P(s)B(s) + Q(s)Z(s)A(s)}\bar{v}'(s)$$

Closed-loop system input

$$\bar{u}(s) = \frac{L(s)}{1+L(s)} \frac{Z(s)A(s)}{P(s)B(s)} \bar{z}(s) \quad (5)$$

where the equivalent setpoint $\bar{z}(s)$ is given by

$$\bar{z}(s) = R(s)\bar{w}(s) - \frac{C'(s)F(s)}{C(s)A(s)Z^{+}(s)}\bar{v}(s) \quad (6)$$

It follows that the design polynomial C(s) affects the poles and zeros of the closed-loop response to disturbances, but has no effect on the setpoint response. It plays a similar role to the observer pole-polynomial in state-space theory[17,9].

To give unique solutions to the emulator design, we usually impose the

C(s) design rule

$$C(0) = 1 \tag{7}$$

3.10. INTEGRAL ACTION

As stated in[18,19], the large number of PI (proportional + integral) and PID (proportional + integral + derivative) controllers used routinely for process control applications may be regarded as experimental evidence for their usefulness.

As PI and PID controllers are so common, there must be something about the dynamics of many systems which makes such control appropriate. It follows that it should not be necessary to force an adaptive controller to have a PI or PID structure, but rather this structure should arise naturally from reasonable assumptions about the dynamics of the controlled process. It is shown in this section that this is indeed so: suitable modelling of non-zero mean disturbances leads to an algorithm with integral action, and the additional assumption of a first (second) order system gives rise to a PI (PID) controller.

This approach of letting the integral action arise naturally from the specification of a suitable disturbance model rather than forcing integral action into the controller distinguishes the algorithms of this book from some previous methods. As will be shown, this approach automatically removes offsets from both the controller and the estimator.

An extensive discussion of the method (but resticted to the model-reference case) appears in[19]. Details of the self-tuning version appear in chapter 6.

Two common forms of disturbance in control systems are constants and piecewise constant signals with random jumps. As discussed in sections 1.8 and 1.9, each form of disturbance corresponds to a transfer function

$$\frac{B^t(s)}{A^t(s)} = \frac{B^f(s)}{A^f(s)} = \frac{k}{s} \qquad (1)$$

the former corresponding to the initial condition response of an integrator, the latter to the forced response of an integrator to a random sequence of impulses. In either case, the results of section 1.9 indicate that A(s) and B(s) will have a common factor s; as C(s) is chosen, this common factor need not appear in C(s). This gives rise to the following design rule:

<u>PI design rule 1</u>

A(s) and B(s) have a common root at s=0:

$$A(s) = A_0(s)s; \; B(s) = B_0(s)s \qquad (2)$$

In addition, we make the following design rule:

<u>PI design rule 2</u>

$Z^-(s)$ has no root at s=0: $Z^-(0) \neq 0$. This implies that, in this case, $B^+(s)$ contains the factor s in $B(s) = sB_0(s)$.

To see the implications of these design rules consider the defining identity leading to $\bar{\Phi}_3^*(s)$ (equation 2.5.33):

$$e^{sT} \frac{P(s)C(s)}{Z(s)A(s)} = e^{sT} \frac{E_3(s)}{Z^-(s)} + \frac{F_3(s)}{Z^+(s)A(s)} \qquad (3)$$

evaluated at s=0. As, by assumption, A(s) has a factor s and $Z^+(s)$ hasn't, it follows that:

$$F_3(0) = \frac{P(0)C(0)}{Z^-(0)} = \frac{P(0)C(0)}{Z(0)} = 1 \qquad (4)$$

Where the last equality follows from the P(s), Z(s) and C(s) design rules. Hence, in this case, $F_3(s)$ can be rewritten as

$$F_3(s) = 1+sF_{30}(s) \qquad (5)$$

Turning to equation 3 (2.5.33), $\bar{\phi}_3^{\star}(s)$ can be written as

$$\bar{\phi}_3^{\star}(s) = \frac{1+sF_{30}(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{sE_3(s)B_0(s)}{C(s)Z^-(s)}\bar{u}(s) \qquad (6)$$

PID control

As discussed in detail elsewhere[19,18] certain forms of assumed system give rise to PI and PID controllers. We give two examples based on the model-reference and pole-placement examples given in previous sections.

Example (Model-reference PID)

Consider the example of section 2.2 and section 3.4 but a cancelling s term is included to model offset. The augmented system is given by

$$A(s) = s^2(s+1); B(s) = s(1+0.1s) \qquad (7)$$

The design polynomials are as before except that C(s) is now second order:

$$P(s) = 1+0.5s; \; Z(s) = 1; \; C(s) = (1+0.5s)^2 \qquad (8)$$

As in section 2.2, the corresponding emulator (without initial conditions) is:

$$\bar{\phi}^{\star}(s) = \bar{\phi}^{\star}_{1}(s) = \frac{0.125s(1+0.1s)}{(1+0.5s)^2}\bar{u}(s) + \frac{1+1.5s+0.625s^2}{(1+0.5s)^2}\bar{y}(s) \quad (9)$$

Combining this with the control law 3.2.1 with $Q(s)=0$ and $R(s)=1$

$$\bar{\phi}^{\star}(s) = R\bar{w}(s) \quad (10)$$

gives:

$$\hat{u}(s) = \frac{8}{1+0.1s}\left[\frac{\bar{w}(s)-\bar{y}(s)}{s} + (\bar{w}(s)-1.5\bar{y}(s)) + s(0.25\bar{w}(s)-0.625\bar{y}(s))\right] \quad (11)$$

This has the structure of a PID controller with filtering and modified proportional and derivative setpoint terms.

□

Example (Pole-placement PID)

Consider the example of section 2.4 and section 3.4 but a cancelling s term is included to model offset. The augmented system is given by

$$A(s) = s^2(s+1); B(s) = s(1-s) \quad (12)$$

The design polynomials are as before except that $C(s)$ is now second order:

$$P(s) = (1+0.5s)^2;\ Z(s) = 1-s;\ C(s) = (1+0.5s)^2 \quad (13)$$

As in section 2.4, the corresponding emulator (without initial conditions) is:

$$\bar{\phi}^{\star}(s) = \bar{\phi}^{\star}_{2}(s) = \frac{s(2.460+0.0625s)}{(1+0.5s)^2}\bar{u}(s) + \frac{1+3.0s+2.0313s^2}{(1+0.5s)^2}\bar{y}(s) \quad (14)$$

Combining this with the control law 3.2.1 with $Q(s)=0$ and $R(s)=1$

$$\bar{\phi}^{\star}(s) = R\bar{w}(s) \tag{15}$$

gives:

$$\hat{u}(s) = \frac{0.405}{1+0.0254s}\left[\frac{\bar{w}(s)-\bar{y}(s)}{s}\right.$$

$$\left. + (\bar{w}(s)-3.00\bar{y}(s)) + s(0.250\bar{w}(s)-2.033\bar{y}(s))\right] \tag{16}$$

This has the structure of a PID controller with filtering and modified proportional and derivative setpoint terms. Note that the proportional gain is lower, and the derivative gain much higher, than for the model-reference example - the system is much harder to control.

□

3.11. A DETUNED MODEL-REFERENCE CONTROLLER

In the sequel (chapters 7&8 in particular), we shall analyse a particular form of detuned model-reference controller, introduced in[20].

This controller is defined by the Table:

Parameter	Value
$P(s)$	Desired closed loop pole polynomial
$Z^+(s)$	1
$Z^-(s)$	$P(\varepsilon s)$; $0<\varepsilon<1$
$Q(s)$	$\frac{q(s)}{Z^-(s)}$ $\deg(q) = \deg(P)$
$C(s)$	Desired disturbance closed-loop poles
T	0

Note that $Z^-(s)$ is not used for zero cancellation here.

This particular emulator based controller is unusual in that the notional feedback loop is realisable. At first sight, it would seem that there is no purpose to be served in implementing the emulator or its self-tuning version. However, as discussed in detail in chapter 8, the high-frequency gain of the transfer function $\frac{P(s)}{Z(s)}$ is:

$$\frac{P(\infty)}{Z(\infty)} = \frac{P(\infty)}{P(\varepsilon\infty)} = \frac{1}{\varepsilon^n};\ n \triangleq \deg(P) \tag{1}$$

This may be excessive for small ε and lead to amplification of unwanted high-frequency sensor noise. The replacement of the realisable transfer function by a suitable emulator can remove this undesirable effect - see chapter 8 for a detailed discussion of the relative merits of implementing the notional feedback loop and the self-tuning emulator.

The corresponding closed-loop system is defined by:

Notional loop-gain

$$L(s) = \frac{P(s)B(s)}{q(s)A(s)} \tag{2}$$

Closed-loop system output

$$\bar{y}(s) = \frac{L(s)}{1+L(s)}\left[\frac{P(\varepsilon s)}{P(s)}(R(s)\bar{w}(s) + \bar{e}^{\star}(s))\right] \tag{3}$$

$$+ \frac{1}{1+L(s)}\frac{C(s)}{A(s)}\bar{v}(s)$$

$$= \frac{B(s)Z(s)}{P(s)B(s) + q(s)A(s)}[R(s)\bar{w}(s) + \bar{e}^{\star}(s)] \tag{4}$$

$$+ q(s)\frac{C(s)}{P(s)B(s) + q(s)A(s)}\bar{v}(s)$$

Closed-loop system input

$$\bar{u}(s) = \frac{L(s)}{1+L(s)}\frac{Z(s)A(s)}{P(s)B(s)}\bar{z}(s) \tag{5}$$

This controller can be thought of as an approximate model-reference controller in the sense that as $Q(s)\rightarrow 0$ the control law approaches that discussed in the model-reference section. The importance of these particular algorithms is that can be made into an implicit self-tuning controller with global robustness properties. It is a continuous-time generalisation of the discrete-time generalised minimum variance control law[2,5]

Example

Consider the example in section 2.4, where the system is given by

$$A(s) = s(s+1); B(s) = 2s \tag{6}$$

Thus the system is now first order and has a constant disturbance. This example is to be used later to investigate robustness. The example is that of Rohrs[21]. The corresponding design parameters (see chapter 7) are

$$P(s) = 1+0.3s;\; C(s) = 1+0.3s \tag{7}$$

Choosing $\varepsilon = 0.1$ then gives

$$Z(s) = Z^-(s) = 1+0.03s \tag{8}$$

Using the results from the example of section 2.4 with

$$P(s)C(s) = (1+0.3s)^2 = 1 + 0.6s + 0.09s^2;\; z = 0.03 \tag{9}$$

gives

$$E_z(s) = E_{20} + 0.6E_{21} + 0.09E_{22} \tag{10}$$

$$= \frac{1}{1-z}(z^2 - 0.6z + 0.09) = 0.07515$$

and

$$F_z(s) = F_{20} + 0.6F_{21} + 0.09F_{22} \tag{11}$$

$$= 1 + \frac{s}{1-z}(-z + 0.6-0.09) = 1 + 0.4948s$$

The corresponding emulator is then

$$\bar{\phi}^*(s) = \bar{\phi}_z^*(s) = \frac{E_z(s)B(s)}{C(s)Z^-(s)}\bar{u}(s) + \frac{F_z(s)}{A(s)Z^+(s)} \tag{12}$$

$$= \frac{0.1503s}{(1+0.3s)(1+0.03s)}\bar{u}(s) + \frac{1+0.4948s}{1+0.3s}\bar{y}(s)$$

Combining this with the control law 3.2.1

$$\bar{\phi}^*(s) + Q(s)\hat{u}(s) - R\bar{w}(s) = \bar{\phi}^*(s) + \frac{q(s)}{Z^-(s)} - \bar{w}(s) = 0 \tag{13}$$

gives:

$$\hat{u}(s) = -\frac{1+0.4948s}{s[(0.07515 + q) + 0.3qs]}\bar{y}(s) \tag{14}$$

$$+ \frac{(1+0.3s)(1+0.03s)}{s[(0.07515 + q) + 0.3qs]}\bar{w}(s)$$

Note that this controller has integral action, and its gain may be varied using the scalar weighting factor q.

References

1. Goodwin, G.C., "An Amplitude constrained minimum variance controller," Electronics Letters, vol. 8, p. 181, 1972.

2. Clarke, D.W. and Gawthrop, P.J., "Self-tuning control," Proceedings IEE, vol. 126, no. 6, pp. 633-640, 1979.

3. Horowitz, I., Synthesis of feedback systems, Academic Press, 1963.

4. Horowitz, I. and Sidi, M., "Synthesis of feedback systems with large plant ignorance for prescribed time-domain tolerances," International Journal of Control, vol. 16, pp. 287-309, 1972.

5. Clarke, D.W. and Gawthrop, P.J., "Self-tuning controller," Proceedings IEE, vol. 122, no. 9, pp. 929-934, 1975.

6. Kumar, R. and Moore, J.B., "On adaptive minimum-variance regulation for non-minimum phase plants," Automatica, vol. 19, p. 499, 1983.

7. Clarke, D.W. and Gawthrop, P.J., "Comments on: 'On adaptive minimum variance regulation for nonminimum phase plants'," Automatica, vol. 20, no. 2, p. 261, 1984.

8. Dorf, R.G., Modern control systems, Addison-Wesley, 1980.

9. Kwakernaak, H, and Sivan, R.,, Linear optimal control systems, Wiley, 1972.

10. Kucera, V., Discrete linear control: The polynomial equation approach., Wiley, Prague, 1979.

11. Astrom, K.J., Introduction to stochastic control theory, Academic Press, New York, 1970.

12. Kailath, T., Linear Systems, Prentice-Hall, 1980.

13. Goodwin, G.C. and Sin, K.S., Adaptive filtering prediction and control, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1984.

14. Gawthrop, P.J., "Some interpretations of the self-tuning controller," Proceedings IEE, vol. 124, no. 10, pp. 889-894, 1977.

15. Smith, O.J.M., "A controller to overcome dead-time," ISA transactions, vol. 6, no. 2, pp. 28-33, 1959.

16. Marshall, J.E., Control of time-delay systems, Peter Peregrinus Ltd., 1979.

17. Astrom, K.J. and Wittenmark, B., Computer controlled systems, Prentice Hall, 1984.

18. Gawthrop, P.J., "Self-tuning PI and PID Controllers," in Proceedings of the IEEE conference on "Applications of Adaptive and Multivariable Control", Hull, 1982.

19. Gawthrop, P.J., "Self-tuning PID controllers: Algorithms and implementation," IEEE Transactions on Automatic Control., vol. AC-31, no. 3, 1986.

20. Gawthrop, P.J., "Robustness of self-tuning controllers. PartI: Single-input single-output systems.," Report CE/T/13, School of Engineering and Applied Sciences, Univ. of Sussex., 1985.

21. Rohrs, C.E., Valavani, L., Athans, M., and Stein, G., "Robustness of continuous-time adaptive control in the presence of unmodeled dynamics," Trans. IEEE, vol. AC-30, pp. 881-889, 1985.

CHAPTER 4
Non-Adaptive Robustness

Aims. To investigate the effect of neglected system dynamics on the stability of (non-adaptive) emulator-based controllers. To relate a number of stability criteria. To provide the background for the robustness analysis of self-tuning controllers.

4.1. INTRODUCTION

In the previous section, it was assumed that the nominal system exactly represented the actual system to be controlled. This is an unrealistic assumption in practice. This chapter presents an analysis of the robustness of the controllers designed in the previous chapter to neglected system dynamics; that is, the extent to which the closed-loop system remains satisfactory in the presence of neglected system dynamics is investigated. The system dynamics are assumed to be linear, but it is possible to extend the results to non-linear systems[1]. The corresponding analysis for self-tuning control is presented in chapter 7, where it will be found that the adaptive and non-adaptive results are closely related. This relationship is explored further in chapter 8.

Three approaches to the robustness problem are presented:

1. A classical Nyquist approach.

2. A method based on a discrete-time analysis of Astrom[2,3].

3. A method based on the discrete-time analysis of Gawthrop and Lim[1].

The advantage of 2 and 3 is that the results are expressed directly in terms of the controller design parameters and the neglected dynamics; the advantage of 3 is that the results are directly applicable to the analysis of certain self-tuning versions. We shall be concerned to relate these three methods as they all provide different insights into the robustness problem.

4.2. NEGLECTED PLANT DYNAMICS

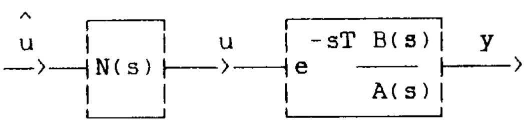


Figure 4.2.1 Neglected plant dynamics

In the previous chapter, it was tacitly assumed that the system was exactly modelled. This assumption is not practically realistic. In this chapter we retain the linearity assumption but account for possible errors in plant modelling. Thus the system equation 1.9.1 is replaced by:

$$\bar{y}(s) = H(s)\hat{u}(s) + \frac{C(s)}{A(s)}v(s) \tag{1}$$

where H(s) is a proper transfer function representing a linear time-invariant system and $\hat{u}(s)$ is the controller output. This true system equation may be rewritten in terms of the nominal system as (see Figure 4.2.1):

$$\bar{u}(s) = N(s)\hat{u}(s) \tag{2}$$

where the <u>neglected dynamics</u> N(s) are given by:

$$N(s) = e^{sT}\frac{A(s)}{B(s)}H(s) \tag{3}$$

4.3. ROBUSTNESS BASED ON THE ACTUAL FEEDBACK SYSTEM

The standard way of analysing the robustness properties of a feedback loop is in terms of the Nyquist diagram based on the actual system loop-gain (see[4], for example). Although this method will not be used very much here, it is introduced to provide a link between such classical methods and the methods discussed later in this chapter.

As an example, consider the emulator-based controller using the signal $\bar{\phi}_3(s)$. The emulator is of the form (see chapter 2, section 5):

$$\bar{\phi}_3^\star(s) = \frac{F_3(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{E_3(s)B(s)}{C(s)Z^-(s)}\bar{u}(s) \tag{1}$$

The corresponding control law can, from section 3.2, be written as

$$\bar{\phi}^\star(s) + Q(s)\hat{u}(s) - R\bar{w}(s) = 0 \tag{2}$$

hence

$$\frac{F_3(s)}{C(s)Z^+(s)}\bar{y}(s)[\frac{E_3(s)B(s)}{C(s)Z^-(s)} + Q(s)]\bar{u}(s) = R(s)\bar{w}(s) \tag{3}$$

We can ignore the setpoint when treating stability; the feedback transfer function relating $\bar{u}(s)$ to $\bar{y}(s)$ is then

$$\frac{\bar{u}(s)}{\bar{y}(s)} = \frac{F_3(s)Z^-(s)}{E_3(s)B(s)Z^+(s) + Q(s)C(s)Z^-(s)} \tag{4}$$

The actual system loop-gain is then given by the product of this transfer function and the system loop-gain as

$$L_a(s) \triangleq N(s)e^{-sT}\frac{B(s)}{A(s)}\frac{F_3(s)Z^-(s)}{E_3(s)B(s)Z^+(s) + Q(s)C(s)Z^-(s)} \tag{5}$$

The well-known theorem of Nyquist (as extended by Desoer[5] to the time-delay case) gives the following robustness criterion:

Non-adaptive criterion 1

The (non-adaptive) closed-loop system is stable iff the Nyquist locus

$$L_a(j\omega) \tag{6}$$

obeys Nyquist's criterion.

4.4. THE ERROR FEEDBACK SYSTEM

The analysis of both non-adaptive and adaptive control is simplified by rewriting the relevant equations to form an error feedback system which exhibits how errors, rather than actual signals, are propagated.

The neglected dynamics give rise to two extra error signals in the notional feedback system, the first due to the system input not being the controller output, the second due to the emulator being no longer exact. These two error sources are considered in turn.

The Control Signal Error

The neglected dynamics can be represented by the equivalent expression

$$\hat{u}(s) = \bar{u}(s) - \tilde{u}(s) \tag{1}$$

where the control signal error $\tilde{u}(s)$ is given by

$$\tilde{u}(s) = [N(s) - 1]\hat{u}(s) \tag{2}$$

The Emulator Approximation Error

The emulator based on the nominal system cannot be used directly in the presence of unmodelled dynamics as the input $\bar{u}(s)$ to the nominal system is not available. An approximate emulator can, however, be easily obtained by replacing the unknown nominal system input $\bar{u}(s)$ by the known controller output $\hat{u}(s)$. The resultant error depends on the deviation of the neglected dynamics N(s) from unity.

The approximate emulator (with output $\bar{\phi}^a(s)$) is thus given by:

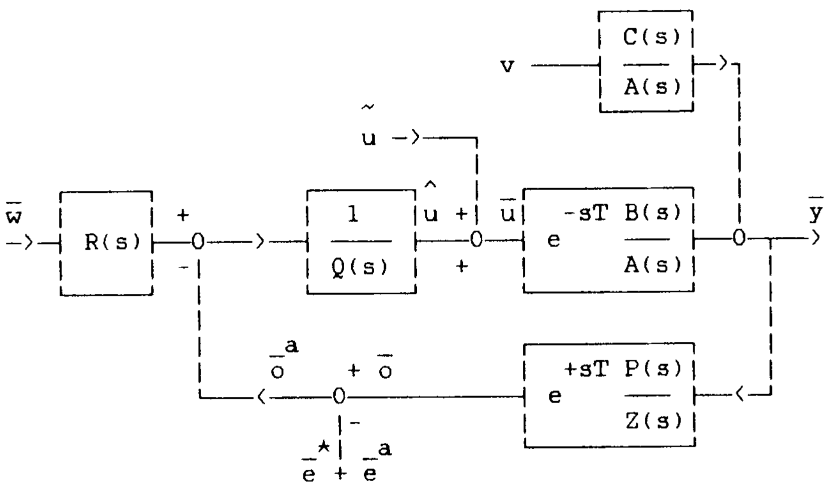
$$\bar{\phi}^a(s) = \frac{F(s)}{C(s)Z^+(s)}\bar{y}(s) + \frac{E(s)B(s)}{C(s)Z^-(s)}\hat{u}(s) \tag{3}$$

The emulator approximation error introduced by replacing $\bar{u}(s)$ by $\hat{u}(s)$ is given by

$$\bar{e}^a(s) = \bar{\phi}^\star(s) - \bar{\phi}^a(s) = \frac{E(s)B(s)}{C(s)Z^-(s)}\tilde{u}(s) \tag{4}$$

<u>The modified notional feedback system</u>

These two errors arising from the neglected dynamics N(s) modify the properties of the notional feedback system of the previous chapter by forming two additional input signals as in Fig 4.4.1.



<u>Figure 4.4.1 The modified notional feedback system</u>

From this block diagram, the control signal can be written in terms of the notional loop-gain as:

$$\hat{u}(s) = \frac{L(s)}{1+L(s)}[- \tilde{u}(s) + \frac{A(s)Z(s)}{B(s)P(s)}(\bar{z}(s) + \bar{e}^{a}(s)) \qquad (5)$$

where the <u>equivalent setpoint</u> $\bar{z}(s)$ is given by equation 3.3.8 as

$$\bar{z}(s) = R(s)\bar{w}(s) - e^{sT} \frac{P(s)C(s)}{Z(s)A(s)}\bar{v}(s) + \bar{e}^{\star}(s) \qquad (6)$$

$$= R(s)\bar{w}(s) + [\frac{E(s)}{Z^{-}(s)} - e^{sT} \frac{P(s)C(s)}{Z(s)A(s)}]\bar{v}(s)$$

The error feedback system

The equations for $\tilde{u}(s)$ and $\bar{e}^a(s)$ are combined with those of the modified notional feedback system in Figure 4.4.1 to give Figure 4.4.2.

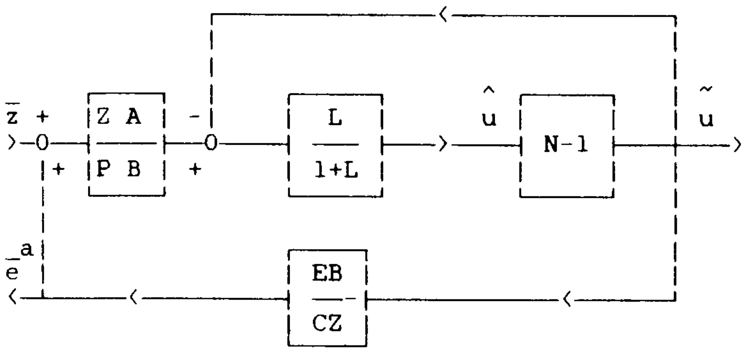


Figure 4.4.2 The error feedback system

This Figure shows a two-loop feedback system which can be transformed to a number of equivalent single-loop systems using standard techniques. Each such equivalent single loop leads to a stability criterion for the non-adaptive feedback systems. Two such criteria are considered here. Both criteria have been given previously in a discrete-time context: the first is due to Astrom[2] (see[3] section 10.6, Theorem 10.3), and the second is similar to that given by Gawthrop and Lim[1]. The second criterion is important because, unlike the first, it extends to the adaptive case.

4.5. ROBUSTNESS - ASTROM'S CRITERION

Looking at the feedback system of Fig 4.4.2 in terms of $\tilde{u}(s)$, it can be written as a single loop system in terms of the intermediate variable $\bar{u}$ as:

$$\tilde{u}(s) = [N(s) - 1][\hat{u}_0(s) - \frac{L(s)}{1+L(s)} \bar{\upsilon}] \tag{1}$$

$$\bar{\upsilon} = (1 - \frac{E(s)A(s)Z^+(s)}{P(s)C(s)})\tilde{u}(s) \tag{2}$$

$$= e^{-sT} \frac{F(s)Z^-(s)}{C(s)P(s)}\tilde{u}(s)$$

where $\hat{u}_0(s)$ is the control signal corresponding to no neglected dynamics and is given by

$$\hat{u}_0(s) = \frac{L(s)}{1+L(s)} \frac{Z(s)A(s)}{P(s)B(s)}\bar{z}(s) \tag{3}$$

This feedback system appears in Figure 4.5.1.

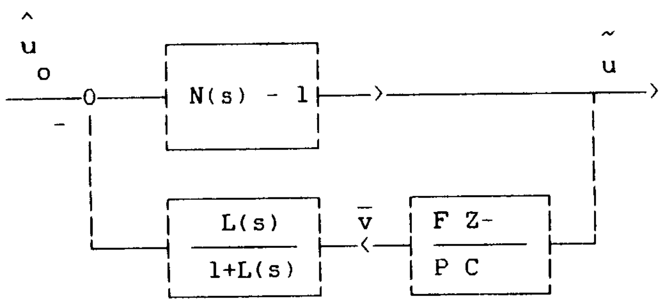


Figure 4.5.1 The single loop error feedback system

From Fig 4.5.1, Nyquist's theorem gives the following robustness criterion:

Non-adaptive criterion 2

The (non-adaptive) closed-loop system is stable iff the Nyquist locus

$$M'(s) \triangleq \frac{L\ e^{-sT}}{1+L} \frac{FZ^{-}(s)}{P(s)C(s)}[1 - N(s)] \tag{4}$$

obeys Nyquist's criterion.

A more conservative criterion is that the modulus of the loop gain is less than unity at all frequencies. Noting that $|e^{-j\omega T}| = 1$, this gives the following robustness criterion:

Non-adaptive criterion 3

The non-adaptive feedback system of Figure 4.5.1 is stable if:

1. $M'(s)$ is stable, and

2. $|M'(j\omega)| < 1$ for all ω.

Astrom's formulation

In the special case that the actual system is given by:

$$H(s) = e^{-sT_0} \frac{B_0(s)}{A_0(s)} \tag{5}$$

then

$$N(s) = e^{-s(T_0-T)} \frac{B_0(s)A}{A_0(s)B} \tag{6}$$

The relevant Nyquist locus is then given by:

$$M'(s) = \frac{L(s)}{1+L(s)} \frac{F(s)A(s)Z^{-}(s)}{P(s)B(s)C(s)}\left[e^{-sT_0}\frac{B_0(s)}{A_0(s)} - e^{-sT}\frac{B(s)}{A(s)}\right] \tag{7}$$

Part 2 of the conservative criterion then may be rearranged as:

$$\left| e^{-sT_0}\frac{B_0(s)}{A_0(s)} - e^{-sT}\frac{B(s)}{A(s)} \right| < \left| \frac{1+L(s)}{L(s)} \frac{B(s)P(s)C(s)}{A(s)Z^{-}(s)F(s)} \right| \tag{8}$$

for all $s = j\omega$.

In the particular case that $L(s) = \infty$, and so

$$\frac{L(s)}{1+L(s)}=1, \tag{9}$$

and both the nominal and actual systems are stable, this reduces to the criterion derived by Astrom[2] Theorem 1, and reproduced in[3] section 10.6 as Theorem 10.3.

4.6. ROBUSTNESS - THE M-LOCUS

An alternative way of analysing the error feedback system of Fig. 4.4.2 is in terms of $\bar{e}^a(s)$. Solving for the upper feedback loop:

$$\bar{e}^a(s) = \frac{Z^+(s)A(s)E(s)}{P(s)C(s)} \frac{L(s)[N(s)-1]}{1+L(s)N(s)}[\bar{z}(s) + \bar{e}^a(s)] \tag{1}$$

Combining this with the rest of the block diagram:

$$\bar{e}^a(s) = - M(s)[\bar{z}(s) + \bar{e}^a(s)] \tag{2}$$

(see Figure 4.6.1) where the transfer function M(s) is

$$M(s) = \frac{Z^+(s)E(s)A(s)}{P(s)C(s)} \frac{N^{-1}(s)-1}{1+L^{-1}(s)N^{-1}(s)} \tag{3}$$

$$= \frac{B(s)E(s)}{Z^-(s)Q(s)C(s)} \frac{1-N(s)}{1+L(s)N(s)}$$

This leads to an alternative robustness criterion:

Non-adaptive criterion 4

The (non-adaptive) closed-loop system is stable iff the Nyquist locus

$$M(j\omega) \tag{4}$$

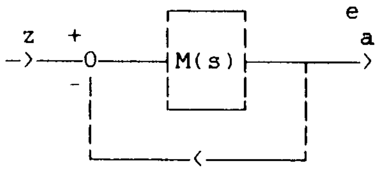


Figure 4.6.1 The single loop error feedback system

obeys Nyquist's criterion.

Once again, a more conservative criterion is:

Non-adaptive criterion 5

1. M(s) represents a stable system (all poles have negative real parts)
2. $|M(j\omega)| < 1$ for all ω

4.7. ROHRS EXAMPLE

In a celebrated paper[6], Rohrs and his colleagues illustrated the poor robustness properties of a particular model-reference adaptive control algorithm by examining its performance on two particular example systems. In this section, the second of these example systems is used to illustrate the non-adaptive robustness properties of the detuned model-reference adaptive controller of section 3.10.

The system

Rohrs' system, in our notation, is described by:

$$H(s) = \frac{200}{(s+1)(s^2 + 8s + 100)} \tag{1}$$

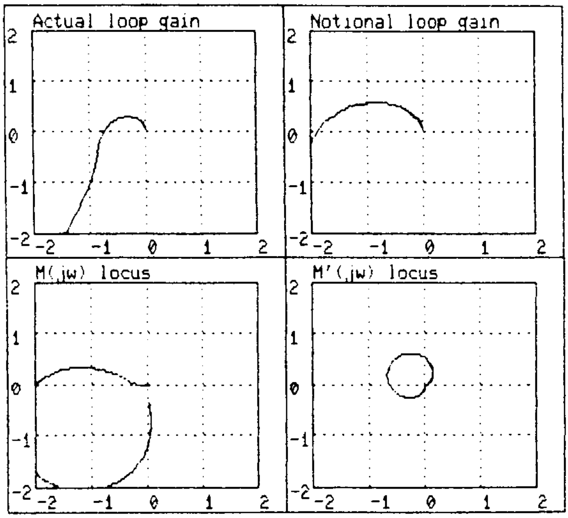


Figure 4.7.1 Example 1

One possible decomposition into nominal (B(s)/A(s)) and neglected dynamics (N(s)) is

$$\frac{B(s)}{A(s)} = \frac{2b}{1+s}; \quad N(s) = \frac{1}{b}\,\frac{100}{s^2 + 8s + 100} \tag{2}$$

Thus the actual system is third order; we are assuming for design purposes that it is first order. The neglected dynamics are second order with natural frequency 10rad sec^{-1} and damping ratio 0.4. There are clearly an infinite number of possible decompositions having the property that

$$H(s) = N(s)\frac{B(s)}{A(s)} \tag{3}$$

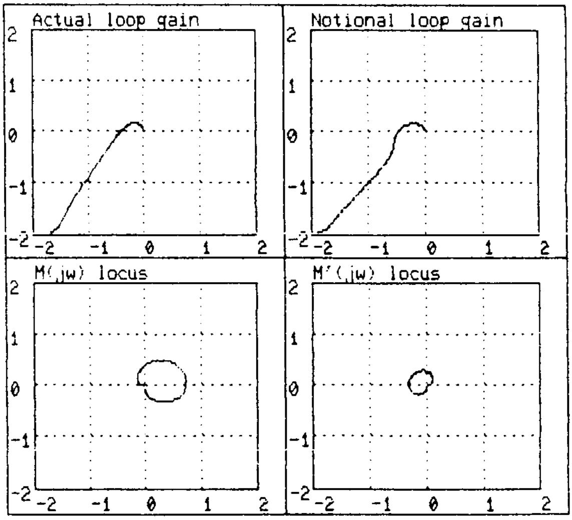


Figure 4.7.2 Example 2

The design parameters

Rohrs and colleagues attempt to match the reference model

$$\frac{3}{s+3} \simeq \frac{1}{1+0.3s} \tag{4}$$

For consistency with this requirement, choose

$$P(s) = 1 + 0.3s \tag{5}$$

As, for practical reasons, we would like integral action, choose

$$A(s) = s(1+s);\ B(s) = 2s \tag{6}$$

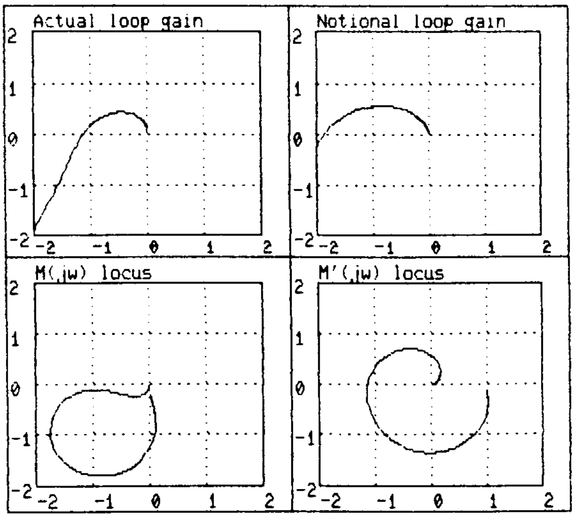


Figure 4.7.3 Example 3

This leaves C(s), Q(s) and Z(s) to choose. To achieve the right sort of disturbance response, choose

$$C(s) = P(s) = 1 + 0.3s \tag{7}$$

To make $\bar{\phi}(s)$ realisable, choose 1/Z(s) to be the first order low-pass filter:

$$Z(s) = 1 + 0.03s \tag{8}$$

Finally, make Q(s) zero at s=0 by choosing

$$Q(s) = \frac{qs}{Z(s)} = \frac{qs}{1+0.03s} \tag{9}$$

Note that q=0 would give exact model following; q>0 detunes the controller at high frequencies.

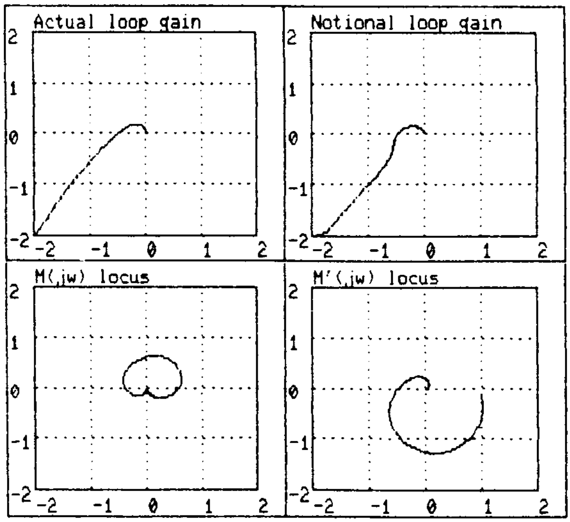


Figure 4.7.4 Example 4

Robustness analysis

To exemplify the use of the various criteria presented in this chapter, we will consider four examples (Figures 4.7.1-4) based on that of Rohrs.

The four examples have the following in common:

1. Four frequency loci are plotted for values of $\omega > 0$:

   a) The actual loop gain: $L_a(j\omega)$ (equation 4.3.5)

   b) The notional loop gain (with neglected dynamics included): $N(j\omega)L(j\omega)$

   c) The M-locus $M(j\omega)$ (equation 4.6.4)

d) The M'-locus M'(jω) (equation 4.5.4)

2. The actual system H(s) is as given in equation 4.7.1.

3. The emulator and controller design parameters are as given in equations 4.7.4-9.

The four examples are different in the following ways. The parameter b determining the decomposition of equation 2, and the control weighting factor q of equation 9, are varied as in the following table (see Figures 4.7.1-4):

Example	b	q
1	1.0	0.05
2	1.0	0.2
3	0.5	0.05
4	0.5	0.2

<u>Remarks</u>

1. As both the nominal and actual systems are stable, the loci corresponding to $L_a(s)$ and M'(s) imply stability if there are no encirclements of the -1 point. Both these loci predict stability for examples 1,2&4 and instability for example 3.

2. In this example, stability of the transfer function M(s) depends on the stability of

$$\frac{L(s)N(s)}{1+L(s)N(s)}$$

In examples 1 and 3, the N(s)L(s) locus encircles the -1 point, indicating instability; in examples 2 and 4 it does not, indicating stability. In examples 2 and 4, the M-locus does not encircle the -1 point, indicating stability. In example 1, the M-locus encircles -1 the requisite number of times in an anti-clockwise sense, indicating stability; whereas in example 3 the M-locus does not encircle the -1 point, indicating instability.

3. As criteria 1,2 and 3 are all necessary and sufficient, it is not surprising that they all give the same stability predictions. The conservative criteria, however, do not always agree.

4. The N(s)L(s) locus is the same for examples 1 and 3, and for 2 and 4. This locus is not affected by the choice of the decomposition of H(s) into N(s) and $\frac{B(s)}{A(s)}$.

References

1. Gawthrop, P.J. and Lim, K.W., "On the robustness of self-tuning controllers," Proc. IEE, vol. 129 ptD, pp. 21-29, 1982.

2. Astrom, K.J., "Robustness of a design method based on assignment of poles and zeros," Transactions IEEE, vol. AC-25, pp. 588-591, 1980.

3. Astrom, K.J. and Wittenmark, B., Computer controlled systems, Prentice-Hall, 1984.

4. Dorf, R.G., Modern control systems, Addison-Wesley, 1980.

5. Desoer, C.A., "A general formulation of the Nyquist criterion," IEEE Trans. on Circuit Theory, vol. CT-12, pp. 230-234, 1965.

6. Rohrs, C.E., Valavani, L., Athans, M., and Stein, G., "Robustness of continuous-time adaptive control in the presence of unmodeled dynamics," Trans. IEEE, vol. AC-30, pp. 881-889, 1985.

CHAPTER 5
Least-Squares Identification

Aims. To discuss linear-in-the-parameter system models. To introduce and derive the continuous-time least-squares method and to analyse its properties. To show that discrete-time least-squares methods can be used to identify continuous-time parameters.

5.1. INTRODUCTION

Least-squares parameter identification has been used in self-tuning control for a long time[1,2,3,4]. However this has usually been in a discrete-time context. A notable exception is the work of Young[5] who combined digital least-squares with analogue components to give estimates of continuous time transfer function parameters and hence to control a system. In a survey paper[6], Young points out that as well as discrete-time estimation of discrete-time system parameters, discrete-time and continuous-time estimation of continuous-time system parameters is also possible. These two latter approaches to the identification of continuous-time parameters are considered here: continuous-time identification of continuous-time parameters and discrete-time identification of continuous-time parameters. The former is of theoretical interest as a limiting case; the latter is more appropriate to practical application. In each case, we require a linear in the parameters system representation; so this is considered first.

5.2. LINEAR IN THE PARAMETERS SYSTEMS

The standard linear in the parameters model to be used in this book is

$$\Psi(t) = \underline{X}^T(t)\underline{\theta} + e(t) \tag{1}$$

where $\Psi(t)$ is the scalar system output, $\underline{X}(t)$ is a column vector of measured variables, $\underline{\theta}$ is a column vector of parameters and $e(t)$ is the linear in the parameters error. Thus the scalar output of a linear in the parameters model is composed of two terms: the sum of products of measurements and parameters, and an error term. Particular cases will be derived in detail in chapter 6; for the purposes of this chapter the linear in the parameters model is motivated with a simple example.

Example: Linear in the parameters model

Consider the first order system:

$$\bar{y}(s) = \frac{b}{s+a}\bar{u}(s) + \frac{d}{s+a} + \frac{1}{s+a}\bar{v}(s) \tag{2}$$

where d represents the effect of initial conditions. Choosing a polynomial $C_s(s) = s+c$ $(c>0)$, this may be rewritten as

$$\frac{s+a}{s+c}\bar{y}(s) = \frac{b}{s+c}\bar{u}(s) + \frac{d}{s+c} + \frac{1}{s+c}\bar{v}(s) \tag{3}$$

Rearranging gives

$$\bar{y}(s) = c-a\ \frac{\bar{y}(s)}{s+c} + b\ \frac{\bar{u}(s)}{s+c} + d\ \frac{1}{s+c} + \frac{1}{s+c}\bar{v}(s) \tag{4}$$

This is in the linear in the parameters form with

$$\bar{\Psi}(s) = \bar{y}(s); \; \bar{e}(s) = \frac{1}{s+c}\bar{v}(s) \tag{5}$$

and

$$\underline{\theta} = \begin{vmatrix} c-a \\ b \\ d \end{vmatrix}; \; \underline{\bar{X}}^T(s) = \frac{1}{s+c}\begin{vmatrix} \bar{y}(s) \\ \bar{u}(s) \\ 1 \end{vmatrix} \tag{6}$$

The data vector $\underline{X}(t)$ is formed from the output of three low-pass filters with transfer function $\frac{1}{s+c}$, one driven by y(t), one driven by u(t) and the other with no input. The first two filters have zero initial condition; the third has unit initial condition. See[7] for more details.

<u>Example</u>: <u>The effect of offset</u>

Consider the same first order system but with a unit constant added:

$$\bar{y}(s) = \frac{b}{s+a}\bar{u}(s) + \frac{d}{s+a} + \frac{1}{s+a}\bar{v}(s) + \frac{1}{s} \tag{7}$$

$$= \frac{sb}{s(s+a)}\bar{u}(s) + \frac{(sd+s+a)}{s(s+a)} + \frac{s}{s+a}\bar{v}(s)$$

Where d represents the effect of initial conditions and 1/s represents a constant. Choosing a polynomial $C(s) = (s+c)^2$ $(c>0)$, this may be rewritten as

$$\frac{s(s+a)}{(s+c)^2}\bar{y}(s) = \frac{sb}{(s+c)^2}\bar{u}(s) + \frac{s(1+d)+a}{(s+c)^2} + \frac{s}{(s+c)^2}\bar{v}(s) \tag{8}$$

Rearranging gives

$$\left[1 - \frac{c^2}{(s+c)^2}\right]\bar{y}(s) = (2c-a)\,\frac{s\bar{y}(s)}{(s+c)^2} + b\,\frac{s\bar{u}(s)}{(s+c)^2} \tag{9}$$

$$+ 1+d\,\frac{s}{(s+c)^2} + \frac{a}{(s+c)^2} + \frac{s}{(s+c)^2}\bar{v}(s)$$

This is in the linear in the parameters form with

$$\bar{\Psi}(s) = [1 - \frac{c^2}{(s+c)^2}]\ \bar{y}(s) = \frac{s^2+2cs}{(s+c)^2}\ \bar{y}(s) \qquad (10)$$

$$\bar{e}(s) = + \frac{s}{(s+c)^2}\bar{v}(s) \qquad (11)$$

and

$$\underline{\theta} = \begin{vmatrix} 2c-a \\ b \\ d+1 \\ a \end{vmatrix}; \quad \bar{\underline{X}}^T(s) = \frac{1}{(s+c)^2}\begin{vmatrix} s\bar{y}(s) \\ s\bar{u}(s) \\ s \\ 1 \end{vmatrix} \qquad (12)$$

This model has the important property that the filtering of $\bar{y}(s)$ and $\bar{u}(s)$ removes constant components.

5.3. CONTINUOUS-TIME LEAST-SQUARES CRITERION

Suppose we have a linear-in-the parameters system as in equation 1 of the previous section, with output $\Psi(t)$, parameter vector $\underline{\theta}$ and data vector $\underline{X}$:

$$\Psi(t) = \underline{X}(t)\underline{\theta} + e(t) \qquad (1)$$

Assume that $\Psi(t)$ and $\underline{X}$ can be measured but that the nominal parameter vector $\underline{\theta}$ is unknown. Suppose that we choose an estimate $\hat{\underline{\theta}}(t)$ of $\underline{\theta}$. Then we can deduce an estimate $\hat{\Psi}(\tau)$ of $\Psi(\tau)$ at a time τ (less that the current time t) based on the current estimate $\hat{\underline{\theta}}(t)$ from the equation

$$\hat{\Psi}(\tau) = \underline{X}^T(\tau)\hat{\underline{\theta}}(t) \qquad (2)$$

The resultant estimation error $\hat{e}(t,\tau)$ is then defined as

$$\hat{e}(t,\tau) \triangleq \Psi(\tau) - \hat{\Psi}(\tau) \qquad (3)$$

For convenience, we shall write the estimation error based on the current parameter estimate as

$$\hat{e}(t) \triangleq \hat{e}(t,t) = \Psi(t) - \hat{\Psi}(t) \tag{4}$$

The aim of least-squares estimation is to choose the current estimate $\hat{\underline{\theta}}(t)$ to minimise a weighted average estimation error over all measurements from time 0 to time t. The choice of the particular criterion leading to the weighted average is somewhat arbitrary. As is usual, a quadratic form with exponential weighting ('least-squares') is used in this book. This method (particularly in its discrete-time version) has a long track record of successful application. It will also be shown in the sequel that using the least-squares approach endows an self-tuning algorithm with desirable robustness properties.

The exponentially weighted least-squares cost function which we will use here is

$$J(\hat{\underline{\theta}}(t),t) = \tfrac{1}{2}e^{-\beta t}(\hat{\underline{\theta}}(t) - \hat{\underline{\theta}}_0)^T\underline{S}_0(\hat{\underline{\theta}}(t) - \hat{\underline{\theta}}_0) \tag{5}$$

$$+ \tfrac{1}{2}\int_0^t e^{-\beta(t-\tau)}\hat{e}(t,\tau)^2\, d\tau$$

where β is a non-negative scalar:

$$\beta \geq 0 \tag{6}$$

$\underline{S}_0$ is a positive definite matrix:

$$\underline{S}_0 > 0 \tag{7}$$

The first term in the cost allows us to include a prior estimate in the algorithm; often we would wish to start a self-tuning controller off with a known 'safe' set of coefficients, and this feature allows this. The second term brings the measured data into the criterion; it is a weighted average of the square of past estimation errors based on the current parameter estimate. The exponential weighting coefficient β acts as a <u>forgetting factor</u>. As time t increases, the effect of old data at time $\tau < t$ is discounted exponentially with the elapsed time $t - \tau$; the initial parameter estimate $\underline{\hat{\theta}}_0$ is discounted in a similar way. S(0) varies the weight given to the initial parameter estimate.

Note that J is a function of two variables: time t and parameter estimate $\underline{\hat{\theta}}(t)$.

The <u>least-squares estimate</u> is that value of $\underline{\hat{\theta}}(t)$ which minimises this cost for each time $t \geq 0$. At such a minimum, the partial derivative of $J(\underline{\hat{\theta}}(t),t)$ with respect to $\underline{\hat{\theta}}(t)$ is zero:

$$J_1(\underline{\hat{\theta}}(t),t) \triangleq \frac{\partial}{\partial\hat{\theta}} J(\underline{\hat{\theta}}(t),\ t) = 0 \tag{8}$$

Note that $J_1(\underline{\hat{\theta}}(t),t)$ is a <u>vector</u> of the same dimension as $\underline{\hat{\theta}}(t)$.

5.4. MINIMISATION OF THE COST FUNCTION

We consider the minimisation of the cost function in three stages:

1. Existence and uniqueness of a minimum.

2. A <u>non-recursive</u> (integral) form of the solution.

3. A <u>recursive</u> (differential equation) form of the solution.

Existence of solutions

Before performing the minimisation, it is important to know if a minimum (with respect to $\hat{\underline{\theta}}(t)$) exists. The cost function is quadratic in $\hat{\underline{\theta}}(t)$, so existence depends on the second derivative:

$$J_2(\hat{\underline{\theta}}(t),t) = \frac{\partial^2}{\partial\theta^2}J(\hat{\underline{\theta}}(t),t) = \underline{S}(t) \tag{1}$$

where

$$\underline{S}(t) \triangleq e^{-\beta t}\underline{S}_0 + \int_0^t e^{-\beta(t-\tau)}\underline{X}(\tau)\underline{X}^T(\tau)d\tau \tag{2}$$

$\underline{S}_0$ is, by definition, positive definite; hence so is $e^{-\beta t}\underline{S}_0$. The second term $\underline{S}(t)$ depends on the data but, because of its form, is non-negative definite. Thus

$$\underline{S}(t) = J_2(\hat{\underline{\theta}}(t),t) > 0 \tag{3}$$

This condition is sufficient to ensure existence and uniqueness of the solution of the minimisation problem. There is one global minimum and it occurs when the first derivative of $J(\hat{\underline{\theta}}(t),t)$ with respect to $\hat{\underline{\theta}}(t)$ is zero.

However, for practical purposes, this is not good enough, as $J_2(\hat{\underline{\theta}}(t),t)$ may become nearly singular. Not only must $J_2(\hat{\underline{\theta}}(t),t)$ be non-singular, but it must be <u>numerically non-singular</u>. Also, for later theoretical reasons, we require that $\underline{S}(t)$ be <u>uniformly</u> positive definite (even when $\beta>0$) in the sense that

$$\underline{S}(t) > \Sigma \tag{4}$$

where Σ is a <u>constant</u> positive definite matrix.

In practice, then, the data-dependent persistent excitation condition

$$\underline{S}(t) > \Sigma > 0 \tag{5}$$

is often required.

Non-recursive solution

Taking the partial derivative of $J(\hat{\underline{\theta}}(t),t)$ with respect to $\hat{\theta}(t)$

$$J_1(\hat{\underline{\theta}}(t),t) = \tfrac{1}{2}e^{-\beta t}\underline{S}_0(\hat{\underline{\theta}}(t) - \hat{\underline{\theta}}_0) \tag{6}$$

$$+ \int_0^t e^{-\beta(t-\tau)}\underline{X}(\tau)(\underline{X}^T(\tau)\hat{\underline{\theta}}(t) - \Psi(\tau))d\tau$$

$$= \tfrac{1}{2}e^{-\beta t}\underline{S}_0(\hat{\underline{\theta}}(t) - \hat{\underline{\theta}}_0)$$

$$+ [\int_0^t e^{-\beta(t-\tau)}\underline{X}(\tau)\underline{X}^T(\tau)d\tau]\hat{\underline{\theta}}(t)$$

$$- \int_0^t e^{-\beta(t-\tau)}\underline{X}(\tau)\Psi(\tau))d\tau$$

Setting $J_1(\hat{\Theta}(t),t)=0$, it follows that the value $\hat{\underline{\theta}}(t)$ corresponding to the minimum of $J(\hat{\underline{\theta}}(t),t)$ is given by

$$\underline{S}(t)\hat{\theta}(t) = e^{-\beta t}\underline{S}_0\hat{\underline{\theta}}_0 + \int_0^t e^{-\beta(t-\tau)}\underline{X}(\tau)\Psi(\tau)d\tau \tag{7}$$

This equation, together with that for $\underline{S}(t)$ (5.4.2), forms the non-recursive solution of the least-squares estimation problem. This solution is unique at time t iff $\underline{S}(t)$ is non-singular, and is then given by

$$\hat{\underline{\theta}}(t) = \underline{S}^{-1}(t)\ e^{-\beta t}\underline{S}_0\hat{\underline{\theta}}_0 + \int_0^t e^{-\beta(t-\tau)}\underline{X}(\tau)\Psi(\tau)d\tau \tag{8}$$

Recursive solution

To get a recursive solution, we first convert the integral form of the cost (5.3.2) to a differential form by taking partial derivatives with respect to time. Taking partial derivatives with respect to time

$$\frac{\partial}{\partial t}J(\hat{\underline{\theta}}(t),t) + \beta J(\hat{\underline{\theta}}(t),t) = \frac{1}{2}\hat{e}(t)^2 \tag{9}$$

and then taking i partial derivatives with respect to $\hat{\underline{\theta}}(t)$

$$\frac{\partial}{\partial t}J_i(\hat{\underline{\theta}}(t),t) + \beta J_i(\hat{\underline{\theta}}(t),t) = \frac{1}{2}\frac{\partial^i}{\partial\theta^i}\hat{e}(t)^2 \tag{10}$$

where

$$J_i(\hat{\underline{\theta}}(t),t) \triangleq \frac{\partial^i}{\partial\theta^i}J(\hat{\underline{\theta}}(t),t) \tag{11}$$

The total derivative with respect to time is then given by the formula

$$\frac{d}{dt}J_i(\hat{\underline{\theta}}(t),t) = \frac{\partial}{\partial t}J_i(\hat{\underline{\theta}}(t),t) + \frac{\partial}{\partial\hat{\theta}}J_i(\hat{\underline{\theta}}(t),t).\frac{d}{dt}\hat{\underline{\theta}}(t) \tag{12}$$

$$= \frac{\partial}{\partial t}J_i(\hat{\underline{\theta}}(t),t) + J_{i+1}(\hat{\underline{\theta}}(t),t).\frac{d}{dt}\hat{\underline{\theta}}(t)$$

A formula for the optimum value $\hat{\underline{\theta}}(t)$

Recalling that our condition for the optimal value $\hat{\underline{\theta}}(t)$ is $J_1(\hat{\theta}(t),t) = 0$, it follows from 12 with i=1 that

$$J_2(\hat{\underline{\theta}}(t),t)\frac{d}{dt}\hat{\underline{\theta}}(t) = \frac{1}{2}\frac{\partial}{\partial\hat{\theta}}\hat{e}(t)^2 \tag{13}$$

$$= \underline{X}(t)\hat{e}(t)$$

Noting from equation 5.4.1 that $J_2(\hat{\underline{\theta}}(t),t) = \underline{S}(t)$, it follows that

$$\frac{d}{dt}\hat{\underline{\theta}}(t) = \underline{S}^{-1}(t)\underline{X}(t)\hat{e}(t) \tag{14}$$

A formula for $J_2(\hat{\underline{\theta}}(t),t)$

As J is quadratic in $\hat{\underline{\theta}}(t)$, it follows that $J_i(\hat{\underline{\theta}}(t),t)=0$ for i>2. Thus $J_2(\hat{\underline{\theta}}(t),t)=\underline{S}(t)$ is given by:

$$\frac{d}{dt}\underline{S}(t) + \beta\underline{S}(t) = \underline{X}(t)\underline{X}^T(t) \tag{15}$$

(note that $\frac{\partial}{\partial t}\underline{S}(t) = \frac{d}{dt}\underline{S}(t)$ as $\underline{S}(t)$ is independent of $\hat{\underline{\theta}}(t)$.

This formula can also be obtained by differentiating the non-recursive formula 5.4.2.

Initial conditions

Considering $J(\hat{\underline{\theta}}(t),t)$ at time t=0, it follows from the non-recursive solution that

$$\hat{\Theta}(0) = \hat{\underline{\theta}}_0 \tag{16}$$

and also that

$$J_2(\hat{\underline{\theta}}(t),\ 0) = \underline{S}_0 \tag{17}$$

5.5. THE RECURSIVE LEAST-SQUARES ALGORITHM

We are now in a position to state the continuous-time recursive-least-squares algorithm.

Recursive least-squares - inversion

The recursive least-squares algorithm is, from equations 14&15, defined by the pair of differential equations:

$$\underline{S}(t)\frac{d}{dt}\hat{\underline{\theta}}(t) = \underline{X}(t)\hat{e}(t) \tag{1}$$

$$\frac{d}{dt}\underline{S}(t) + \beta\underline{S}(t) = \underline{X}(t)\underline{X}^T(t) \tag{2}$$

and the algebraic equation

$$\hat{e}(t) = \Psi(t) - \hat{\Psi}(t) = \Psi(t) - \underline{X}^T(t)\hat{\underline{\theta}}(t) \tag{3}$$

with initial conditions:

$$\hat{\Theta}(0) = \hat{\underline{\theta}}_0;\ \underline{S}(0) = \underline{S}_0 \tag{4}$$

A disadvantage of this approach is that $\hat{\underline{\theta}}(t)$ does not appear explicitly; essentially $\underline{S}(t)$ must be inverted to obtain a solution. This problem is removed by the following reformulation.

Recursive least-squares - no inversion

Assuming $\underline{S}(t)$ is non-singular, the equations can be expressed directly in terms of $\underline{S}^{-1}(t)$ as

$$\frac{d}{dt}\hat{\underline{\theta}}(t) = \underline{S}^{-1}(t)\underline{X}(t)\hat{e}(t) \tag{5}$$

$$\frac{d}{dt}\underline{S}^{-1}(t) + \beta\underline{S}^{-1}(t) = \underline{S}^{-1}(t)\underline{X}(t)\underline{X}^{T}(t)\underline{S}^{-1}(t) \tag{6}$$

Note that, for numerical reasons, it is better to update the square root of $\underline{S}(t)$ rather than $\underline{S}(t)$ itself[8].

5.6. ANALYSIS OF RECURSIVE LEAST-SQUARES

The continuous-time recursive least-squares algorithm has some important properties which lead to robust self-tuning control. These properties are now derived.

The 'ideal' cost

For the purposes of this section, we shall define the ideal conditions for the estimator by having zero error e(t) and by having the correct initial estimate:

$$e(t) = 0;\ \hat{\underline{\theta}}_0 = \underline{\theta} \tag{1}$$

Such ideal conditions do not reflect a practical situation, but rather provide a basis for analysing the recursive least-squares algorithm operating under non-ideal conditions. With ideal conditions, the estimation error is given by:

$$\hat{e}(t,\tau) \triangleq \Psi(\tau) - \hat{\Psi}(\tau) = \underline{X}(\tau)\tilde{\underline{\theta}}(t) \tag{2}$$

where the error in the parameters $\tilde{\underline{\theta}}(t)$ is defined as

$$\underline{\tilde{\theta}}(t) \triangleq \underline{\theta} - \underline{\hat{\theta}}(t) \tag{3}$$

Under these conditions, the <u>ideal cost</u> (which will be called $J^\star(\underline{\hat{\theta}}(t),t)$) is given from 5.3.5 by

$$J^\star(\underline{\hat{\theta}}(t),t) = \tfrac{1}{2}e^{-\beta t}(\underline{\hat{\theta}}(t) - \underline{\hat{\theta}}_0)^T\underline{S}_0(\underline{\hat{\theta}}(t) - \underline{\hat{\theta}}_0) \tag{4}$$

$$+ \tfrac{1}{2}\int_0^t e^{-\beta(t-\tau)}(\underline{X}^T(\tau)\underline{\hat{\theta}}(t))^2 d\tau$$

$$= \tfrac{1}{2}\underline{\tilde{\theta}}(t)^T\underline{S}(t)\underline{\tilde{\theta}}(t)$$

Under these conditions, the ideal cost $J^\star(\underline{\theta}^\wedge(t),t)$ is given by the quadratic form $\tfrac{1}{2}\underline{\tilde{\theta}}(t)^T\underline{S}(t)\underline{\tilde{\theta}}(t)$. Its minimum value is clearly zero, corresponding to $\underline{\hat{\theta}}(t) = \underline{\theta}$.

Guided by this result, we <u>define</u> the quadratic function $V(t)$:

$$V(t) \triangleq \tfrac{1}{2}\underline{\tilde{\theta}}(t)^T\underline{S}(t)\underline{\tilde{\theta}}(t) \tag{5}$$

As we have shown, under ideal conditions $J^\star(\underline{\hat{\theta}}(t),t) = V(t)$. In the sequel, the behaviour of $V(t)$ under non-ideal conditions, but using the least-squares algorithm, will be found to be of interest.

To obtain a differential equation for $V(t)$, we first differentiate with respect to time to give:

$$\frac{d}{dt}V(t) = \tfrac{1}{2}\underline{\tilde{\theta}}(t)^T\frac{d}{dt}\underline{S}(t)\underline{\tilde{\theta}}(t) + \underline{\tilde{\theta}}(t)^T\underline{S}(t)\frac{d}{dt}\underline{\tilde{\theta}}(t) \tag{6}$$

Using the least-squares algorithm 5.5.1&2 and noting that

$$\frac{d}{dt}\underline{\tilde{\theta}}(t) = -\frac{d}{dt}\underline{\hat{\theta}}(t) \tag{7}$$

this becomes

$$\frac{d}{dt}V(t) = \frac{1}{2}\underline{\tilde{\theta}}(t)^T[-\beta\underline{S}(t) + \underline{X}(t)\underline{X}^T(t)]\underline{\tilde{\theta}}(t) \tag{8}$$

$$- \underline{\tilde{\theta}}(t)^T\underline{X}(t)\hat{e}(t)$$

At this stage, it is convenient to define the <u>parameter-induced error</u>

$$e^{\theta}(t) \triangleq \underline{\tilde{\theta}}(t)^T\underline{X}(t) \tag{9}$$

This gives

$$\frac{d}{dt}V(t) + \beta V(t) = \frac{1}{2}e^{\theta}(t)^2 - e^{\theta}(t)\hat{e}(t) \tag{10}$$

Now

$$\hat{e}(t) = \Psi(t) - \hat{\Psi}(t) \tag{11}$$

$$= (\Psi(t) - \underline{X}^T(t)\underline{\theta}) + (\underline{X}^T(t)\underline{\theta} - \underline{X}^T(t)\underline{\hat{\theta}}(t))$$

$$= e(t) + e^{\theta}(t)$$

So we can replace $e^{\theta}(t)$ by $\hat{e}(t) - e(t)$ to give

$$\frac{d}{dt}V + \beta V = \frac{1}{2}(\hat{e}(t) - e(t))^2 - (\hat{e}(t) - e(t))\hat{e}(t) \tag{12}$$

$$= \frac{1}{2}[e(t)^2 - \hat{e}(t)^2]$$

This gives the following <u>property of the ideal cost</u>

$$\frac{d}{dt}V + \beta V = \frac{1}{2}(e(t)^2 - \hat{e}(t)^2) \tag{13}$$

This is discussed in the following section.

<u>Properties</u>

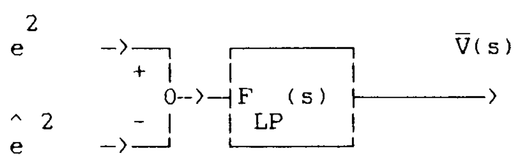
The equation

$$\frac{d}{dt}V + \beta V = \frac{1}{2}(e(t)^2 - \hat{e}(t)^2) \tag{14}$$

can be interpreted as follows: the (positive) ideal cost V is the output of the low-pass filter $F_{LP}(s)$ (Figure 5.6.1) with transfer function

$$F_{LP}(s) \triangleq \frac{1}{s + \beta} \tag{15}$$

with input $\frac{1}{2}[e(t)^2 - \hat{e}(t)^2]$ and initial condition $V(0)$.



<u>Figure 5.6.1 The low-pass filter</u>

If the two signals $e(t)$ and $\hat{e}(t)$ are <u>exponentially multiplied</u> (as in section 1.5) by $e^{\alpha t}$ to give $e_\alpha(t)$ and $\hat{e}_\alpha(t)$:

$$e_\alpha(t) \triangleq e^{\alpha t}\, e(t);\ \hat{e}_\alpha(t) \triangleq e^{\alpha t}\, \hat{e}(t) \tag{16}$$

then

$$e_\alpha^2(t) = e^{2\alpha t}\, e(t)^2;\quad \hat{e}_\alpha^2(t) = e^{2\alpha t}\, \hat{e}(t)^2 \tag{17}$$

Similarly define

$$V_\alpha(t) = e^{2\alpha t}\, V(t) \tag{18}$$

It follows from chapter 1, section 5, that the (positive) exponentially multiplied ideal cost $V_\alpha(t)$ is the output of the low-pass filter $F_{LP}(s - 2\alpha)$ with transfer function

$$F_{LP}(s - 2\alpha) = \frac{1}{s + \beta - 2\alpha} \tag{19}$$

with input $\frac{1}{2}(e_\alpha^2(t) - \hat{e}_\alpha^2(t))$ and initial condition $V(0)$. In particular, if

$$\alpha = \frac{1}{2}\beta \tag{20}$$

The low-pass filter becomes an integrator and

$$V_\alpha(t) = V(0) + \frac{1}{2}\int_0^t (e_\alpha^2(\tau) - \hat{e}_\alpha^2(\tau))d\tau \tag{21}$$

<u>The small gain property</u>

The estimator can be regarded as a single input single output system Ω with input $e(t)$ and output $\hat{e}(t)$ (Figure 5.6.2). We now derive a simple property of this system.

Noting that $V_\alpha(t) \geq 0$, it follows that

$$\frac{1}{2}\int_0^t \hat{e}_\alpha^2(\tau)d\tau \leq \frac{1}{2}\int_0^t e_\alpha^2(\tau)d\tau + V(0) \tag{22}$$

Intuitively, this expresses the fact that the integral over

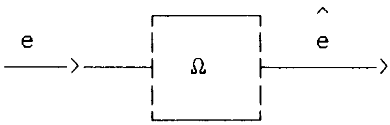


Figure 5.6.2 The estimator 'system'

time of the exponentially multiplied estimator squared 'output' $\hat{e}(t)$ is less than, or equal to, the integral over time of the exponentially multiplied estimator squared 'input' e(t) plus a constant.

Noting that

$$\frac{1}{2}\int_0^t e_\alpha^2(\tau)d\tau+V(0) \le \frac{1}{2}\int_0^t e_\alpha^2(\tau)d\tau+V(0)+\sqrt{\int_0^t e_\alpha^2(\tau)d\tau}.\sqrt{2V(0)} \qquad (23)$$

$$= \frac{1}{2}\left[\sqrt{\int_0^t e_\alpha^2(\tau)d\tau} + \sqrt{2V(0)}\right]^2$$

it follows that

$$\sqrt{\int_0^t \hat{e}_\alpha^2(\tau)d\tau} \le \sqrt{\int_0^t e_\alpha^2(\tau)d\tau} + \sqrt{2V(0)} \qquad (24)$$

In this sense (see[9,10] for details) the gain of the estimator system Ω is unity.

Ideal behaviour - estimates

Suppose that the external system is such that the signal $e(t)=0$, that is there are no neglected dynamics and no disturbances. As $\hat{e}_\alpha^2(t) \ge 0$, it follows that

$$V_\alpha(t) \leq V(0) \tag{25}$$

hence

$$V(t) \leq e^{-2\alpha t}\, V(0) \tag{26}$$

That is, the ideal cost V(t) is proportional to the initial cost V(0) and decays at least exponentially with time. Recalling that the quadratic function V(t) is

$$V(t) \triangleq \frac{1}{2}\underline{\tilde{\theta}}(t)^T \underline{S}(t)\underline{\tilde{\theta}}(t) \tag{27}$$

it follows that this result does not say much about the parameter estimate error $\underline{\tilde{\theta}}(t)$ unless the matrix $\underline{S}(t)$ is non-singular. However, if we <u>assume</u> the data-dependent <u>persistent excitation</u> condition

$$\underline{S}(t) > \Sigma > 0 \tag{28}$$

it follows that

1. $\underline{\tilde{\theta}}(t)$ is bounded.

2. $\underline{\tilde{\theta}}(t)$ converges to zero exponentially.

<u>Ideal behaviour - estimation error</u>

If e(t) = 0, the sole input to the lowpass filter $F_{LP}(s)$ is the signal $-\hat{e}(t)^2$. Hence the filter output V(t) can be written in terms of the filtered signal $\hat{e}_{LP}(t)$ representing the contribution of $\hat{e}(t)^2$ to the filter output:

$$\frac{d}{dt}\hat{e}_{LP}(t) + \alpha\hat{e}_{LP}(t) = \hat{e}(t)^2 \;;\; \hat{e}_{LP}(0) = 0 \tag{29}$$

as

$$V(t) = V(0) - \hat{e}_{LP}(t)^2 \qquad (30)$$

As the output V(t) of the filter must remain positive, it follows that the low-pass filtered signal $\hat{e}_{LP}(t)$ must be bounded by V(0):

$$\hat{e}_{LP}(t)^2 \leq V(0) \qquad (31)$$

This is not sufficient to ensure that $\hat{e}(t)$ is bounded (for example, passing a δ function into a low-pass filter gives a bounded output).

5.7. DISCRETE-TIME PARAMETER ESTIMATION

Digital implementation of the continuous-time estimator implies a sample rate similar to that of the corresponding digital controller. In this section, it is shown that discrete-time estimation of continuous-time parameters is possible[5,6] without introducing any sampling error. This allows the estimation sample rate to be divorced from the controller sample rate.

The-linear-in-the parameters model

The linear-in-the parameters model

$$\Psi(t) = \underline{X}^T(t)\underline{\theta} + e(t) \qquad (1)$$

is non-dynamic; it is just an algebraic relation. It may thus be sampled at any time t_m to give

$$\Psi_m = \underline{X}_m^T\underline{\theta} + e_m \qquad (2)$$

where

$$\Psi_m \triangleq \Psi(t_m); \ \underline{X}_m^T \triangleq \underline{X}^T(t_m); \ e_m = e(t_m) \qquad (3)$$

Note that this relation holds whether or not the samples t_m are equispaced or indeed in the correct order.

The Least-Squares Algorithm

The discrete-time least-squares algorithm appropriate to the discrete-time linear in the parameters model is well known and will not be derived here. See any of the textbooks[11,12,13,14,15] for details.

The parameter update algorithm is

$$\hat{\underline{\theta}}_{m+1} = \hat{\underline{\theta}}_m + \underline{S}_d{}^{-1})_m \underline{X}_m [\Psi_m - XTm\hat{\underline{\theta}}_m] \quad (4)$$

where the matrix $\underline{S}_d$ is given by

$$\underline{S}_{dm} = \beta_d \underline{S}_{dm-1} + \underline{X}_m \underline{X}_m{}^T \quad (5)$$

As discussed in the references ([8] in particular), the inverse, or the square-root of the inverse, of $\underline{S}_d$ is updated in practice. These exact discrete-time equations may be regarded as an approximation to the continuous-time equations. Assuming a constant sample interval Δ, the equations can be rewritten as

$$\frac{\hat{\underline{\theta}}_{m+1} - \hat{\underline{\theta}}_m}{\Delta} = (\Delta \underline{S}_{dm})^{-1} \underline{X}_m [\Psi_m - \underline{X}_m{}^T \hat{\underline{\theta}}_m] \quad (6)$$

where the matrix $\Delta\underline{S}_d$ is given by

$$\frac{\Delta\underline{S}_{dm} - \Delta\underline{S}_{dm-1}}{\Delta} = \frac{(\beta_d - 1)}{\Delta} \Delta\underline{S}_{dm-1} + \underline{X}_m \underline{X}_m{}^T \quad (7)$$

Regarding the left-hand side of each equation as an approximate time derivative, and comparing with equations 5.5.1&2, shows that:

$$\hat{\underline{\theta}}_m \cong \hat{\underline{\theta}}(t_m); \; \underline{S}_{dm} \cong \frac{1}{\Delta}\underline{S}(t_m); \; \beta_d \cong 1 - \Delta\beta \quad (8)$$

References

1. Kalman, R.E., "Design of a self-optimizing control system," Trans. ASME, vol. 80, p. 468, 1958.

2. Astrom, K.J. and Wittenmark, B., "On self-tuning regulators," Automatica, vol. 9, pp. 185-199, 1973.

3. Clarke, D.W. and Gawthrop, P.J., "Self-tuning controller," Proceedings IEE, vol. 122, no. 9, pp. 929-934, 1975.

4. Clarke, D.W. and Gawthrop, P.J., "Self-tuning control," Proceedings IEE, vol. 126, no. 6, pp. 633-640, 1979.

5. Young, P.C., "Process parameter estimation and self-adaptive control," in Theory of self-adaptive systems, ed. P.H. Hammond, Plenum Press, New York, 1966.

6. Young, P.C., "Parameter estimation for continuous-time models - A survey," Automatica, vol. 17, no. 1, pp. 23-39, 1981.

7. Gawthrop, P.J., "Parametric identification of transient signals," IMA Journal of Mathematical Control and Information, vol. 1, pp. 117-128, 1984.

8. Bierman, G.J., Factorization methods for discrete sequential estimation, Academic Press, New York, 1977.

9. Desoer, C.A. and Vidyasagar, M., Feedback systems : Input-output properties, Academic Press, London, 1975.

10. Vidyasagar, M., Input-output analysis of large-scale interconnected systems, Springer, Berlin, 1981.

11. Harris, C. and Billings, S., Self-tuning and adaptive control - theory and applications, Peter Peregrinus, London, 1981.

12. Ljung, L. and Soderstrom, T., Parameter identification, MIT Press., London, 1983.

13. Eykhoff, P., System identification, Wiley, 1974.

14. Astrom, K.J. and Wittenmark, B., Computer controlled systems, Prentice-Hall, 1984.

15. Goodwin, G.C. and Sin, K.S., Adaptive filtering prediction and control, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1984.

CHAPTER 6

Self-Tuning Control

Aims. To introduce a class of self-tuning controllers based on self-tuning emulators in a feedback loop. To distinguish between implicit and explicit methods. To distinguish between off-line and on-line emulator design. To show that some standard self-tuning methods, such as model-reference, generalised minimum variance, pole-placement and PID, are special cases of the more general class. To illustrate some self-tuning controllers using simulation.

6.1. INTRODUCTION

Self-tuning controllers (in the sense of this book) have two parts: a tunable feedback controller and a parameter identification based tuning method. Emulator-based feedback control has been considered in chapter 3 and least-squares identification has been considered in chapter 5. Putting these two ingredients together gives a self-tuning controller.

In chapter 3, it was found that the notion of an emulator embedded in a feedback loop unifies a number of apparently diverse control laws; they are all examples of an emulator within a feedback loop. In the same way, the notion of a self-tuning emulator in a feedback loop unifies a number of self-tuning controllers.

Astrom and Wittenmark[1] make the distinction between two types of self-tuning algorithm:

1. Explicit algorithms which explicitly identify the system parameters and then deduce the corresponding emulator parameters. These have also been called indirect methods.

2. Implicit algorithms which identify the emulator parameters directly; system parameters are implicit in the identified emulator parameters. These have also been called direct methods.

Implicit self-tuning control in a continuous-time setting has been considered by Egardt[2,3,4]. In particular, he unifies a number of algorithms and gives relations between self-tuning control and the classical model-reference approaches[5]. This chapter deals with implicit methods in the same spirit as Egardt; in particular, the intention is to unify a number of methods. The difference is that a wider class of algorithms is considered here and the self-tuning is based on recursive least-squares. The approach extends and amplifies that given in[6].

This twofold division of algorithms is not sufficient for the purpose of this book. We make the further distinction between on-line and off-line emulator design:

1. Off-line design. The emulator design parameters $P(s)$, $Z(s)$, $C(s)$ and T, the control weighting $Q(s)$ and the setpoint filter $R(s)$ are chosen off-line, that is before the self-tuning algorithm starts.

2. <u>On-line design</u>. Some, or all, of the emulator design parameters P(s), Z(s), C(s) and T, the control weighting Q(s) and the setpoint filter R(s) are automatically varied during self-tuning. There is <u>two-level tuning</u> taking place: both emulator parameters (G(s), F(s) etc.) and emulator design parameters are automatically tuned. The adjectives 'implicit' and 'explicit' refer to the former tuning process.

Examples of on-line emulator design in a discrete-time context are the algorithm of Allidina and Hughes[7] where P(s), Q(s) and R(s) are chosen on-line; and the discrete-time LQ method of Grimble[8] where the continuous-time equivalent is to choose the polynomial P(s) on-line via a spectral factorisation of the form:

$$P(s)P(-s) = B(s)B(-s) + \lambda A(s)A(-s) \quad (1)$$

where the system polynomials A(s) and B(s) are estimated on-line.

<u>Organisation of the chapter</u>

Section 2 considers feedback control in a self-tuning context and relates the algorithms to those of chapter 3. Section 3 considers system identification; that is a method of deriving system parameters using least-squares methods is given. Section 4 considers <u>explicit self-tuning</u> control; the system identification algorithms of section 3 are combined with the design methods of chapter 2. Section 5 introduces <u>implicit self-tuning</u> methods where emulator parameters are identified <u>without</u> identifying system parameters or using the design methods of chapter 2. The section is subdivided into off-line approaches where the emulator design parameters P(s), Z(s) and T, and the controller parameters Q(s) and R(s), are chosen a-priori, and on-line design methods where the emulator design parameters P(s), Z(s) and T, and the controller parameters Q(s) and R(s), are chosen on-line using an additional system identification stage. Section 6 provides some simulations.

6.2. FEEDBACK CONTROL

In chapter three, a range of non-adaptive feedback control algorithms is described and discussed. The feature common to all these controllers is that they may be described as an emulator in a feedback loop. The disadvantage of these non-adaptive controllers is that the system parameters (coefficients of A(s), B(s) and T) must be known if the desired performance is to be achieved. The aim of self-tuning control is to remove this restriction. In particular, the fixed emulator of chapter 3 is replaced by a self-tuning emulator.

The self-tuning controller is described by an equation identical to the non-adaptive controller of section 3.2 (equation 1) except that the emulator output $\bar{\phi}^{*}(s)$ is replaced by an estimated value $\hat{\phi}(s)$:

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \hat{\phi}(s)] \tag{1}$$

where

Symbol	Quantity
$\hat{u}(s)$	Control signal
$\hat{\phi}(s)$	self-tuning emulator output
$\bar{w}(s)$	setpoint
$Q(s)$	control weighting
$R(s)$	setpoint filter

$1/Q(s)$ and $R(s)$ are proper transfer functions. $\hat{\phi}(s)$ is the self-tuning emulator output corresponding to one of the emulators described in chapter 2. That is,

$$\hat{\phi}(s) = \begin{cases} \hat{\phi}_1(s) \\ \hat{\phi}_2(s) \\ \hat{\phi}_3(s) \\ \hat{\phi}_4(s) \end{cases} \quad \text{according to context} \tag{2}$$

where $\hat{\phi}(s)$ is the Laplace-transformed output of the appropriate self-tuning emulator

$$\hat{\phi}(t) = \underline{X}_e^T(t)\hat{\underline{\theta}}_e(t) \tag{3}$$

and $\underline{X}_e(t)$ and $\hat{\underline{\theta}}_e(t)$ are the appropriate emulator data vector and parameter estimate vector respectively.

6.3. SYSTEM IDENTIFICATION

Explicit self-tuning methods require estimates of the system parameters. The approach taken here is to write the system as its own emulator; the coefficients arising from the corresponding self-tuning emulator give the required system parameters. Most systems are subject to disturbances containing a constant component. If not properly accounted for, such disturbances can give rise to very poor parameter estimation; so this subject is given a section of its own.

This section is organised into the following subsections:

1. An emulator for the system
2. A self-tuning emulator
3. Non-zero mean disturbances.

An emulator for the system

Consider the particular case where the emulator is designed to emulate the system itself and that the delay T is zero; that is

$$\bar{\phi}(s) = \bar{y}(s) \tag{1}$$

The identity 2.2.2 then becomes

$$\frac{C(s)}{A(s)} = E(s) + \frac{F(s)}{A(s)} \tag{2}$$

If we make the choice deg(C) = deg(A)-1, the identity gives

$$E(s) = 0;\ F(s) = C(s) \tag{3}$$

giving

$$\bar{\phi}^{\star}(s) = \bar{y}(s) \tag{4}$$

which is not useful. If, however, we choose

$$C(s) = C_s(s) \tag{5}$$

$\deg(C_s(s))=\deg(A(s))$ and, in addition, choose the highest-order terms of $A(s)$ and $C_s(s)$ to be 1,

$$c_o = a_o = 1 \tag{6}$$

(this may always be done by suitably rescaling the disturbance), then the identity gives

$$E(s) = 1;\ F(s) = C_s(s) - A(s) \tag{7}$$

and so

$$\bar{\phi}^{\star}(s) = \frac{B(s)}{C_s(s)}\bar{u}(s) + \frac{C_s(s) - A(s)}{C_s(s)}\bar{y}(s) + \frac{I(s)}{C_s(s)} \tag{8}$$

Thus the system can be written as its own emulator; $\bar{\phi}^*(s)$ can be regarded as the system output $\bar{y}(s)$ minus the disturbance term $\bar{v}(s)$.

An example appears in chapter 5, section 2.

If the delay T is not zero but is known, the control signal $\bar{u}(s)$ can be replaced by a delayed version:

$$\bar{u}_T(s) = e^{-sT}\,\bar{u}(s) \tag{9}$$

in the above equations. As in section 2.5, we assume that the time-delay initial conditions are zero.

<u>A self-tuning emulator</u>

The system, rewritten as an emulator and including initial conditions associated with the rational part, can be written in the linear-in-the-parameters form of chapter 5 as

$$y(t) = \underline{X}_s^T(t)\underline{\theta}_s + e_s(t) \tag{10}$$

where the <u>data vector</u> $\underline{X}_s(t)$ and the <u>parameter vector</u> $\underline{\theta}_s$ are given, in Laplace-transform terms by

$$\bar{\underline{X}}_s(s) \triangleq \begin{vmatrix} \bar{\underline{X}}_u(s) \\ \bar{\underline{X}}_y(s) \\ \bar{\underline{X}}_i(s) \end{vmatrix}; \quad \underline{\theta}_s = \begin{vmatrix} \underline{\theta}_u \\ \underline{\theta}_y \\ \underline{\theta}_i \end{vmatrix} \tag{11}$$

where

$$\bar{\underline{X}}_u(s) = \frac{1}{C_s(s)}\begin{vmatrix} s^{n-1} \\ s^{n-2} \\ \vdots \\ 1 \end{vmatrix} e^{-sT}\,\bar{u}(s); \quad \bar{\underline{X}}_y(s) = \frac{1}{C_s(s)}\begin{vmatrix} s^{n-1} \\ s^{n-2} \\ \vdots \\ 1 \end{vmatrix}\bar{y}(s) \tag{12}$$

$$\bar{\underline{X}}_i(s) = \frac{1}{C_s(s)}\begin{vmatrix} s^{n-1} \\ s^{n-2} \\ . \\ 1 \end{vmatrix} \tag{13}$$

and $\underline{\theta}_s$ is given by

$$\underline{\theta}_u = \begin{vmatrix} b_1 \\ b_2 \\ . \\ b_n \end{vmatrix}; \ \underline{\theta}_y = \begin{vmatrix} c_1 - a_1 \\ c_2 - a_2 \\ . \\ c_n - a_n \end{vmatrix}; \ \underline{\theta}_i = \begin{vmatrix} i_1 \\ i_2 \\ . \\ i_n \end{vmatrix} \tag{14}$$

The vectors $\bar{\underline{X}}_u(s)$, $\bar{\underline{X}}_y(s)$ and $\bar{\underline{X}}_i(s)$ are the Laplace transforms of vectors in <u>controllable</u> <u>form</u> (see section 1.6). The time-domain versions may therefore be computed from the <u>differential</u> <u>equations</u> 1.6.1.

This linear-in-the-parameters model is suitable for the least-squares estimation algorithms of chapter 5:

$$\Psi(t) = \underline{X}^T(t)\underline{\theta} + e(t) \tag{15}$$

if we set

$$\Psi(t) = y(t); \ \underline{X}(t) = \underline{X}_s(t); \ \underline{\theta} = \underline{\theta}_s; \ e(t) = e_s(t) \tag{16}$$

The coefficients b_i of B(s), and i_i of I(s) are identified directly; the coefficients a_i of A(s) are obtained by subtracting the appropriate entries of $\underline{\theta}$ from the <u>known</u> coefficients c_i of $C_s(s)$.

The advantages of including initial condition terms in parameter estimation is discussed in detail elsewhere[9,10].

Non-zero mean disturbances

As pointed out in chapters 1 and 3, the almost inevitable non-zero mean component of a disturbance can be included in the system model by assuming that

$$A(s) = sA_0(s);\ B(s) = sB_0(s) \tag{17}$$

With this assumption, the system emulator becomes

$$\bar{\phi}^{\star}(s) = \frac{sB_0(s)}{C_s(s)}\bar{u}(s) + \frac{s(C_0(s) - A_0(s)) + c_n}{C_s(s)}\bar{y}(s) + \frac{I(s)}{C_s(s)} \tag{18}$$

where

$$C_s(s) = c_n + sC_0(s) \tag{19}$$

This can be written in linear-in-the-parameters form as

$$y_0(t) = \underline{X}_0^T(t)\underline{\theta}_{so} + e_{so}(t) \tag{20}$$

where $\bar{y}_0(s)$ is the high-pass filtered system output

$$\bar{y}_0(s) \triangleq s\frac{C_0(s)}{C_s(s)}\bar{y}(s) \tag{21}$$

the data vector $\bar{\underline{X}}_0(s)$ and the parameter vector $\underline{\theta}_{so}$ are now given by

$$\bar{\underline{X}}_0(s) \triangleq \begin{vmatrix} \bar{\underline{X}}_{uo}(s) \\ \bar{\underline{X}}_{yo}(s) \\ \bar{\underline{X}}_{io}(s) \end{vmatrix};\ \underline{\theta} = \begin{vmatrix} \underline{\theta}_{uo} \\ \underline{\theta}_{yo} \\ \underline{\theta}_{io} \end{vmatrix} \tag{22}$$

where

$$\bar{\underline{X}}_{uo}(s) = \frac{1}{C_s(s)}\begin{vmatrix} s^{n-1} \\ s^{n-2} \\ . \\ s \end{vmatrix} e^{-sT}\bar{u}(s);\ \bar{\underline{X}}_{yo}(s) = \frac{1}{C_s(s)}\begin{vmatrix} s^{n-1} \\ s^{n-2} \\ . \\ s \end{vmatrix}\bar{y}(s) \tag{23}$$

$$\bar{\underline{X}}_{io}(s) = \frac{1}{C_s(s)}\begin{vmatrix} s^{n-1} \\ s^{n-2} \\ . \\ 1 \end{vmatrix}$$

and $\underline{\theta}$ is given by

$$\underline{\theta}_{uo} = \begin{vmatrix} b_1 \\ b_2 \\ . \\ b_{n-1} \end{vmatrix}; \quad \underline{\theta}_{yo} = \begin{vmatrix} c_1 - a_1 \\ c_2 - a_2 \\ . \\ c_{n-1} - a_{n-1} \end{vmatrix} \tag{24}$$

The vectors $\bar{\underline{X}}_u(s)$, $\bar{\underline{X}}_y(s)$ and $\bar{\underline{X}}_i(s)$ are the Laplace transforms of vectors in <u>controllable</u> <u>form</u> (see section 1.6). The time-domain versions may therefore be computed from the <u>differential</u> <u>equations</u> 1.6.1. This linear-in-the-parameters model is of the correct form for the least-squares estimation algorithms of chapter 5:

$$\Psi(t) = \underline{X}^T(t)\underline{\theta} + e(t) \tag{25}$$

if we set

$$\Psi(t) = y_0(t); \ \underline{X}(t) = \underline{X}_{so}(t); \ \underline{\theta} = \underline{\theta}_{so}; \ e(t) = e_{so}(t) \tag{26}$$

where

$$\bar{y}_0(s) = s\frac{C_0(s)}{C_s(s)}\bar{y}(s) \tag{27}$$

Both sides of this equation comprise high-pass filtered quantities, but note that the <u>same</u> system parameters are to be found in $\underline{\theta}_{so}$ as in $\underline{\theta}_s$. The importance of using this <u>zero-gain</u> <u>emulator</u> in practice cannot be overstated. See[11] for a discussion of this point from a discrete-time point of view and[12,6] for a discussion from the continuous-time point of view.

It is also emphasised that the use of high-pass filtering in this context, because it arises naturally from the system model, does not involve any approximation.

An example appears in chapter 5, section 2. The simulation examples 7 and 9 of section 6.6.2 illustrate the advantages of the zero-gain method.

6.4. EXPLICIT SELF-TUNING CONTROL

The adjective 'explicit' implies that the system parameters corresponding to A(s) and B(s) are estimated on-line, and these estimates (together with the polynomials P(s), Z(s) and C(s) are then used to design the emulator on-line. The self-tuning system emulator provides these system parameters. There are two types of explicit algorithm:

1. Off-line design. The emulator design parameters P(s), Z(s), C(s) and T, the control weighting Q(s) and the setpoint filter R(s) are chosen off-line, that is before the self-tuning algorithm starts.

2. On-line design. Some, or all, of the emulator design parameters P(s), Z(s), C(s) and T, the control weighting Q(s) and the setpoint filter R(s) are automatically varied during self-tuning.

These are considered in separate subsections. Each type of algorithm has two phases of operation:

1. The off-line (a-priori) design phase. This occurs before tuning starts.

2. The on-line tuning phase.

6.4.1. Off-line design

<u>The off-line (a-priori) design phase</u>

1. Choose the emulator polynomials $P(s)$, $Z^+(s)$, $Z^-(s)$, $C(s)$ and the delay T.

2. Choose the weighting filter $Q(s)$.

3. Choose the setpoint filter $R(s)$.

4. Choose the system order.

<u>The on-line tuning phase</u>

1. Update the system data vector $\underline{X}_s(t)$ (or $\underline{X}_{so}(t)$) as in section 6.3.

2. Update the system parameter estimate vector $\hat{\underline{\theta}}_s(t)$ of $\underline{\theta}_s$ (or $\hat{\underline{\theta}}_{so}(t)$ of $\underline{\theta}_{so}$) using either the continuous or discrete algorithms of chapter 5.

3. Use an appropriate emulator design algorithm from chapter 2 to generate the parameters of the required emulator from the estimated system parameters. These are placed in the the vector $\hat{\underline{\theta}}_e(t)$ as an approximation to the ideal emulator vector $\underline{\theta}_e$.

4. Generate the emulator data vector $\underline{X}_e(t)$ as in section 2.7. If the same denominator polynomial is used for both the system emulator and the emulator ($C(s) = C_s(s)$) and so $\underline{X}_e(t) = \underline{X}_s(t)$, this step may be omitted.

5. Generate the emulated signal $\hat{\phi}(t)$ using (see equation 2.7.9) $\hat{\phi}(t) = \underline{X}_e^T(t)\hat{\underline{\theta}}_e(t)$.

6. Generate the control signal as in section 6.2. In Laplace-transform terms, this is:

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \hat{\phi}(s)] \qquad (1)$$

6.4.2. On-line design

The off-line (a-priori) design phase

1. Choose a design rule giving the emulator design polynomials $P(s)$, $Z^+(s)$, $Z^-(s)$, $C(s)$ and the delay T in terms of the system parameters. For example, a pole-placement design rule would be to choose

$$Z(s) = \frac{B(s)}{B(0)} \qquad (1)$$

   and to choose the other polynomials a-priori.

2. Choose a design rule weighting filter $Q(s)$ in terms of system parameters.

3. Choose a design rule giving the setpoint filter $R(s)$ in terms of system parameters.

4. Choose the system order.

In practice, some of these rules can be purely a-priori. Thus, for example, $Q(s)$ and $R(s)$ could be chosen a-priori. If all the rules are, in fact, a-priori, then the on-line design reduces to the off-line design.

The on-line tuning phase

1. Update the system data vector $\underline{X}_s(t)$ (or $\underline{X}_{so}(t)$) as in section 6.3.

2. Update the system parameter estimate vector $\hat{\underline{\theta}}_s(t)$ of $\underline{\theta}_s$ (or $\hat{\underline{\theta}}_{so}(t)$ of $\underline{\theta}_{so}$) using either the continuous or

discrete algorithms of chapter 5.

3a. From the estimated system parameters, derive the corresponding emulator design parameters P(s), $Z^+(s)$, $Z^-(s)$, C(s) and the delay T in terms of the estimated system parameters.

3b. From the estimated system parameters, derive the corresponding control weighting transfer function Q(s) in terms of system parameters.

3c. From the estimated system parameters, derive the corresponding setpoint filter transfer function R(s) in terms of system parameters.

3d. Use an appropriate emulator design algorithm from chapter 2 to generate the parameters of the required emulator from the estimated system parameters. These are placed in the the vector $\hat{\underline{\theta}}_e(t)$ as an approximation to the ideal emulator vector $\underline{\theta}_e$.

4. Generate the emulator data vector $\underline{X}_e(t)$ as in section 2.7. If the same denominator polynomial is used for both the system emulator and the emulator ($C(s) = C_s(s)$) and so $\underline{X}_e(t) = \underline{X}_s(t)$, this step may be omitted.

5. Generate the emulated signal $\hat{\phi}(t)$ using (see equation 2.7.9) $\hat{\phi}(t) = \underline{X}_e^T(t)\hat{\underline{\theta}}_e(t)$.

6. Generate the control signal as in section 6.2. In Laplace-transform terms, this is

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \hat{\phi}(s)] \tag{2}$$

This differs from the off-line design in that the additional on-line steps 3a-3c are added; 3d is as step 3 of the off-line design.

6.5. IMPLICIT SELF-TUNING CONTROL

Implicit self-tuning control avoids the separate design phase by identifying the emulator parameters directly.

Tuning the emulator

As discussed in chapter 2, the emulator can be written in linear-in-the-parameters form as:

$$\phi(t) = \underline{X}_e^T(t)\underline{\theta}_e + e^{\star}(s) \tag{1}$$

In many emulators, $\phi(t)$ is not a realisable quantity, but can be made so by appending a realisability filter $\Lambda(s)$ to give a realisable signal $\phi_\Lambda(t)$:

$$\bar{\phi}_\Lambda(s) = \Lambda(s)\bar{\phi}(s) \tag{2}$$

such that

$$e^{sT}\frac{P(s)}{Z(s)}\Lambda(s) \text{ is realisable and proper} \tag{3}$$

As will be seen in chapter 7, we will also require that the inverse be proper:

$$e^{-sT}\frac{Z(s)}{P(s)}\Lambda(s)^{-1} \text{ is realisable and proper} \tag{4}$$

(As this filter is under our control, we may choose the initial conditions associated with $\Lambda(s)$ to be zero; this will be assumed in the sequel).

One possibility is to choose

$$\Lambda(s) = e^{-sT}\frac{Z(s)}{P(s)} \tag{5}$$

giving

$$\Phi_\Lambda(t) = y(t) \tag{6}$$

The corresponding linear-in-the-parameters model is then

$$\Phi_\Lambda(t) = \underline{X}_\Lambda{}^T(t)\underline{\theta} + e_\Lambda(t) \tag{7}$$

where

$$\bar{\underline{X}}_\Lambda(s) \triangleq \Lambda(s)\bar{\underline{X}}(s);\ \bar{e}_\Lambda(s) \triangleq \Lambda(s)\bar{e}(s) \tag{8}$$

Note that $\bar{\underline{X}}_\Lambda(s)$ can be generated in the same way as $\bar{\underline{X}}_e(s)$ except that the signals $\bar{u}(s)$ and $\bar{y}(s)$ are prefiltered by $\Lambda(s)$.

<u>Example 1</u>

If P(s) = Z(s) = 1, and equation 5 is used, then

$$\Lambda(s) = e^{-sT}\ ;\ \bar{\underline{X}}_\Lambda(s) = e^{-sT}\ \bar{\underline{X}}(s) \tag{9}$$

so

$$\underline{X}_\Lambda(t) = \underline{X}(t-T) \tag{10}$$

This corresponds to many discrete-time algorithms, including the self-tuning regulator[13].

<u>Example 2</u>

If Z(s) = 1 and T=0, the filtering effect of $\Lambda(s)$ is closely related to the filtering approach discussed by Egardt in chapter 3 of his book[2].

This linear-in-the-parameters model is suitable for the least-squares estimation algorithms of chapter 5:

$$\Psi(t) = \underline{X}^T(t)\underline{\theta} + e(t) \tag{11}$$

if we set

$$\Psi(t) = \phi_{\Lambda}(t);\ \underline{X}(t) = \underline{X}_{\Lambda}(t);\ e(t) = e_{\Lambda}(t) \tag{12}$$

There are two types of implicit algorithm:

1. Off-line design. The emulator design parameters P(s), Z(s), C(s) and T, the control weighting Q(s) and the setpoint filter R(s) are chosen off-line, that is before the self-tuning algorithm starts.

2. On-line design. Some, or all, of the emulator design parameters P(s), Z(s), C(s) and T, the control weighting Q(s) and the setpoint filter R(s) are automatically varied during self-tuning.

These are considered in separate subsections. Each type of algorithm has two phases of operation:

1. The off-line (a-priori) design phase. This occurs before tuning starts.

2. The on-line tuning phase.

6.5.1. Off-line design

The off-line (a-priori) design phase

1. Choose the emulator polynomials P(s), $Z^{+}(s)$, $Z^{-}(s)$, C(s) and the delay T.

2. Choose the weighting filter Q(s).

3. Choose the setpoint filter R(s).

4. Choose the system order.

5. Choose the realisability filter Λ(s) according to equations 6.5.3&4. Typically we would use equation 6.5.5:

$$\Lambda(s) = e^{-sT}\frac{Z(s)}{P(s)} \tag{1}$$

Steps 1 and 5 may not always be possible. For example, if pole-placement is to be used and so $Z(s) = B(s)$, these steps are not possible unless $B(s)$ is known a-priori.

The on-line tuning phase

1. Generate the quantity $\phi_\Lambda(t)$, where $\bar{\phi}_\Lambda(s) = \Lambda(s)\bar{\phi}(s)$.

2. Filter the control signal $u(t)$ and the system output $y(t)$ by $\Lambda(s)$.

3. Generate the emulator data vector $\underline{X}_\Lambda(t)$ using the filtered signals from step 2 together with differential equations 1.6.1.

4. Update the emulator parameter estimate vector $\hat{\underline{\theta}}_e(t)$ using either the continuous or discrete algorithms of chapter 5 and based on the linear-in-the-parameters model of equations 5.2.9&10.

5. Generate the emulated signal $\hat{\phi}(t)$ using (see equation 2.7.9) $\hat{\phi}(t) = \underline{X}_e^T(t)\hat{\underline{\theta}}_e(t)$.

6. Generate the control signal as in section 6.2. In Laplace-transform terms, this is

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \hat{\phi}(s)] \qquad (2)$$

6.5.2. On-line design

The off-line (a-priori) design phase

1. Choose a design rule giving the emulator design polynomials $P(s)$, $Z^+(s)$, $Z^-(s)$, $C(s)$ and the delay T in terms of the system parameters. For example, a pole-placement

design rule would be to choose

$$Z(s) = \frac{B(s)}{B(0)} \tag{1}$$

and to choose the other polynomials a-priori.

2. Choose a design rule weighting filter Q(s) in terms of system parameters.

3. Choose a design rule giving the setpoint filter R(s) in terms of system parameters.

4. Choose the system order.

5. Choose a design rule giving the realisability filter $\Lambda(s)$ in terms of the system parameters and the emulator design parameters according to equations 6.5.3&4. Typically we would use equation 6.5.5:

$$\Lambda(s) = e^{-sT}\frac{Z(s)}{P(s)} \tag{2}$$

In practice, some of these rules can be purely a-priori. Thus, for example, Q(s) and R(s) could be chosen a-priori. If all the rules are, in fact, a-priori, then the on-line design reduces to the off-line design.

The on-line tuning phase

1. Update the system data vector $\underline{X}_s(t)$ (or $\underline{X}_{so}(t)$) as in section 6.3.

2. Update the system parameter estimate vector $\hat{\underline{\theta}}_s(t)$ of $\underline{\theta}_s$ (or $\hat{\underline{\theta}}_{so}(t)$ of $\underline{\theta}_{so}$) using either the continuous or discrete algorithms of chapter 5.

3. From the estimated system parameters, derive the corresponding emulator design parameters P(s), $Z^+(s)$,

$Z^{-}(s)$, $C(s)$ and the delay T in terms of the estimated system parameters.

4. From the estimated system parameters, derive the corresponding control weighting transfer function $Q(s)$ in terms of system parameters.

5. From the estimated system parameters, derive the corresponding setpoint filter transfer function $R(s)$ in terms of system parameters.

6. Deduce the realisability filter $\Lambda(s)$ in terms of the estimated system parameters and the derived values of $P(s)$ and $Z(s)$.

7. Generate the quantity $\phi_\Lambda(t)$, where $\bar{\phi}_\Lambda(s) = \Lambda(s)\bar{\phi}(s)$.

8. Filter the control signal $u(t)$ and the system output $y(t)$ by $\Lambda(s)$.

9. Generate the emulator data vector $\underline{X}_\Lambda(t)$ using the filtered signals from step 2 together with differential equations 1.6.1.

10. Update the emulator parameter estimate vector $\hat{\underline{\theta}}_e(t)$ using either the continuous or discrete algorithms of chapter 5 and based on the linear-in-the-parameters model 5.2.6&7.

11. Generate the emulated signal $\hat{\phi}(t)$ using $\hat{\phi}(t) = \underline{X}_e^T(t)\hat{\underline{\theta}}_e(t)$.

12. Generate the control signal as in section 6.2. In Laplace-transform terms, this is

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \hat{\phi}(s)] \tag{3}$$

This differs from the off-line design in that the additional on-line steps 1-6 are added. At first sight, this looks to be more complex than an explicit algorithms. But in fact it is simpler in that the emulator polynomials G(s) and F(s) are not deduced on line but are rather identified directly.

6.6. SOME SIMULATED EXAMPLES

In this section, a number of simulated illustrative examples are given. The simulations are divided into two sections: algorithms using the realisability filter $\Lambda(s)$ and those which do not.

6.6.1. Using realisability filter

A number of versions of self-tuning algorithms using

$$\Lambda(s) = \frac{Z(s)}{P(s)} \qquad (1)$$

were simulated using the SIMNON language[14,15]. All the examples in this section have the following in common:

1. Four emulator parameters are identified.

2. The initial $\underline{S}^{-1}(t)$ matrix is, in each case, given by:

$$S^{-1}(0) = \begin{vmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{vmatrix} \qquad (2)$$

3. $C(s) = 1+0.5s$.

4. All examples are detuned versions of the underlying algorithms with $Q(s) = \frac{0.01s}{1+0.1s}$.

5. $A(s) = s(1+s)$.

6. The realisability filter is given by $\Lambda(s) = \frac{Z(s)}{P(s)}$.

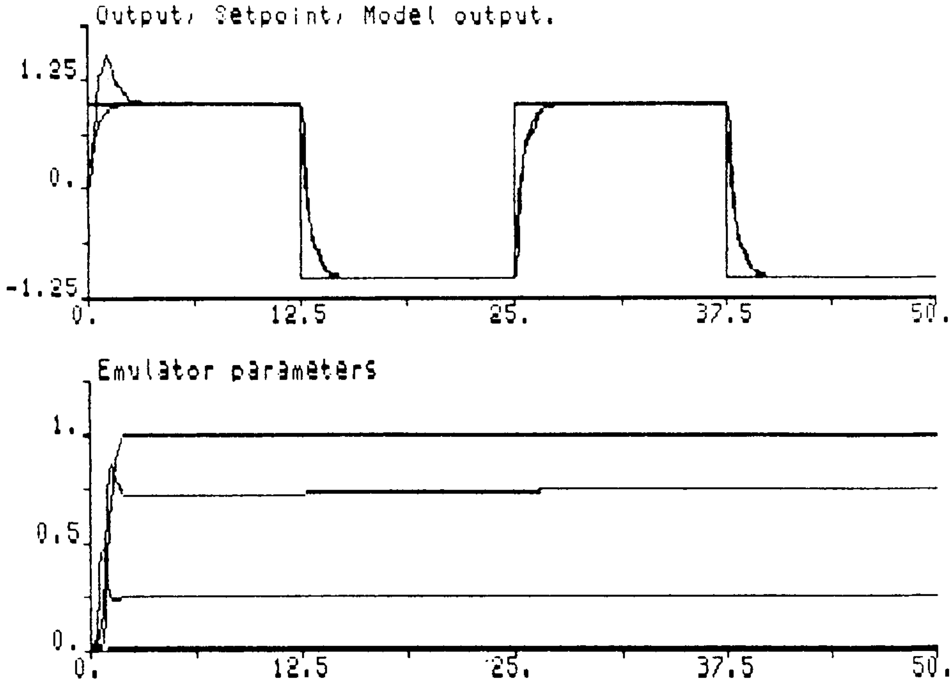


Figure 6.6.1.1 Example 1

7. The algorithms are simulated with a noise-free system having no neglected dynamics for 50 time units.

8. The upper graphs in Figures 6.6.1.1-5 show the setpoint (a square wave between +1 and -1 with a period of 25 units), the actual system output, and the model output. The model output $\bar{y}_m(s)$ corresponds to

$$\bar{y}_m(s) = \frac{Z(s)}{P(s)}\bar{w}(s) \qquad (3)$$

9. The lower graph of Figures 6.6.1.1-5 shows the evolution of the four emulator parameters with respect to time.

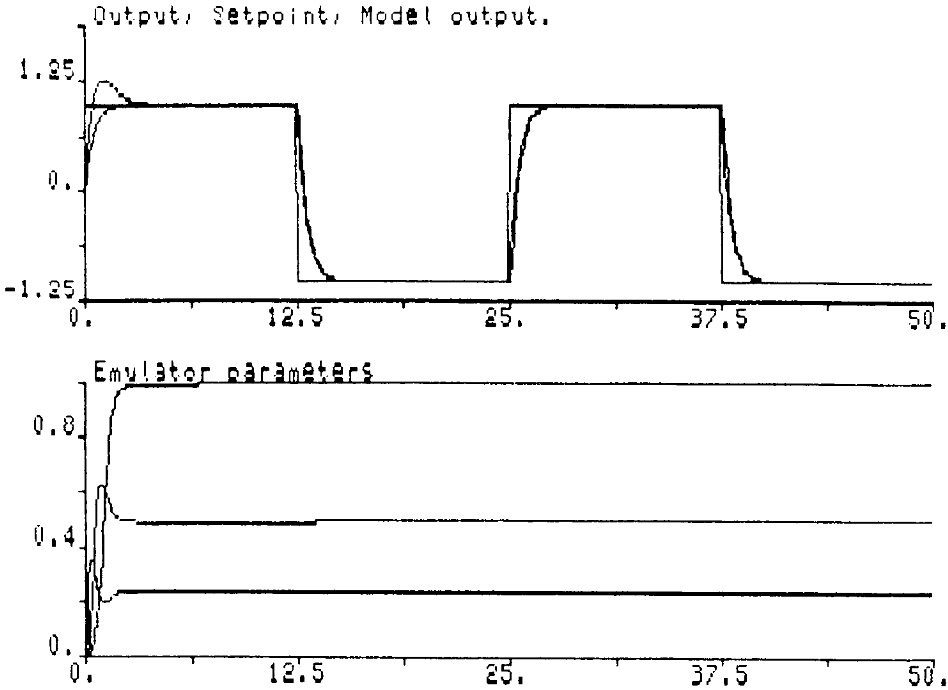


Figure 6.6.1.2 Example 2

10. In each case, the presence of the non-zero Q(s) control weighting prevents the system output following the model-output exactly. But note that the discrepancy is zero at zero frequency (constant setpoint) and only appears at high frequencies (changing setpoint).

Figures 6.6.1.1-5 correspond to examples 1-5 of this section. The differences between the five examples are summarised in the following Table:

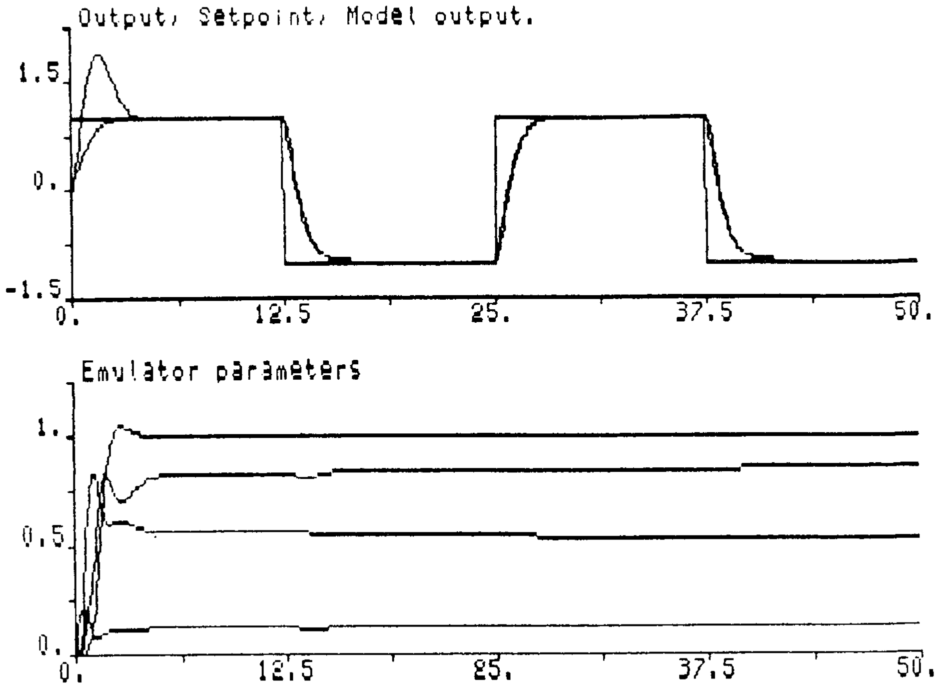


Figure 6.6.1.3 Example 3

SIMULATION SUMMARY					
No.	Method	P(s)	Z(s)	B(s)	Design
1	Model reference	1+0.5s	1	1+0.1s	Off-line
2	Model reference	1+0.5s	1	1+s	Off-line
3	Pole placement	$(1+0.5s)^2$	B(s)	1+0.1s	On-line
4	Pole placement	$(1+0.5s)^2$	B(s)	1+s	On-line
5	Pole placement	$(1+0.5s)^2$	B(s)	1-s	On-line

See chapter 3 for a discussion of these examples in a non-adaptive context.

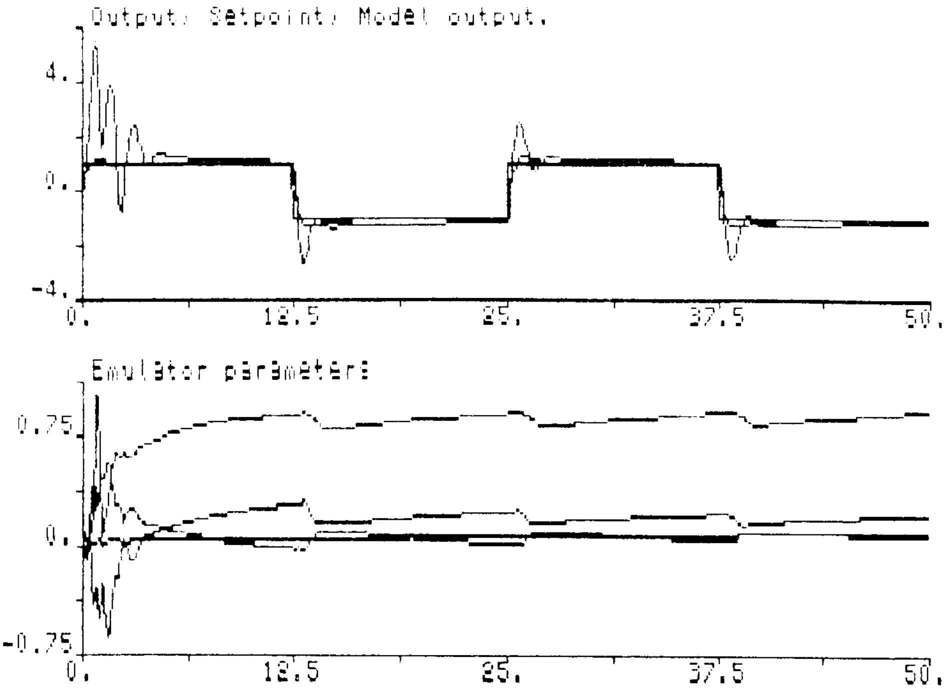


Figure 6.6.1.4 Example 4

Remarks

1. Examples 1 and 2 can use off-line design, as P(s) and Z(s) are both chosen a-priori. Examples 3, 4 and 5 cannot, as Z(s) = B(s) is not known a-priori.

2. The systems in examples 1-4 are minimum phase and so either model-reference or pole-placement design is appropriate. The system in example 5 has a zero at s=1; model-reference control is not possible in this case, but pole-placement is. Note the characteristic non-minimum phase step response of the closed-loop system in example 5.

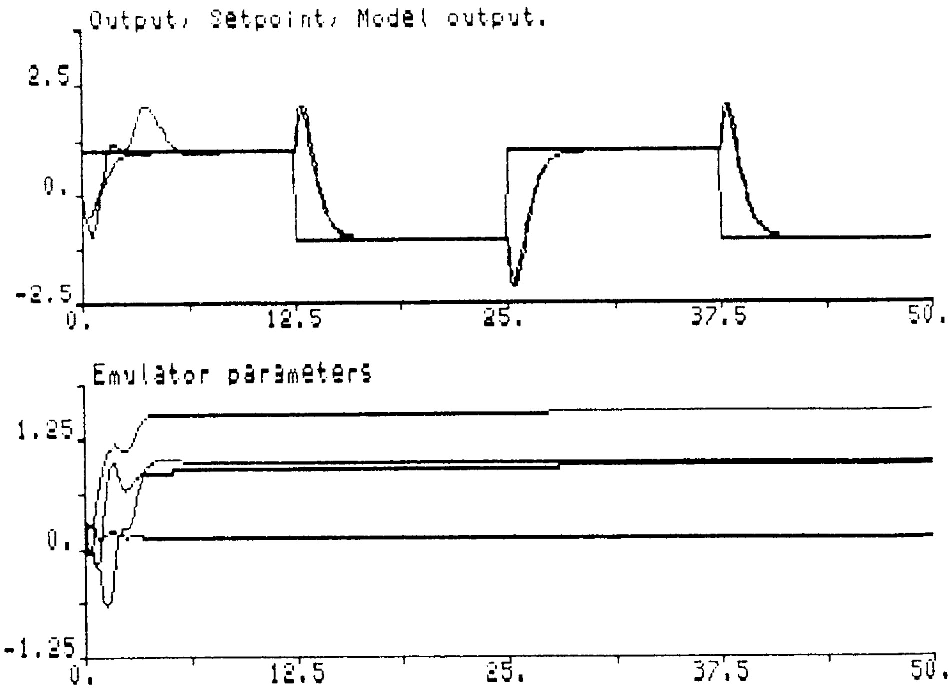


Figure 6.6.1.5 Example 5

3. The system in examples 2 and 4 is

$$\frac{1+s}{s(1+s)} = \frac{1}{s} \tag{4}$$

Thus the apparently second-order system is in fact first order. It can be represented as a second-order system with a first order cancelling factor of the form:

$$\frac{a+s}{a+s} \tag{5}$$

for <u>any</u> values of a. (Note that the coefficient of s is unity, as it is assumed that the coefficient of the highest-order s term is unity as in equation 6.2.6) Thus, in each case, the estimated parameters do not have a unique "true" value. This is revealed in the estimated parameters. In example 4, the desired closed loop system is not unique, as Z(s) = B(s)/B(0) = 1+s/a. In fact the estimator ends up with

$$a = 0.55 \tag{6}$$

in this particular simulation. Note that the model output in this case assumes that B(s) = Z(s) = 1+s and so is different from what is actually achieved.

6.6.2. Not using realisability filter

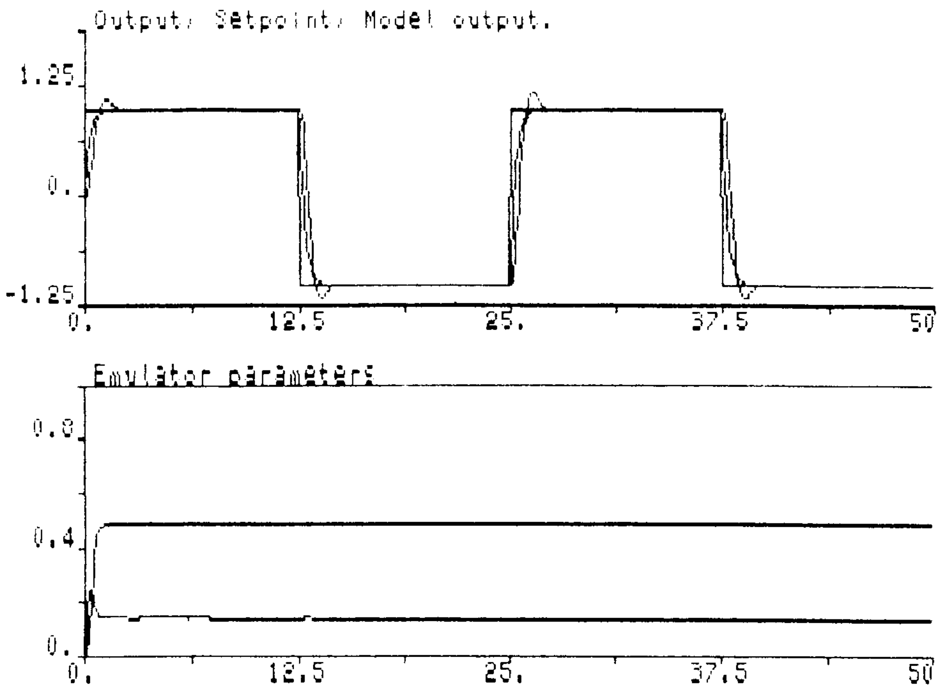


Figure 6.6.2.1 Example 6

A number of versions of self-tuning algorithms using

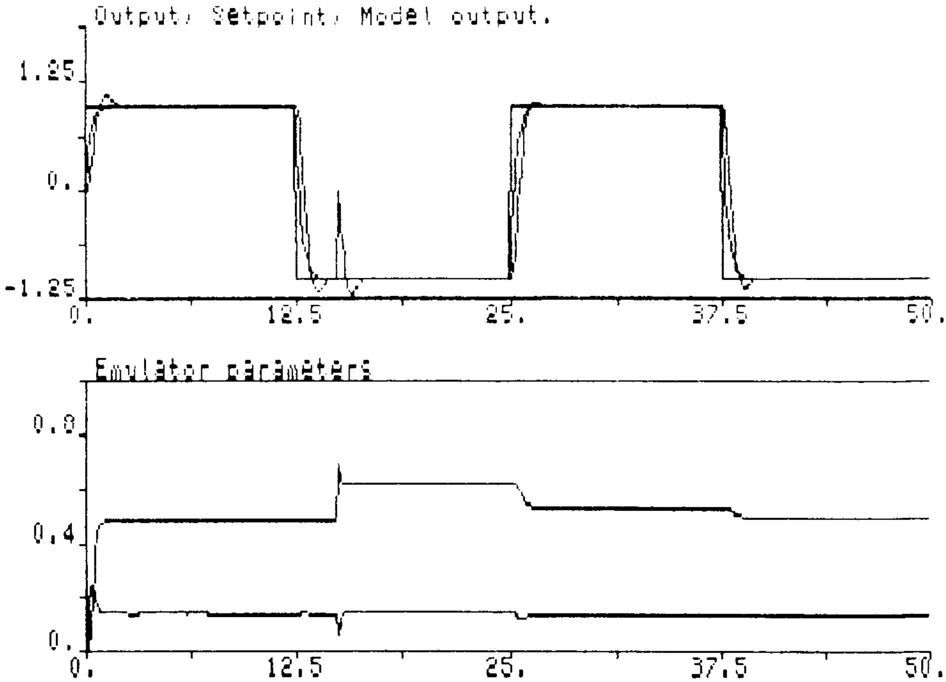


Figure 6.6.2.2 Example 7

$$\Lambda(s) = 1 \tag{1}$$

were simulated using the SIMNON language[14,15]. All examples have the following in common:

1. Two emulator parameters are identified.

2. The initial $\underline{S}^{-1}(t)$ matrix is, in each case, given by:

$$S^{-1}(0) = \begin{vmatrix} 100 & 0 \\ 0 & 100 \end{vmatrix} \tag{2}$$

3. In each case, the emulator design parameters are:

$$P(s) = 1+0.3s; \; Z(s) = Z^{-}(s) = 1+0.03s \tag{3}$$

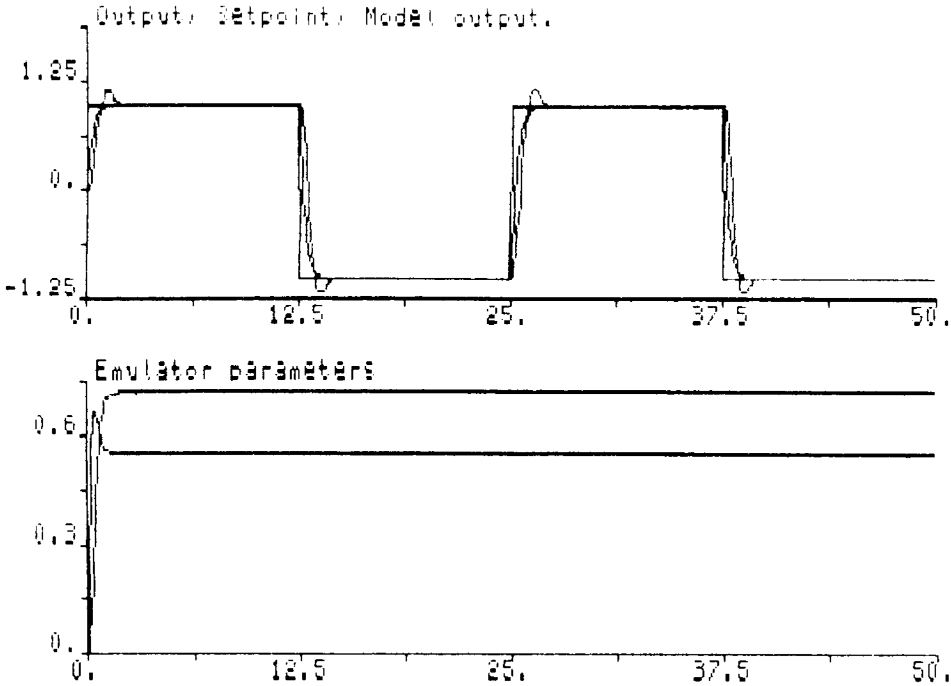


Figure 6.6.2.3 Example 8

See section 3.11 for a discussion of the ideas behind this strategy. Note that $\frac{P(s)}{Z(s)}$ is realisable and so $\Lambda(s) = 1$ may be used here.

4. All examples are <u>detuned</u> versions of the underlying model-reference algorithm with $Q(s) = \frac{0.2s}{1+0.1s}$.

5. The algorithms are simulated using a system having no neglected dynamics for 50 time units. Examples 7 and 9 have a unit output step disturbance occurring at time=15 units; that is, one is added to the system output from time 15 onwards.

6. The upper graph of Figures 6.6.2.1-4 shows the setpoint (a square wave between +1 and -1 with a period of 25

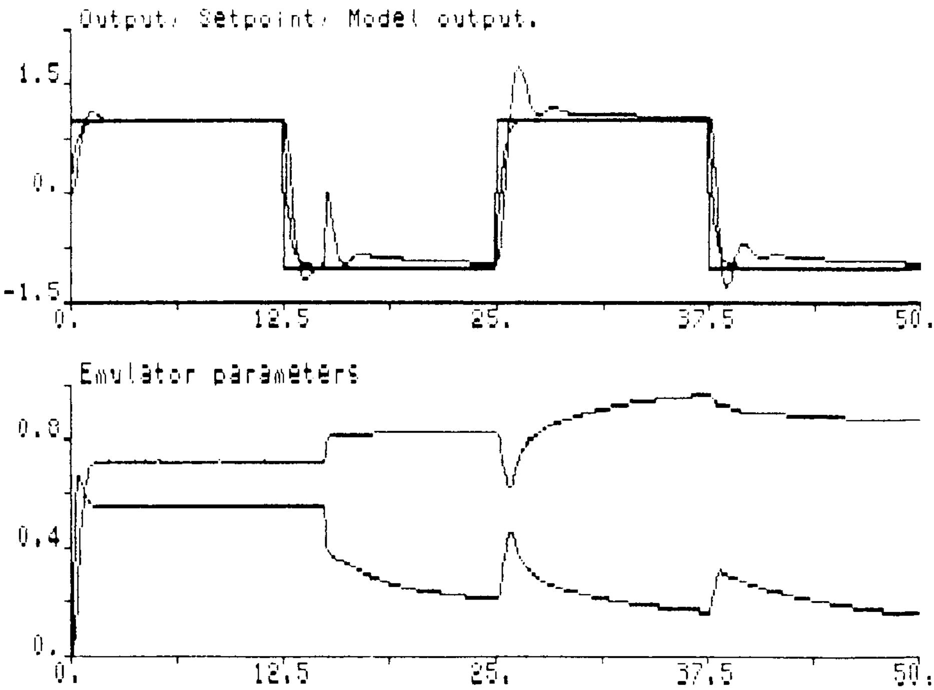


Figure 6.6.2.4 Example 9

units), the actual system output, and the model output. The model output corresponds to

$$\bar{y}_m(s) = \frac{Z(s)}{P(s)}\bar{w}(s) \tag{4}$$

7. The lower graph of Figures 6.6.2.1-4 shows the evolution of the two emulator parameters with respect to time.

Figures 6.6.2.1-4 correspond to examples 6-9 of this section. The differences between the four examples are summarised in the following Table:

SIMULATION SUMMARY				
No.	A(s)	B(s)	C(s)	Disturbance
6	s(1+s)	2s	1+0.3s	No
7	s(1+s)	2s	1+0.3s	Yes
8	1+s	2	1	No
9	1+s	2	1	Yes

See chapter 3 for a discussion of these examples in a non-adaptive context.

Remarks

1. In each case, the presence of the non-zero Q(s) control weighting prevents the system output following the model-output exactly. But note that the discrepancy is zero at zero frequency (constant setpoint) and only appears at high frequencies (changing setpoint).

2. The self-tuning emulators used in examples 6 and 7 are designed on the basis of a system with a cancelling s term - they have integral action. This does not make much difference between examples 6 and 8 which have no step disturbance. Example 7 illustrates the superior performance when non-zero mean disturbances are assumed a-priori as compared with example 9. A similar effect may be observed when using the realisability filter.

3. Despite the different controller structure, examples 6 and 8 end up with the same closed-loop setpoint response, though the disturbance response is different, as discussed in remark 2.

References

1. Astrom, K.J. and Wittenmark, B., "Self-tuning controllers based on pole-zero placement," Proceedings IEE Pt.D., vol. 127, pp. 120-130, 1980.

2. Egardt, B., Stability of adaptive controllers, Springer, 1979.

3. Egardt, B., "Unification of some continuous-time adaptive control schemes," IEEE Trans. Autom. Control, vol. AC-24, p. 588, 1979.

4. Egardt, B., "Stability analysis of continuous-time adaptive control systems," SIAM J. Control & Optimiz., vol. 18, p. 540, 1980.

5. Landau, I.D., Adaptive control: The model reference approach, Marcel Dekker, New York, 1979.

6. Gawthrop, P.J., "A continuous-time approach to discrete-time self-tuning control," Optimal Control: Applications & Methods, vol. 3, no. 4, pp. 399-414, 1982.

7. Allidina, A.Y. and Hughes, F.M., "Generalised self-tuning controler with pole assignment," Proc. IEE, vol. 127 Pt.D, pp. 13-18, 1980.

8. Grimble, M.J., "Implicit and explicit LQG self-tuning regulators," Automatica, vol. 20, pp. 661-670, 1984.

9. Gawthrop, P.J., "Parametric identification of transient signals," IMA Journal of Mathematical Control and Information, vol. 1, pp. 117-128, 1984.

10. Gawthrop, P.J., "Parameter identification from non-contiguous data," Proceedings IEE, vol. 131 pt. D, no. 6, pp. 261-265, 1984.

11. Clarke, D.W., Hodgson, A.J.F., and Tuffs, P.S., "Offset problem and k-incremental predictors in self-tuning control," Proceedings IEE, vol. 130, Pt D., no. 5, pp. 217-225, 1983.

12. Gawthrop, P.J., "Self-tuning PID controllers: Algorithms and implementation," IEEE Transactions on Automatic Control., vol. AC-31, no. 3, 1986.

13. Astrom, K.J. and Wittenmark, B., "On self-tuning regulators," Automatica, vol. 9, pp. 185-199, 1973.

14. Elmqvist, H., "SIMNON: An interactive simulation program for nonlinear systems," Proceedings of Simulation 77, Montreux, 1977.

15. Astrom, K.J. and Wittenmark, B., Computer controlled systems, Prentice-Hall, 1984.

CHAPTER 7

Robustness of Self-Tuning Controllers

Aims. To analyse the behaviour of continuous-time self-tuning controllers in the presence of neglected system dynamics. To introduce the concept of an error feedback system and its role in robustness analysis. To introduce the M-locus approach to analysis and design of robust self-tuning controllers. To illustrate the results using simulation.

7.1. INTRODUCTION

The robustness of non-adaptive emulator based control systems was considered in chapter 4. The purpose of this chapter is to extend those results to include implicit off-line design self-tuning algorithms; that is, the non-adaptive emulators are replaced by self-tuning emulators. The problem is analysed with the realisability filter $\Lambda(s)$ included, but the the results are only complete for the case $\Lambda(s) = 1$. This chapter is based on an internal report[1].

There is a considerable amount of literature concerned with the stability of adaptive controllers. A common thread running through much of this work is the idea of an <u>error feedback system</u>[2]. This error feedback system is a single-loop feedback system composed of two blocks: one a linear transfer function, the other a time varying system representing the effect of the estimator. Although not specifically about adaptive control, many textbooks have been written about the stability of such feedback systems, including[3,4,5,6]. This body of literature provides a valuable source of mathematical tools applicable to the adaptive robustness problem. In particular, Landau[2] applied the <u>hyperstability</u> techniques of Popov[3] to solve a number of adaptive control and estimation problems.

More recently, attention has focused on the input-output approach (as opposed to the state-space Liapunov and Hyperstability approaches). Early work is reported in[7,8,9,10]. Some methods are compared in a discrete-time context in[11]. More recent work appears in[12,13,14,15,16]. An advantage of the input-output approach is that standard textbook[4,5,6] proofs are available for use.

A simpler problem than that considered here arises from the analysis of adaptive algorithms where, unlike in this chapter, neglected dynamics are excluded ($N(s)=1$). Important results (in the discrete-time context) were obtained by Goodwin, Ramadge and Caines[17]. A compendium of results in this area appears in the book by Goodwin and Sin[18].

This chapter provides an analysis of <u>implicit off-line design</u> self-tuning controllers. Complete robust stability results are given when the <u>realisability filter</u> $\Lambda(s)=1$ and partial results when $\Lambda(s) \neq 1$.

7.2. THE ERROR FEEDBACK SYSTEM

In the same vein as chapter 4, an error feedback system describing the evolution of various errors associated with the self-tuning controller can be derived. This has two advantages: an intuitive idea as to what factors are important in determining stability is given; and, in some circumstances, precise robustness criteria may be derived.

The emulation error

The self-tuning emulator gives an output $\hat{\phi}(s)$ which is an approximation to the emulated value $\bar{\phi}(s)$. Define the corresponding emulation error $e^e(t)$ by

$$\bar{e}^e(s) \triangleq \bar{\phi}(s) - \hat{\phi}(s) \tag{1}$$

As in chapter 4, this can be divided into a number of terms which can be written (in terms of Laplace transforms) as

$$\bar{e}^e(s) = [\bar{\phi}^a(s)-\hat{\phi}(s)] + [\bar{\phi}^\star(s)-\bar{\phi}^a(s)] + [\bar{\phi}(s)-\bar{\phi}^\star(s)] \tag{2}$$

$$= \bar{e}^t(s) + \bar{e}^a(s) + \bar{e}^\star(s)$$

where the approximation error $\bar{e}^a(s) = \bar{\phi}^\star(s) - \bar{\phi}^\star(s)$ has been introduced in chapter 4 and the error $\bar{e}^\star(s)$ in chapter 2. The new term due to the tuning $\bar{e}^t(s)$ will be called the tuning error and is given by

$$e^t(t) = \phi^a(t) - \hat{\phi}(t) = \underline{X}^T(t)\underline{\tilde{\theta}}(t) \tag{3}$$

where the error in the parameters $\underline{\tilde{\theta}}(t)$ is given by

$$\underline{\tilde{\theta}}(t) \triangleq \underline{\theta} - \underline{\hat{\theta}}(t) \tag{4}$$

If initial conditions are included in the estimation and design, then equation 2 is replaced by

$$\bar{e}^{e}(s) = [\bar{\phi}^{a}(s)-\hat{\phi}(s)] + [\bar{\phi}^{**}(s)-\bar{\phi}^{a}(s)] + [\bar{\phi}(s)-\bar{\phi}^{**}(s)] \tag{5}$$

$$= \bar{e}^{t}(s) + \bar{e}^{a}(s) + \bar{e}^{**}(s)$$

The approximation error

Following the same analysis as in chapter 4 (section 4.6 in particular), and noting the effect of the additional error term due to tuning $\bar{e}^{t}(s)$, it follows that

$$\bar{e}^{a}(s) = - M(s)[\bar{z}(s) + \bar{e}^{a}(s) + \bar{e}^{t}(s)] \tag{6}$$

$$= - M(s)[\tilde{z}(s) + \bar{e}^{e}(s)]$$

where, as in chapter 3, equation 3.3.11,

$$\tilde{z}(s) = R(s)\bar{w}(s) - e^{sT}\frac{P(s)C(s)}{Z(s)A(s)}\bar{v}(s) \tag{7}$$

The estimation error

The emulation error $\bar{e}^{e}(s)$ is closely related to the estimation error, which was defined in chapter 5 as

$$\hat{e}(s) \triangleq \bar{\Psi}(s) - \hat{\Psi}(s) \tag{8}$$

where $\bar{\Psi}(s)$ is the scalar output of the linear-in-the-parameters model and $\hat{\Psi}(s)$ its estimate. In the particular case of implicit off-line design algorithms, $\hat{e}(s)$ is given by

$$\hat{e}(s) = \bar{\phi}_{\Lambda}(s) - \hat{\phi}_{\Lambda}(s) = \Lambda(s)\bar{\phi}(s) - \hat{\phi}_{\Lambda}(s) \tag{9}$$

where $\Lambda(s)$ is the <u>realisability filter</u>. At first glance, $\hat{e}(s)$ appears to be just a $\Lambda(s)$ filtered version of $\bar{e}^{e}(s)$; but this is not so, as $\hat{\phi}_{\Lambda}(s) \neq \Lambda(s)\hat{\phi}(s)$ (unless $\Lambda(s) = 1$ or $\hat{\underline{\theta}}(t)$ is constant). So we define the <u>filter-induced error</u> $\tilde{e}(s)$ by

$$\tilde{e}(s) \triangleq \Lambda(s)\hat{\phi}(s) - \hat{\phi}_{\Lambda}(s) \tag{10}$$

This error is zero in two cases:

1. $\Lambda(s) = 1$

2. $\hat{\underline{\theta}}(t)$ is constant

Combining these equations gives

$$\hat{e}(s) = \Lambda(s)\bar{\phi}(s) - \hat{\phi}_{\Lambda}(s) \tag{11}$$

$$= \Lambda(s)(\bar{\phi}(s) - \hat{\phi}(s)) - (\Lambda(s)\hat{\phi}(s) - \hat{\phi}_{\Lambda}(s))$$

$$= \Lambda(s)[\bar{\phi}(s) - \hat{\phi}(s)] - \tilde{e}(s)$$

$$= \Lambda(s)\bar{e}^{e}(s) - \tilde{e}(s)$$

Rearranging the last equation gives the emulation error $\bar{e}^{e}(s)$ in terms of the estimation error $\hat{e}(s)$ as:

$$\bar{e}^{e}(s) = \Lambda(s)^{-1}[\hat{e}(s) + \tilde{e}(s)] \tag{12}$$

$$\bar{e}^{e}(s) = \Lambda(s)^{-1}[\hat{e}(s) + \tilde{e}(s)] \tag{12}$$

Example

Suppose that $\Lambda(s) = e^{-sT}$. Then

$$\tilde{e}(t) = \underline{X}^{T}(t)[\hat{\underline{\theta}}(t) - \hat{\underline{\theta}}(t-T)] \tag{13}$$

and

$$e^{e}(t) = \hat{e}(t+T) + \underline{X}^{T}(t)[\hat{\underline{\theta}}(t+T) - \hat{\underline{\theta}}(t)] \tag{14}$$

The filter induced error $\tilde{e}(s)$ is zero if either T=0 or $\hat{\underline{\theta}}(t)$ is constant.

The filter induced error $\tilde{e}(t)$ is then closely related to the difference between the a-priori and a-posteriori errors discussed in a discrete-time context by Landau[2] and others.

□

The estimator input

In chapter 5, it was shown that the least-squares parameter estimator could be viewed as a single-input single-output system Ω with input e(t) and output $\hat{e}(t)$. In particular, the estimator input e(t) is given by (5.5.3):

$$\bar{e}(s) = \Lambda(s)\bar{\phi}(s) - \bar{\underline{X}}^{T}(s)\underline{\theta} \tag{15}$$

$$= \Lambda(s)(\bar{\phi}(s) - \bar{\phi}^{a}(s)) = \Lambda(s)(\bar{e}^{\star}(s) + \bar{e}^{a}(s))$$

Using equation 6 to replace the approximation error $\bar{e}^{a}(s)$,

$$\bar{e}(s) = \Lambda(s)\bar{e}^{\star}(s) \tag{16}$$

$$- \Lambda(s)M(s)[\tilde{z}(s) + \bar{e}^{e}(s)]$$

And using equation 12 to replace the emulation error $\bar{e}^{e}(s)$,

$$\bar{e}(s) = \Lambda(s)\bar{e}^{\star}(s) \tag{17}$$

$$- \Lambda(s)M(s)[\tilde{z}(s) + \Lambda(s)^{-1}(\hat{e}(s) + \tilde{e}(s))]$$

$$= \Lambda(s)[\bar{e}^{\star}(s) - M(s)\tilde{z}(s)] - M(s)\tilde{e}(s) - M(s)\hat{e}(s) \tag{18}$$

Writing the <u>disturbance and setpoint induced error</u> $\bar{e}^{d}(s)$ as

$$\bar{e}^{d}(s) \triangleq \Lambda(s)[\bar{e}^{\star}(s) - M(s)\tilde{z}(s)] \tag{19}$$

the estimator input error $\bar{e}(s)$ is seen to contain three components, the disturbance and setpoint induced error $\bar{e}^{d}(s)$, the filter induced error $\tilde{e}(s)$ filtered by $-M(s)$, and the estimator output error filtered by the transfer function $-M(s)$. That is,

$$\bar{e}(s) = \bar{e}^{d}(s) - M(s)\tilde{e}(s) - M(s)\hat{e}(s) \tag{20}$$

<u>The error feedback system</u>

Equation 20 gives the estimator input $\bar{e}(s)$ in terms of the estimation error $\hat{e}(s)$, the filter-induced error $\tilde{e}(s)$ and the disturbances induced error $\bar{e}^{d}(s)$. Combining this linear system with the time-varying estimator system Ω relating $\hat{e}(t)$ and $e(t)$ gives the <u>error feedback system</u>

displayed in Figure 7.2.1.

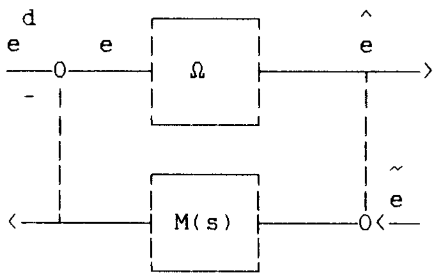


Figure 7.2.1 The error feedback system

The output error

As well as being of interest in its own right, the effect of the emulation error $e^{e}(t)$ on the system output is of interest. This effect can be studied on the basis of the notional feedback system considered in chapters 3 and 4. One difference here is that the difference between the emulator output and the emulated signal is now $\bar{e}^{e}(s) = -\bar{e}^{\star}(s) + \bar{e}^{a}(s) + \bar{e}^{t}(s)$ rather than $-\bar{e}^{\star}(s)$ in chapter 3 and $-\bar{e}^{\star}(s) + \bar{e}^{a}(s)$ in chapter 4. Another difference is that the neglected dynamics N(s) now appear explicitly. The corresponding block diagram appears in Figure 7.2.2. Define $e^{y}(t)$ to be the component of the system output due to the emulation error

$$\bar{e}^{y}(s) = \frac{L(s)}{1+L(s)}\, e^{-sT}\, \frac{Z(s)}{P(s)}\, \bar{e}^{e}(s) \tag{21}$$

Using equation 12 this then gives the output error $\bar{e}^{y}(s)$ in

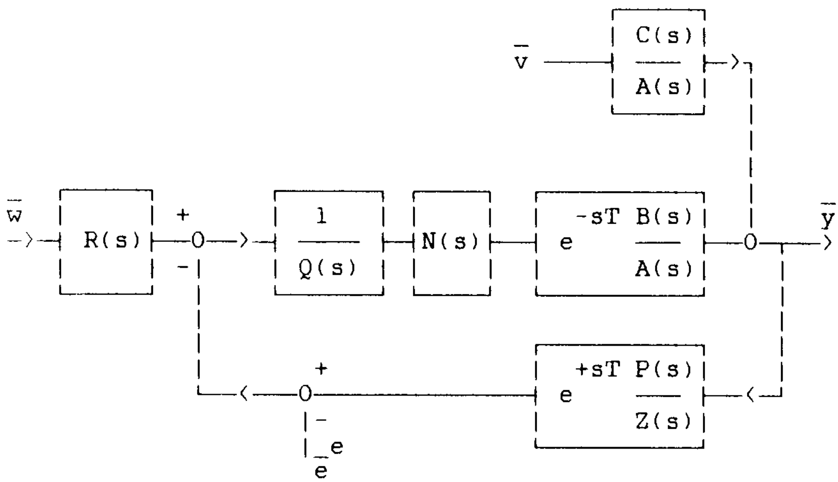


Figure 7.2.2 The notional feedback system

terms of the estimation error $\hat{e}(s)$ and the filter-induced error $\tilde{e}(s)$ as

$$\bar{e}^{y}(s) = \frac{L(s)}{1+L(s)} e^{-sT} \frac{Z(s)}{P(s)} \Lambda(s)^{-1}[\hat{e}(s) + \tilde{e}(s)] \tag{22}$$

Exponential weighting

As in chapter 5, exponentially weighted signals are useful in deriving stability results. In chapter 5, it was shown that $\hat{e}(t)$ and $e(t)$ could be replaced by exponentially weighted versions:

$$\hat{e}_{\alpha}(t) = e^{\alpha t}\, \hat{e}(t);\quad e_{\alpha}(t) = e^{\alpha t}\, e(t) \tag{23}$$

and Ω still has a gain of one as long as $\alpha \leq \beta/2$.

Moreover, pre- and post- exponentially multiplying the linear transfer function M(s) gives $M(s - \alpha)$. The gain of this transfer function is called γ_α and, if $M(s - \alpha)$ is stable, is given by

$$\gamma_\alpha = \sup_\omega \left| M(j\omega - \alpha) \right| \tag{24}$$

This is considered further in the next section.

7.3. THE M-LOCUS

The error feedback system (Figure 7.2.1) for the adaptive case is similar to that in chapter 4 for the non-adaptive case. In particular, the transfer function M(s)

$$M(s) = \frac{Z^+(s)E(s)A(s)}{P(s)C(s)} \frac{N^{-1}(s)-1}{1+L^{-1}(s)N^{-1}(s)} \tag{1}$$

still appears in the feedback loop. The differences are:

1. The unit feedback loop appearing in chapter 4 is replaced by the system Ω, which has a gain of one.

2. The filter induced error $\tilde{e}(s)$ appears as a disturbance.

Not surprisingly, the transfer function M(s) is crucial in analysing the stability of the feedback system. Roughly speaking(details will appear in the next section), a standard Theorem applicable to this sort of feedback loop[4,5,6] says that the feedback loop will be stable if the loop-gain is less than one. As we have already decided that the gain of Ω with exponential weighting is less than one when making

Assumption 1

the exponential weighting coefficient α and the exponential forgetting factor β (section 5.3) are related by

$$\alpha = \frac{1}{2}\beta \ , \tag{2}$$

we get the rather simple result that stability of the feedback loop follows from the gain of $M(s - \alpha)$ being less than one.

There are two parts to this condition:

1. $M(s)$ must be stable. As $P(s)$ and $C(s)$ are chosen to be stable, this condition becomes that the transfer function:

$$\frac{N^{-1}(s)-1}{1+L^{-1}(s)N^{-1}(s)} = \frac{L(s)[1-N(s)]}{1+L(s)N(s)} \tag{3}$$

be stable. This condition is satisfied if two assumptions are true:

Assumption 2

$N(s - \alpha)$ is stable.

Assumption 3

$\frac{L(s - \alpha)}{1+L(s - \alpha)N(s - \alpha)}$ is stable.

2. The gain of $M(s - \alpha)$ is less than 1. This can be written as:

Assumption 4

$$\gamma_\alpha = \sup_\omega \left| M(j\omega - \alpha) \right| \tag{4}$$

Note that assumptions 2 and 4 depend on the choice of $N(s)$ in the decomposition of equation 4.2.3, repeated here as

$$H(s) = e^{-sT}\frac{B(s)}{A(s)}N(s) \tag{5}$$

It is important to realise that, in the adaptive context, the <u>nominal</u> <u>system</u> $\frac{B(s)}{A(s)}$ and the resultant neglected dynamics N(s) are <u>not</u> chosen. Thus all that is required is that such a choice exists satisfying the above criteria.

Finally, to deduce that the signals are bounded, we must also assume that the exogenous signals due to the set-point and disturbance are bounded:

<u>Assumption 5</u>

$$e^{d_2} < \kappa_0 \tag{6}$$

where κ_0 is a constant.

<u>The importance of control weighting</u>

Typical neglected dynamics are low-pass. That is,

$$\operatorname*{Lt}_{\omega\to\infty} N(j\omega) = 0 \tag{7}$$

Hence, at high frequencies,

$$M(s) \cong \frac{Z^+(s)E(s)A(s)}{P(s)C(s)}L(s) \tag{8}$$

Without control weighing, $L(s) = \infty$ at all frequencies and thus the small gain condition cannot be satisfied. It follows that control weighting is essential when low-pass neglected dynamics are present.

Although nothing has been proved so far in this chapter, it seems at this stage that M(s) is crucial in determining the stability of the self-tuning controller when neglected dynamics are present. As shown in the next section, stability can be shown (in terms of M(s)) for the case when $\tilde{e}(s)=0$, that is $\Lambda(s)=1$. Although not proved, simulations suggest that these results may be extended to include $\Lambda(s)\neq 1$.

7.4. ADAPTIVE ROBUSTNESS

In this section, it is assumed that

$$\Lambda(s)=1, \text{ that is } \tilde{e}(s) = 0 \qquad (1)$$

The error equations developed in the previous sections reveal that the robustness problem reduces to examining the single-loop feedback system of Figure 7.4.1. Note that as $\Lambda(s) = 1$ the filter induced error $\bar{e}_\Lambda(s)$ is zero.

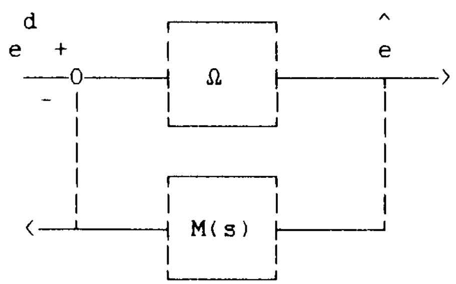


Figure 7.4.1 The exponentially multiplied system

Outline of proof

The proof proceeds as follows:

1. In Lemma 7.1, the exponentially multiplied error

This section involves some technical mathematics. It may be omitted on a first reading.

feedback system is shown to be L_2 stable using the standard small-gain theorem[5] and the formula for the gain γ_α (7.2.24).

2. In Lemma 7.2, it is shown how L_∞ results about the error feedback system can be derived from the L_2 results about the exponentially multiplied error feedback system.

3. Theorem 7.1 combines the two Lemmas to give input-output stability results for the self-tuning controller in terms of the neglected dynamics N(s) and the emulator design polynomials P(s), C(s) and Q(s).

4. Theorem 7.2 (section 7.5) extends these results to include parameter boundedness and estimation error $\hat{e}(t)$ boundedness. This requires a persistent excitation condition to be imposed on the signals affecting the system.

Lemma 7.1 (L_2 stability of the exponentially weighted system)

If assumptions 1-4 of section 7.3 are true, that is $M(s - \alpha)$ is stable, $\alpha = \frac{1}{2}\beta$ and $\gamma_\alpha < 1$. Then the exponentially weighted system of equations displayed in Figure 7.4.1 is L_2 stable in the sense that the estimation error e $\hat{e}(t)$ and the estimator input error e(t) are bounded by

$$\sqrt{\int_0^t e^{2\alpha\tau}\hat{e}^2(\tau)d\tau} < \frac{1}{1-\gamma_\alpha}\sqrt{\int_0^t e^{2\alpha\tau}e_d^2(\tau)d\tau} + \kappa_1 \tag{2}$$

$$\sqrt{\int_0^t e^{2\alpha\tau}e^2(\tau)d\tau} < \frac{1}{1-\gamma_\alpha}\sqrt{\int_0^t e^{2\alpha\tau}e_d^2(\tau)d\tau} + \kappa_2 \tag{3}$$

where κ_1 and κ_2 are finite constants and γ_α is the gain of $M(s - \alpha)$ (see 7.2.24).

Proof

This follows from the small gain theorem III.2.1 on page 41 of[5] and the fact that the gain of $\Omega_\alpha \leq 1$ (see chapter 5). □

Remarks

1. Setting $\alpha = 0$, this theorem gives L_2 stability of the system. This holds even with no forgetting ($\beta = 0$).

2. Using assumptions 2 and 3 and assuming that the disturbance and the setpoint are uniformly bounded, the signal $e^d(t)$ is uniformly bounded.

3. If the quantity $e^d(t)$ is exponentially decreasing faster than $e^{-\alpha t}$, then so is $\hat{e}$.

Lemma 7.2 (Bounds on low-pass filtered signals)

If the error system input $e^d(t)$ is bounded (assumption 4), then the low-pass filtered estimation error:

$$\hat{e}_F(t) \triangleq \sqrt{\int_0^t e^{-2\alpha(t-\tau)}\hat{e}^2(\tau)d\tau} \tag{4}$$

is bounded by:

$$\hat{e}_F(t) < \frac{1}{1-\gamma_\alpha}[\frac{\kappa_0}{\sqrt{(2\alpha)}} + e^{-\alpha t}\kappa_1] \tag{5}$$

Proof: From assumption 4, the integral in the righthand side of equation 2 of Lemma 7.1 is bounded by:

$$\sqrt{\int_0^t e^{2\alpha\tau}e^{d2}(\tau)d\tau} \leq \kappa_0\sqrt{\int_0^t e^{2\alpha\tau}d\tau} \tag{6}$$

$$= \sqrt{(\frac{1}{2\alpha}[e^{2\alpha t}-1]} < \frac{1}{\sqrt{(2\alpha)}}e^{\alpha}t$$

Substituting this into equation 2 of Lemma 7.1 and multiplying by $e^{-\alpha t}$ gives the result.

□

This Lemma gives conditions such that the signal obtained by passing the squared emulator error ($\hat{e}^2$) through the low-pass filter $\frac{1}{s + 2\alpha}$ is bounded. Of course, this does not imply that the emulator error is bounded. A lemma due to Vidyasagar[6] (section 9.1) shows that this result does imply that the output signal obtained by passing the emulator error $\hat{e}$ into any low-pass system whose impulse response decays faster than $e^{-2\alpha t}$ (in particular that generating $\bar{e}^y(s)$) is in L_∞.

This result is used to prove the main robustness theorem of this book.

Theorem 7.1(Adaptive robustness)

If assumptions 1-4 of section 7.3 are satisfied, then the output error $e^y(t)$ is bounded.

Proof

Let m(t) be the inverse Laplace transform (impulse response) of M(s). Then:

$$e^y(t) = \int_0^t m(t - \tau)\hat{e}(\tau)d\tau \tag{7}$$

$$= \int_0^t e^{\alpha(t-\tau)} m(t - \tau) e^{-\alpha(t-\tau)} \hat{e}(\tau)d\tau$$

Using Schwartz's inequality:

$$e^{y2}(t) \leq \int_0^t e^{2\alpha(t-\tau)} m^2(t - \tau)d\tau . \int_0^t e^{-2\alpha(t-\tau)} \hat{e}^2(\tau)d\tau \tag{8}$$

Using assumptions 2 and 3 of section 7.3, it follows that:

$$\int_0^t e^{2\alpha(t-\tau)} m^2(t - \tau)d\tau < K \qquad (9)$$

where K is a constant. The result then follows from Lemmas 7.1 and 7.2, and assumptions 1-4.

□

Remarks.

1. The adaptive and non-adaptive results are both based on the Nyquist locus of M(s). In the adaptive case, the locus must lie within the unit circle, and in the non-adaptive case, must not encircle the -1 point.

2. In the adaptive case, it is required only that there exist a nominal system $\frac{B}{A}$ such that the condition on M(s) is satisfied. If the orders of B and A correspond to those of the numerator and denominator of the actual system G(s), then such a system always exists, namely $\frac{B}{A}$ = G(s) which gives N(s) = 1 and thus M(s) = 0.

   In the non-adaptive case, the condition on M(s) must be satisfied for the particular nominal system chosen by the designer. Even if the orders of B and A are correct, parametric error can give a non-zero M(s) for the chosen system.

3. This result may be related to that of Kosut, Johnson and Friedlander[12,13] by

$$M(j\omega - \alpha) < 1 \Leftrightarrow Re\{H_{ev}(j\omega - \alpha)\} > \frac{1}{2}$$

   where

$$H_{ev} \triangleq [1+M(s)]^{-1}$$

4. The results differ from those of Kosut, Johnson and Friedlander[12,13] in that we consider an algorithm with control weighting which makes it possible to satisfy assumptions 3 and 4 of section 7.3.

7.5. INTERNAL STABILITY

Section 6 deals entirely with input-output stability; it does not directly give information about the properties of the parameter error $\underline{\tilde{\theta}}$ or about the data vector $\underline{X}$. This section considers this problem, again for the special case of $\Lambda(s)=1$, that is $\phi(t)$ is realisable.

This section shows that both the data vector $\underline{X}$ and the parameter error $\underline{\tilde{\theta}}$ are bounded. Not surprisingly, the latter result requires a persistent excitation condition on the data vector $\hat{\underline{X}}$.

The properties of the data vector $\underline{X}$ are treated in the following Lemma:

Lemma 7.3 (Boundedness of the data vector $\underline{X}$)

Under the same conditions as Theorem 7.1 , all elements of the data vector $\underline{X}$ (equation 6.5.12) are uniformly bounded.

Proof

From Theorem 7.1, the system output y is uniformly bounded.

The control signal is obtained from

$$\hat{u}(s) = \frac{1}{Q(s)}[R(s)\bar{w}(s) - \hat{\phi}(s)] \qquad (1)$$

This section involves some technical mathematics. It may be omitted on a first reading.

$$= \frac{1}{Q(s)}[R(s)\bar{w}(s) - \frac{P(s)}{Z(s)}\bar{y}(s) - \hat{e}(s)]$$

$1/Q(s)$ and $P(s)/Z(s)$ are proper in the case considered here. The corresponding components of the data vector $\underline{X}$ are obtained by filtering $\hat{u}(s)$ by the low-pass filter $1/Z(s)$. This filtered signal has three components driven by $\bar{w}(s)$, $\bar{y}(s)$ and $\hat{e}(s)$. $w(t)$ is, by assumption, bounded. $\bar{y}(s)$ has been shown to be bounded. The component due to $\hat{e}(s)$ is also bounded,as we have shown that $\hat{e}(s)$ is bounded when passed though a low-pass filter.

The elements of the $\underline{X}$ vector are obtained by passing $\bar{y}(s)$ or $\bar{u}(s)/Z(s)$ through <u>proper</u> transfer functions of the form $s^i/C(s)$; so these elements are also uniformly bounded.

□

The boundedness result for the parameter error $\tilde{\underline{\theta}}$ is contained in the following Theorem:

<u>Theorem 7.2</u> (<u>Bounded parameter error</u>)

If, in addition to the conditions of Theorem 7.1, the data vector $\hat{\underline{X}}$ is persistently exciting in the sense that

<u>Assumption 6</u>

$$S(t) = \int_0^t e^{-\beta(t-\tau)}\hat{\underline{X}}(\tau)\hat{\underline{X}}^T(\tau)d\tau > \Sigma \qquad (2)$$

where Σ is a positive definite matrix, then the parameter error $\tilde{\underline{\theta}}$ is uniformly bounded.

Note that $S(t)$ is the output of the low-pass filter used in the parameter estimator (equation 5.5.2).

Proof

Equation 5.6.21 can be rearranged as

$$V(t,\alpha) = V(0,\alpha) + \int_0^t e^{2\alpha\tau} e^2(\tau)d\tau - \int_0^t e^{2\alpha\tau} \hat{e}^2(\tau)d\tau \tag{3}$$

Multiplying each side of the equation by $e^{-2\alpha t}$ gives:

$$\underline{\tilde{\theta}}(t)^T S \underline{\tilde{\theta}}(t) = e^{-2\alpha t} V(0,\alpha) + e_F^2(t) - \hat{e}_F^2(t) \tag{4}$$

where $\hat{e}_F(t)$ and $e_F(t)$ are the filtered error signals defined as in equation 7.4.5 (Lemma 7.2). Lemma 7.2 then shows that the right-hand side of equation 4 is bounded and so:

$$\underline{\tilde{\theta}}(t)^T S \underline{\tilde{\theta}}(t) < \kappa_3 \tag{5}$$

where κ_3 is a constant. The result follows from assumption 6.

□

7.6. ROHRS EXAMPLE

In a celebrated paper[19], Rohrs and his colleagues illustrated the poor robustness properties of a particular model-reference adaptive control algorithm by examining its performance on two particular example systems. In chapter 4, the non-adaptive robustness properties were examined; in this section, the second of these example systems is used to illustrate the robustness results for the detuned model-reference adaptive controller analysed in the previous section together with some related controllers. Simulations appear in the next section.

The system and the design parameters

These have already been considered in the example considered in section 4.7.

Robustness analysis

As discussed in section 7.3, the basic requirement (assumption 2) is that the exponentially multiplied notional feedback loop (with neglected dynamics) should be stable.

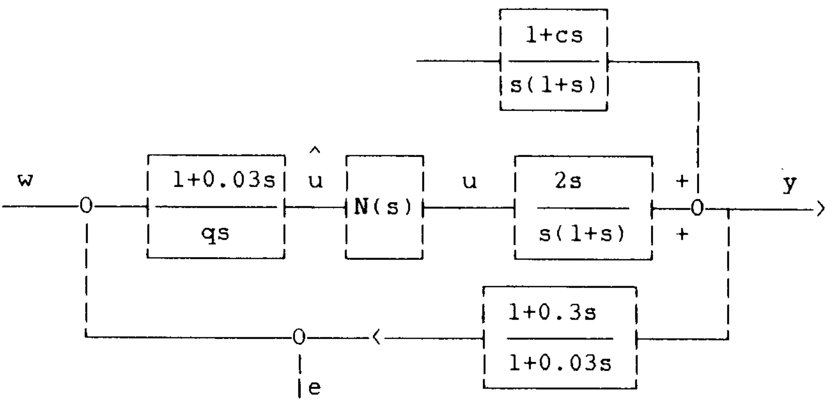


Figure 7.6.1 The notional feedback system

From Figure 7.6.1, the notional loop gain L(s) is

$$L(s) = \frac{2(1+0.3s)}{qs(1+s)} \qquad (1)$$

We can get a rough estimate of the value of q required for stability as follows. At high frequencies:

$$L(j\omega) \cong \frac{0.6}{j\omega q} \qquad (2)$$

and in particular the argument of L is about $-\pi/2$ radians. At a frequency of 10 radians $\sec^{-1}$, the argument of $N(j\omega)$ is also $-\pi/2$ radians and its gain is 100/80. Thus for the $L(j\omega)N(j\omega)$ locus to pass though the -1 point:

$$\frac{0.6}{10q} \cdot \frac{100}{80} = 1 \tag{3}$$

that is

$$q \cong \frac{6}{80} \cong 0.1 \tag{4}$$

To exemplify the use of the various criteria presented in this chapter, we will consider four examples based on that of Rohrs. These four examples are identical to those considered in chapter 4 except that we now consider adaptive control.

The four examples have the following in common:

1. Four frequency loci are plotted for values of $\omega > 0$ in Figures 4.7.1-4:

   a) The actual loop gain: $L_a(j\omega)$ (equation 4.3.5).

   b) The notional loop gain (with neglected dynamics included) $N(j\omega)L(j\omega)$.

   c) The M-locus $M(j\omega)$ (equation 4.6.3).

   d) The M'-locus $M'(j\omega)$ (equation 4.5.4).

2. The actual system $H(s)$ is as given in equation 4.7.1.

3. The emulator and controller design parameters are as given in equation 4.7.4-9.

The four examples are different in the following ways: The parameter b determining the decomposition of equation 4.7.2, and the control weighting factor q of equation 4.7.9, are varied as in the following Table:

Example	b	q
1	1.0	0.05
2	1.0	0.2
3	0.5	0.05
4	0.5	0.2

Remarks

1. The loci for La and $M'(s)$ are not relevant to adaptive control.

2. In each case, $N(s)$ is stable and so assumption 2 of section 7.3 is satisfied for sufficiently small α.

3. In examples 1 and 3, the $N(s)L(s)$ locus encircles the -1 point indicating instability; in examples 2 and 4 it does not, indicating stability. Thus examples 2 and 4 satisfy assumption 3 of section 7.3 for sufficiently small α; examples 1 and 3 do not.

4. In examples 2 and 4, the $M(j\omega)$ locus has magnitude less than one at all frequencies. Thus assumption 4 of section 7.3 is satisfied for sufficiently small values of α.

5. The $L(j\omega)N(j\omega)$ locus does not depend on b. Thus it is the same for examples 1 and 3 and for 2 and 4.

6. In the adaptive context, all that is required is that a suitable nominal system $\frac{B(s)}{A(s)}$, together with $N(s)$, exist satisfying 4.2.3:

$$H(s) = e^{-sT}\frac{B(s)}{A(s)}N(s) \tag{5}$$

Thus in this context it is merely required that the criteria be satisfied for some value of b. In fact, the criteria are satisfied for both of examples 2 and 4.

To summarise, if q=0.2, the adaptive controller is stable, but if q=0.05 it has not been shown to be stable and may be unstable.

7.7. SIMULATION RESULTS

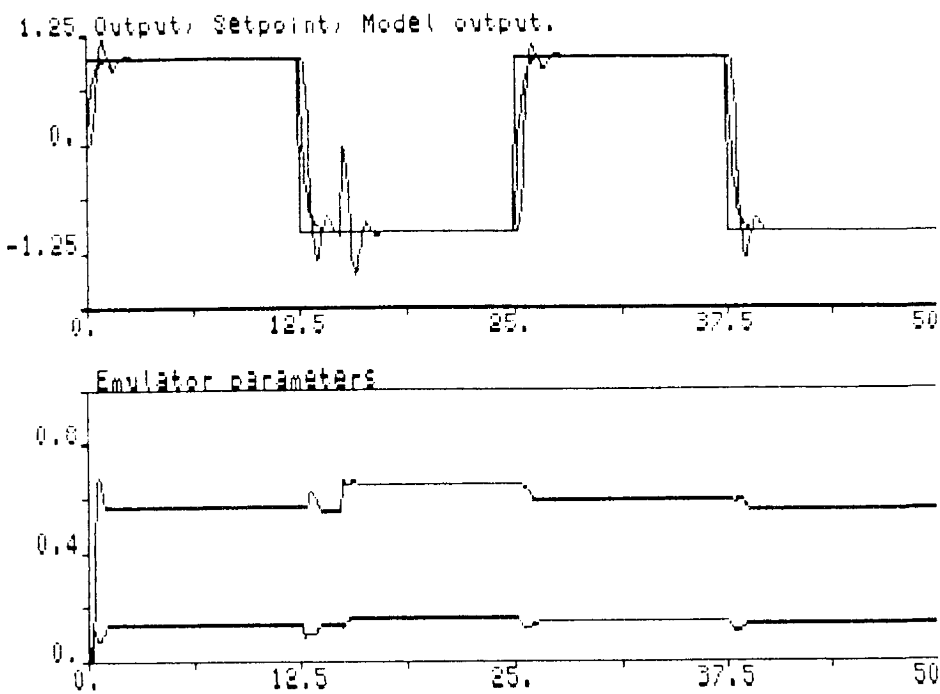


Figure 7.7.1 Example 1

The simulation results of this section illustrate the results of this chapter and indicate that the results also seem to apply to a wider class of self-tuners than actually analysed. To enable comparisons to be made to the results of other workers, the example of Rohrs[19] discussed in the previous section is considered.

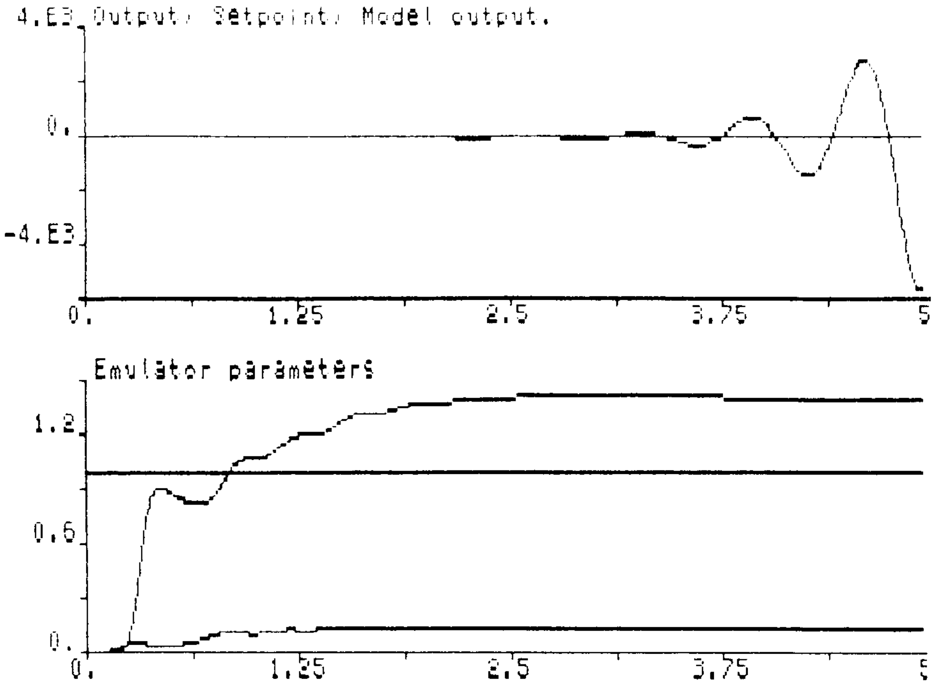


Figure 7.7.2 Example 2

As in chapter 6, the self-tuning algorithms were simulated using the SIMNON language[20,21] (Figures 7.7.1-6). All examples have the following in common:

1. Two emulator parameters are identified.

2. The initial $\underline{S}^{-1}(t)$ matrix is, in each case, given by:

$$S^{-1}(0) = \begin{vmatrix} 100 & 0 \\ 0 & 100 \end{vmatrix} \qquad (1)$$

3. The emulator design parameters are chosen according to the various strategies.

4. All examples are <u>detuned</u> versions of the underlying algorithm. Q(s) is given in Tables 7.1 and 7.2 (pages

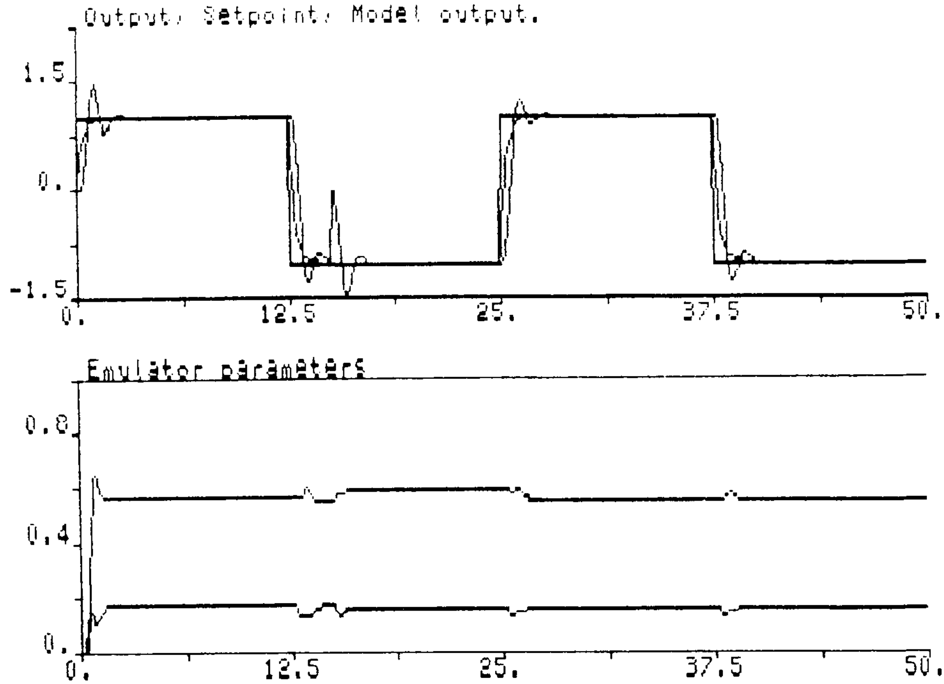


Figure 7.7.3 Example 3

7-29&30).

5. The algorithms are simulated using a system having the neglected dynamics

$$N(s) = \frac{100}{s^2 + 8s + 100} \tag{2}$$

dynamics for 50 time units. All examples have a unit output step disturbance occurring at time=15 units; that is, one is added to the system output from time 15 onwards.

6. The upper graph of Figures 7.7.1-6 shows the setpoint (a square wave between +1 and -1 with a period of 25 units), the actual system output, and the model output.

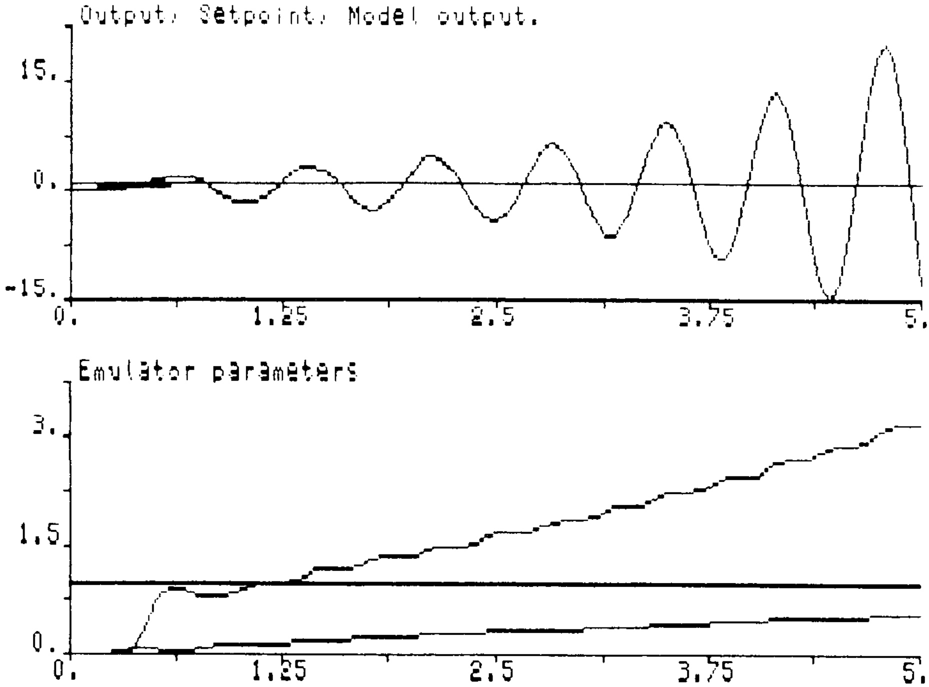


Figure 7.7.4 Example 4

The model output corresponds to:

$$\bar{y}_m(s) = \frac{Z(s)}{P(s)}\bar{w}(s) \qquad (3)$$

7. The lower graph of Figures 7.7.1-6 shows the evolution of the two emulator parameters with respect to time.

The differences between the six examples are summarised in Tables 7.1 and 7.2 (pages 7-29&30). In Table 7.1, MR means model reference and PP pole placement.

Remarks

1. Despite the diversity of algorithms treated here, they

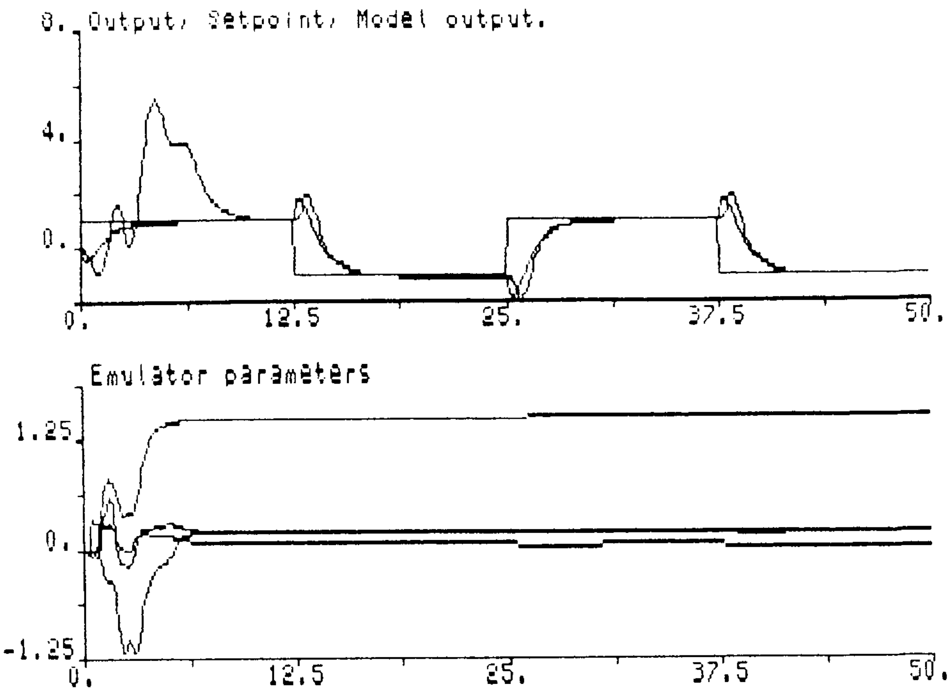


Figure 7.7.5 Example 5

all have a common notional loop-gain:

$$L(s) = \frac{P(s)B(s)}{Z(s)Q(s)A(s)} = \frac{2(1+0.3s)}{qs(1+s)} \tag{4}$$

Thus the L(s)N(s) locus of Figure 4.7.1 (for q=0.05) is appropriate to examples 2, 4 and 6; and Figure 4.7.2 (for q=0.2) is appropriate to examples 1, 3 and 5.

2. Examples 1 and 2 are as discussed in the previous section. The self-tuning controller of example 1 is stable as predicted; that of example 2 was not predicted to be stable and is, in fact, unstable. Simulations starting off at the correct (that is, based on the correct nominal system) parameters and with a reduced initial variance did, however, give stability in both examples 1

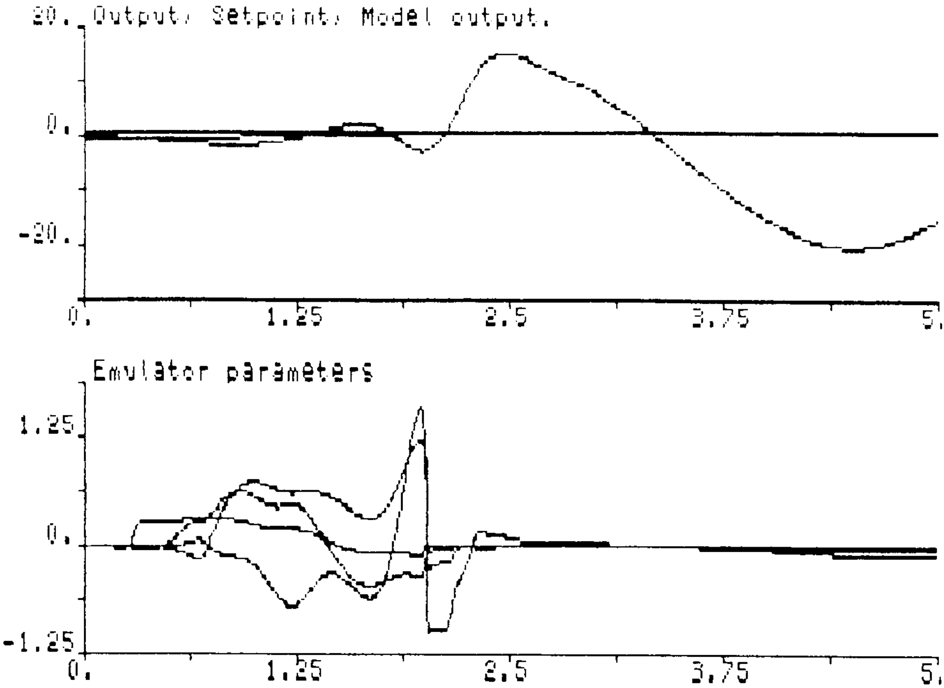


Figure 7.7.6 Example 6

Table 7.1 SIMULATION SUMMARY

No.	Method	A(s)	B(s)	P(s)	Z(s)
1	MR	s(1+s)	2s	1+0.3s	1+0.03s
2	MR	s(1+s)	2s	1+0.3s	1+0.03s
3	MR	s(1+s)	2s	1+0.3s	1
4	MR	s(1+s)	2s	1+0.3s	1
5	PP	$(1+s)^2$	2(1-s)	(1+0.3s)(1+s)	0.5B(s)
6	PP	$(1+s)^2$	2(1-s)	(1+0.3s)(1+s)	0.5B(s)

and 2.

3. Examples 3-6 were not analysed in the previous section. But, as pointed out in remark 1, the L(s)N(s) loci are

Table 7.2 SIMULATION SUMMARY

No.	Q(s)	Λ(s)	Design
1	0.2/(1+0.03)	1	Off-line
2	0.05/(1+0.03)	1	Off-line
3	0.2	Z(s)/P(s)	Off-line
4	0.05	Z(s)/P(s)	Off-line
5	0.2	Z(s)/P(s)	On-line
6	0.05	Z(s)/P(s)	On-line

appropriate. Thus M(s) is stable in examples 3 and 5 and unstable in examples 4 and 6. It was suggested, but not proved, that stability of M(s) was essential for global stability of all the algorithms treated here. As shown in the appropriate Figures, this tentative prediction is realised; the self-tuning controller in examples 3 and 5 is stable but unstable in examples 4 and 6.

4. The importance of the control weighting Q(s) was emphasised in section 7.3. In these simulations, Q(0)=0 in each case giving no low-frequency weighting. The weighting in examples 1, 3 and 5 is four times that in examples 2,4 and 6; as predicted, the robustness of the algorithms is improved by the control weighting.

References

1. Gawthrop, P.J., Robust continuous-time self-tuning control of single-input single-output systems, University of Sussex, School of Engineering and Applied Sciences, 1986.

2. Landau, I.D., Adaptive control: The model reference approach, Marcel Dekker, New York, 1979.

3. Popov, V.M., Hyperstability of control systems, Springer-Verlag, 1973.

4. Willems, J.C., The analysis of feedback systems, MIT Press, 1971.

5. Desoer, C.A. and Vidyasagar, M., Feedback systems : Input-output properties, Academic Press, London, 1975.

6. Vidyasagar, M., Input-output analysis of large-scale interconnected systems, Springer, Berlin, 1981.

7. Egardt, B., Stability of adaptive controllers, Springer, 1979.

8. Egardt, B., "Stability analysis of continuous-time adaptive control systems," SIAM J. Control & Optimiz., vol. 18, p. 540, 1980.

9. Gawthrop, P.J., "On the stability and convergence of a self-tuning controller," Int. J. Control, vol. 31, no. 5, pp. 973-998, 1980.

10. Gawthrop, P.J. and Lim, K.W., "On the robustness of self-tuning controllers," Proc. IEE, vol. 129 ptD, pp. 21-29, 1982.

11. Gawthrop, P.J., "Some properties of discrete adaptive controllers," in Self-tuning and adaptive control -

theory and applications, ed. Harris and Billings, Peter Peregrinus, 1981.

12. Kosut, R.L. and Johnson, C.R., "An input-output view of robustness in adaptive control," Automatica, vol. 20, no. 5, pp. 569-582, 1984.

13. Kosut, R. and Friedlander, B., "Robust adaptive control: Conditions for global stability," IEEE Transactions on Automatic Control, vol. AC-30, no. 7, pp. 610-624, 1985.

14. Ortega, R., Praly, L., and Landau, I.D., "Robustness of discrete-time direct adaptive controllers," IEEE Transactions on Automatic Control, vol. AC-30, no. 12, pp. 1179-1187, 1985.

15. Gawthrop, P.J., "Robustness of self-tuning controllers. PartI: Single-input single-output systems.," Report CE/T/13, School of Engineering and Applied Sciences, Univ. of Sussex., 1985.

16. Gawthrop, P.J., "Robustness of self-tuning controllers PartII: Two-input two-output systems.," Report CE/T/12, School of Engineering and Applied Sciences, Univ. of Sussex., 1985.

17. Goodwin, G.C., Ramadge, P.J., and Caines, P.E., "Discrete-time multivariable adaptive control," IEEE Trans., vol. AC-25, pp. 449-456, 1980.

18. Goodwin, G.C. and Sin, K.S., Adaptive filtering prediction and control, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1984.

19. Rohrs, C.E., Valavani, L., Athans, M., and Stein, G., "Robustness of continuous-time adaptive control in the presence of unmodeled dynamics," Trans. IEEE, vol. AC-30, pp. 881-889, 1985.

20. Elmqvist, H., "SIMNON: An interactive simulation program for nonlinear systems," Proceedings of Simulation 77, Montreux, 1977.

21. Astrom, K.J. and Wittenmark, B., Computer controlled systems, Prentice-Hall, 1984.

CHAPTER 8

Non-Adaptive and Adaptive Robustness

Aims. To compare and contrast adaptive and non-adaptive approaches to sensitivity reduction by feedback. To suggest a three degree of freedom approach to the design of self-tuning controllers.

8.1. INTRODUCTION

It is now over 20 years since Horowitz[1,2] discussed the relationship between adaptive and non-adaptive feedback systems used for removing the effects of plant uncertainty.

(Some readers may prefer the terms "passive-adaptive" or "ordinary feedback" to the term "non-adaptive" and the terms "active-adaptive", "plant adaptive" or "parameter-adaptive" to the term "adaptive". Perhaps they could make the necessary translations themselves.)

In his book[2] he gives a detailed discussion of some of the limitations of non-adaptive feedback and how these might be overcome using adaptive methods. In section 8.21[1], he discusses the "inflexible relationship between sensitivity over system response bandwidth and sensitivity to rate of parameter variation". In particular, he says that

> ".. suppose that in practice the parameter variations are slow. It therefore seems that the design is wasteful in its ability to cope with faster parameter variations than actually occur. It would be extremely desirable to exchange this unrequired benefit of feedback for something else, specifically for reduced system sensitivity to feedback transducer noise."

He goes on to consider a particular example and concludes that

> ".. Some other kind of feedback data-processing is therefore required."

In his book, however, no specific method of adaptive control is treated, and it is left as an open question whether an adaptive controller can, in fact, improve matters.

Since 1963, there has been much work on adaptive control; but much of this work has been isolated from the fundamental issues of feedback control theory. Indeed, all too often, adaptive control has been justified by the erroneous assumption that processes with uncertain dynamics require adaptive control. A recent critique of the field by Kidd[3] states:

> "Many researchers have jumped on the adaptive control bandwagon, but none seem to have publicly taken any trouble to to look deeply at the justifications for using adaptive control."

Another crucial point raised by Kidd[3] is that, too often, adaptive control is used as an alternative to thinking about a control problem in terms of the fundamental principles of feedback control.

This chapter makes a start on bringing together the apparently opposing disciplines of adaptive and non-adaptive control. In particular, we examine the suggestion of Horowitz, mentioned above, that adaptive control can provide a means of reducing the effect of sensor noise when controlling plants with large but slow parameter variations. We use the particular self-tuning controller (generalised minimum variance) for which robustness results have been found in chapter 7.

Following[4], a plant with parameters which, though constant, are uncertain within a prescribed domain is considered. It is assumed that a two degree of freedom[2] high gain controller can be designed to satisfy performance criteria in terms of the system response to setpoint changes, in the face of the plant uncertainty, using the methods of Horowitz and Sidi[2,4], of Ashworth[5] or as simplified by East and Longdon[6,7,8]. It is assumed that these performance criteria are of, or have been converted to[4], the form that the frequency response relating system output to setpoint changes lies between specified bounds for all frequencies $\omega < \omega_c$. Above ω_c, the loop gain is assumed to be reduced as fast as possible consistent with an adequate phase margin[2,4,6,7,8]. Based on this design, a self-tuning algorithm is presented which, by actively reducing uncertainty via parameter estimation, allows the high-frequency loop gain to be reduced, thus reducing the effect of high-frequency sensor noise. Using the robustness results of chapter 6, the design implications of the self-tuning approach are discussed and interpreted as a three degree of freedom design.

This chapter is based on a conference paper[9]

8.2. TWO DEGREE OF FREEDOM DESIGN

In chapter 6 of his book[2], Horowitz shows that, with non-adaptive control, any linear feedback controller for single input-single-output systems

$$\bar{y}(s) = H(s)\hat{u}(s) \tag{1}$$

based on only two measurements (the system output and the setpoint) is equivalent to the two degree of freedom control law:

$$\hat{u}(s) = H_1(s)\bar{w}(s) - H_2(s)\bar{y}(s) \tag{2}$$

displayed in Figure 8.2.1,

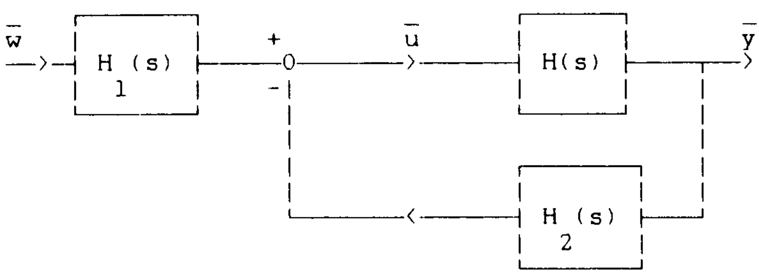


Figure 8.2.1 A two degree of freedom controller

where H(s) is the system to be controlled, $H_1(s)$ and $H_2(s)$ are the two controller transfer functions (giving the two degrees of freedom), $\hat{u}(s)$ is the control signal, $\bar{y}(s)$ is the system (plant) output, and $\bar{w}(s)$ is the setpoint. It is important to realise that any linear control system with these constraints (for example, conditional feedback) may be written in this form[1].

With these two degrees of freedom, there are at least three objectives to be achieved by the control system:

1. Desired response of the system to the setpoint.
2. Insensitivity of the closed-loop system to plant parameter variation.
3. Satisfactory response to plant disturbances and measurement noise.

Sometimes, it is possible to satisfy all three sets of requirements, sometimes it isn't. In particular, requirements 2 and 3 may be conflicting: 2 may require a feedback element $H_2(s)$ with high gain at high-frequencies which could give problems with high-frequency measurement noise, and so conflict with requirement 3.

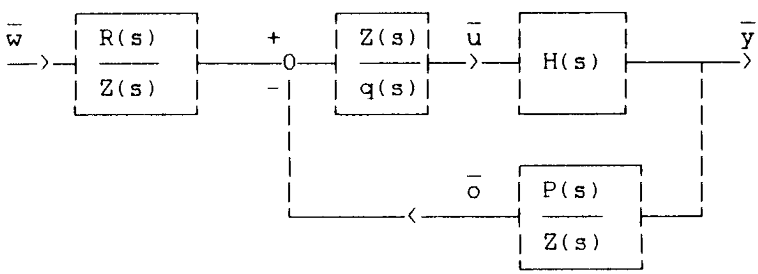
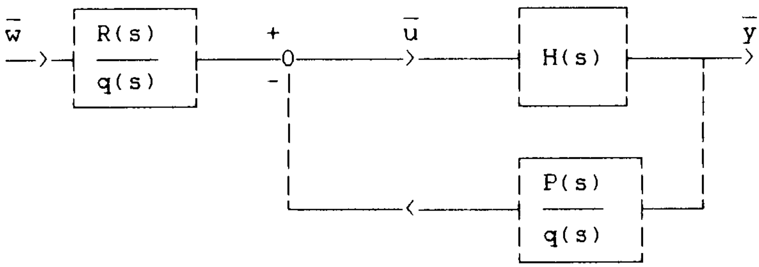


Figure 8.2.2 Another two degree of freedom controller

The two degree of freedom controller can be rewritten (Figure 8.2.2) as:

$$\hat{u}(s) = \frac{Z^-(s)}{q(s)}\left[\frac{R(s)}{Z^-(s)}\bar{w}(s) - \bar{\phi}(s)\right] \quad (3)$$



<u>Figure 8.2.3 A further two degree of freedom controller</u>

$$\bar{\phi}(s) = \frac{P(s)}{Z^{-}(s)}\bar{y}(s) \qquad (4)$$

where $\hat{u}(s)$ is the control signal $P(s)$, $q(s)$ and $Z^{-}(s)$ are polynomials in the operator s; $R(s)$ is a transfer function. This can be reorganised as in Figure 8.2.3, from which it follows that

$$\frac{P(s)}{q(s)} = H_2(s); \quad \frac{R(s)}{q(s)} = H_1(s) \qquad (5)$$

To avoid ambiguity, $P(s)$ is chosen to have unit zero-frequency gain:

$$P(0) = 1 \qquad (6)$$

P is thus the suitably normalised numerator of $H_2(s)$ and Q the corresponding denominator. The polynomial $Z^{-}(s)$ is, at this stage, redundant, but it will be used in the next section. It is chosen to have unit zero-frequency gain and poles further away from the imaginary s-plane axis than those of the system. It follows that <u>both</u> $P(s)/Z^{-}(s)$ and $Z^{-}(s)/q(s)$ are proper:

$$Z^{-}(0) = 1; \ \text{degree}(P) \leq \text{degree}(Z^{-}(s)) \leq \text{degree}(Q) \qquad (7)$$

This control scheme corresponds to the notional feedback loop associated with the detuned model-reference control of section 3.11. In this particular case, the notional feedback loop is realisable.

Example (Horowitz)

The example used in this chapter is drawn from chapter 6 of[2]. The system is of the form:

$$H(s) = \frac{1250K}{s(s^2 + 2\zeta_p\omega_p s + \omega_p^2)} \quad (8)$$

where K may vary from 1 to 4 and the two complex system poles can vary over a wide range with real parts between 0 and -6 and with imaginary parts between j2 and j10.

A design objective is that the closed-loop setpoint response has a dominant pole-pair within circles of radius 1.2 centred at $-10 \pm j10$. A number of design solutions are given by Horowitz[2]; one of these is

$$H_2(s) = 6.2\ 10^9 \frac{s^2 + 18s + 167.5}{(s^2 + 1040s + 590^2)^2} \quad (9)$$

This corresponds to the alternative form where:

$$P(s) = 1 + 1.07\sigma + 0.597\sigma^2 \quad (10)$$

$$q(s) = q(1 + 0.0299\sigma + 0.000287\sigma)^2 \quad (11)$$

where

$$q = 0.1167 \quad (12)$$

and the definition $\sigma = s/10$ has been made for clarity of presentation. P(s) has roots at about $s=-9\pm j9.3$, and q(s) has roots at about $s=-520\pm j280$. Roughly speaking, the compensator $H_2(s)$ is chosen as follows. The compensator zeros are near to the desired closed-loop poles. The compensator has high enough gain to keep the two complex closed-loop poles close to the compensator zeros despite plant parameter variation, and to move the remaining plant pole far to the left. The compensator poles are chosen far enough away from the zeros to avoid stability problems with the far-off closed-loop poles.

The feedback compensator has a gain of just under 20dB at low frequencies, rising to over 70dB. Horowitz comments (section 8.21[2]) that:

> " .. suppose that the system .. has exceedingly slow parameter variations, such that a year may elapse before the poles move from $\pm j2$ to $-6\pm j10$. The final design is very sensitive to high-frequency feedback transducer noise .. but it seems ridiculous that it should be so, in view of the extremely slow parameter variations. Common sense tells us that the feedback data may be evaluated more slowly .. such that high-frequency noise has negligible effect. However .. slower evaluation by means of linear time-invariant networks cannot ensure the desired insensitivity."

The purpose of this chapter is to suggest that the self-tuning emulator-based approach of this book is one possibility to implement the sort of control implied by Horowitz.

8.3. THE EMULATOR

A particular emulator was given in section 3.11 with

$$Z^{-}(s) = P(\varepsilon s); \; Q(s) = \frac{q(s)}{Z^{-}(s)} \tag{1}$$

and so $\bar{\Phi}_2(s)$ is realisable and given by:

$$\bar{\phi}_2(s) = \frac{P(s)}{P(\varepsilon s)} \tag{2}$$

This choice corresponds to the two degree of freedom structure in equations 8.2.3&4.

If the control law

$$\bar{u}(s) = \frac{1}{Q(s)} [\bar{w}(s) - \bar{\phi}(s)] = \frac{P(\varepsilon s)}{q(s)}\bar{w}(s) - \frac{P(s)}{P(\varepsilon s)}\bar{y}(s) \tag{3}$$

is applied (corresponding to the <u>notional</u> <u>feedback</u> <u>system</u>), the disturbance $\bar{v}(s)$ together with a high gain $H_2(s) = P(s)/P(\varepsilon s)$ (as in the Example) can lead to unacceptably large control signals when the high-gain control law of the previous section is used. To see this, the <u>notional</u> closed-loop system may be written as

$$y = \frac{L(s)}{1+L(s)} [\frac{1}{P(s)}R(s)\bar{w}(s) + \frac{q(s)C(s)}{P(s)B(s)}\bar{v}(s)] \tag{4}$$

$$\bar{u}(s) = \frac{L(s)}{1+L(s)} [\frac{A}{BP}R(s)\bar{w}(s) - \frac{C}{B}\bar{v}(s)] \tag{5}$$

where the <u>nominal</u> <u>loop</u> <u>gain</u> $L(s)$ is

$$L(s) = H_2(s)\frac{A(s)}{B(s)} = \frac{P(s)B(s)}{q(s)A(s)} \tag{6}$$

This approach corresponds to implementing the <u>notional</u> <u>feedback</u> <u>system</u> directly; in this particular case, this is possible as $\frac{P(s)}{Z(s)}$ is realisable.

Over the range of frequencies for which $L(s)$ is large, $\bar{v}(s)$ is amplified by the transfer function $C(s)/B(s)$, which will be improper for a system with at least two more poles than zeros - this leads to large control signals.

As a first step in solving this problem, the high gain design is converted into a low gain design via the emulator. This low gain design no longer amplifies the high-frequency noise, but is, of course, sensitive to plant variation. As discussed in the following sections, the long-term sensitivity due to replacing $P(s)/Z(s)$ by the emulator may be overcome by using a self-tuning emulator.

Noting from chapter 2 that $\bar{\phi}(s)$ is the sum of the emulator output $\bar{\phi}^{\star}(s)$ and the error $\bar{e}^{\star}(s)$:

$$\bar{\phi}(s) = \bar{\phi}^{\star}(s) + \bar{e}^{\star}(s) \tag{7}$$

$$\bar{u}(s) = \frac{Z^{-}(s)}{q(s)}\left[\frac{R(s)}{Z^{-}(s)}\bar{w}(s) - \bar{\phi}^{\star}(s)\right] \tag{8}$$

Of course, this only works if the nominal system parameters A and B and the nominal input $\bar{u}(s)$ are available to implement the emulator. In practice, this method is sensitive to parameter uncertainty and the unknown quantity $\bar{u}(s)$ has to be replaced by the known control signal $\hat{u}(s)$, so the advantage of the high gain control is lost. Effectively, another two degree of freedom structure has been created and, as such, has no particular advantages over that of equation 1.

8.4. THREE DEGREE OF FREEDOM DESIGN

The input-output predictor structure removes high-frequency noise at the expense of sensitivity to parameter variation. If, however, plant parameters vary slowly, a self-tuning emulator can be used.

This adaptive algorithm has two additional free polynomials C and $Z^{-}(s)$ in addition to the $P(s)$, $q(s)$ and $R(s)$ already fixed by the two degree of freedom design. These appear in the identity 2.3.4 as a transfer function $C(s)/Z^{-}(s)$ and thus give rise to one more transfer function degree of freedom, making three in all.

As discussed in section 3.11, one possible choice of $Z^{-}(s)$ is:

$$Z^{-}(s) = P(\varepsilon s) \quad 0 < \varepsilon \leq 1 \tag{1}$$

$\varepsilon=1$ gives $Z^{-}(s) = P(s)$ and $\bar{\phi}(s) = y$, and thus the algorithm corresponds to the original two degree of freedom design. On the other hand, $\varepsilon \cong 0$ gives the maximum noise reduction via the self-tuning emulator. Intermediate values allow a trade-off between the two extremes.

Thus the self-tuning approach can be interpreted as a <u>three degree of freedom</u> design method. The additional degree of freedom allows an additional trade-off to be made in the design process.

8.5. ROBUSTNESS

To examine the robustness of controllers to plant uncertainty the uncertainty must be modelled. For simplicity, the disturbances will not be included in the analysis of this chapter. As in chapter 4, the plant is assumed to be linear, and thus can be represented as the <u>nominal</u> plant B(s)/A(s) in series with the <u>neglected dynamics</u> N(s):

$$y = \frac{B(s)}{A(s)}\bar{u}(s); \quad \bar{u}(s) = N(s)\hat{u}(s) \tag{1}$$

where N(s) (see chapter 4) is a transfer function given by

$$N(s) = \frac{\text{actual system}}{\text{nominal system}} = H(s)\,\frac{A(s)}{B(s)} \tag{2}$$

and $\hat{u}(s)$ is the control input. As in chapter 4, this system equation can be rewritten in terms of an additive disturbance $\tilde{u}(s)$ as

$$\bar{y}(s) = \frac{B}{A}[\bar{u}(s) + \tilde{u}(s)] \tag{3}$$

where (in the absence of disturbances):

$$\tilde{u}(s) = (1 - N(s)^{-1})\frac{A}{B}y \quad (4)$$

Two degree of freedom design

Using the two degree of freedom control law (either 2 or 3&4), the closed-loop system response can be written as:

$$\bar{y}(s) = \bar{y}_0(s) + \tilde{y}(s) \quad (5)$$

where $\tilde{y}(s)$ is the output error (compare with $\bar{e}^y(s)$ in chapter 7). The nominal system output is

$$\bar{y}_0(s) = \frac{L(s)}{1+L(s)}\,\frac{1}{P(s)}w \quad (6)$$

L is the nominal loop gain and $\tilde{y}(s)$ is given by

$$\tilde{y}(s) = \frac{B}{A}\,\frac{1}{1+L(s)}\tilde{u}(s) = \bar{\Delta}_0(s)\ y \quad (7)$$

where

$$\bar{\Delta}_0(s) = \frac{1 - N(s)^{-1}}{1+L(s)} \quad (8)$$

The two degree of freedom design method[2] as simplified by East[6,7,8] is based on making $\bar{\Delta}_0(s)$ sufficiently small at each frequency w within a frequency band $0 \leq \omega \leq \omega_c$ to satisfy design specifications. For $w > \omega_c$, the nominal loop gain is reduced as rapidly as possible.

Alternatively, the output error can be expressed as

$$\tilde{y}(s) = \bar{\Delta}(s)\ \bar{y}_0(s) \quad (9)$$

where

$$\bar{\Delta}(s) = \frac{\bar{\Delta}_0(s)}{1-\bar{\Delta}_0(s)} = \frac{1 - N(s)^{-1}}{L(s) + N(s)^{-1}} \quad (10)$$

Typically, the design will ensure that $\bar{\Delta}_0(s)$ is small for $\omega < \omega_c$; in addition, if N(s) represents low-pass dynamics, N^{-1} will be large at high-frequencies and so

$$\bar{\Delta}(s) \cong -1 \text{ for sufficiently large } \omega > \omega_c \tag{11}$$

Three degree of freedom design.

There is an additional source of error when applying self-tuning control: $\hat{e}(s) = \bar{\phi}(s) - \hat{\phi}(s) \neq 0$. It is shown in chapter 5 that

$$\hat{e}(s) = \Omega\bar{e}(s) \tag{12}$$

where Ω is a time-varying system representing the tuning algorithm. In addition , as discussed in chapters 4 and 7, the estimator input error is related to the estimation error, the setpoint and disturbances by

$$\bar{e}(s) = \bar{e}^d(s) - M(s)\hat{e}(s) \tag{13}$$

where

$$M(s) = \frac{E(s)A(s)}{P(s)C(s)} L(s)\bar{\Delta}(s) = \frac{Z^+(s)E(s)A(s)[N^{-1}(s)-1]}{P(s)C(s)[1+L^{-1}(s)N^{-1}(s)]} \tag{14}$$

These equations form a feedback system. It is shown in chapter 7 that a sufficient condition for stability is that the gain of the linear transfer function M(s) be less than one at all frequencies.

The system output is given by

$$\bar{y}(s) = \bar{y}_0(s) + \tilde{y}(s) + \frac{L(s)}{1+L(s)}\hat{e}(s) \tag{15}$$

As well as requiring $\tilde{y}(s)$ to be small, we require $\hat{e}(s)$ to be small. This implies that M should be small at the relevant frequencies.

8.6. COMPARATIVE ROBUSTNESS

The aim of each design method is to make the system output y sufficiently close to the nominal system output y0 to satisfy the design objectives within the frequency range $0 \le \omega \le \omega_c$.

It is important to distinguish between the methods used by the non-adaptive and adaptive controllers to reduce the effect of plant uncertainty. In the non-adaptive case, the nominal plant B(s)/As is chosen by the designer, and this implies the value of $N(s) = H(s)A(s)/B(s)$. In the adaptive case, however, all that is required is that a suitable nominal plant B/A exist so that, together with the corresponding value of N, the robustness conditions are satisfied. If B/A had the same structure as H(s), such a nominal system would be $B(s)/A(s) = H(s)$ and N=1 and so the robustness conditions would be satisfied. But, in practice, this would not normally be the case. Indeed, for the purposes of this discussion, it will be assumed that the neglected dynamics are low-pass:

$$\operatorname*{Lt}_{\omega \to \infty} N(j\omega) = 0 \tag{1}$$

and hence that

$$\operatorname*{Lt}_{\omega \to \infty} \Delta(j\omega) = -1 \tag{2}$$

Two degree of freedom design

The two basic design rules for two degree of freedom non-adaptive design are (roughly speaking)[4,5,6,7,8]:

NA1. $\bar{\Delta}_0(s)(j\omega)$ must be sufficiently small for $\omega < \omega_c$ to satisfy the design constraints.

NA2. $L(j\omega)$ must be reduced as fast as possible (consistent with adequate phase margin) for $\omega > \omega_c$.

The first rule gives insensitivity to plant variation; the second reduces the effect of high-frequency sensor noise as much as possible.

Three degree of freedom design

The self-tuning method also requires that the underlying design method be insensitive to plant variations, so the first adaptive design rule is the same as the first non-adaptive design rule:

A1. NA1

In addition, it is required that $M(j\omega)$ be small at all frequencies. The two frequency ranges above and below ω_c are considered separately.

$\omega < \omega_c$ Here $L(s)$ is large, so $L\bar{\Delta}(s) \cong 1-N^{-1}$. The adaptive controller must thus be capable of reducing the uncertainty $N(s)$ in this frequency range. Hence the second design rule is:

A2. The structure of the adaptive emulator must be such as to capture all significant plant dynamics at frequencies $\omega < \omega_c$:

$\omega > \omega_c$ degree(EA) = degree(PC), so for high frequencies $EA/PC \to \kappa$, where κ is a non-zero constant. In addition, $\bar{\Delta}(s) \cong 1$, and so $M \cong -\kappa L$. Hence, $L(s)$ must be small at high frequencies and thus the third adaptive design rule is the same as the second non-adaptive design rule:

A3. NA2

As pointed out by Horowitz and Sidi[4], minimum phase systems can, in principle, support a feedback control design with infinite loop-gain at all frequencies; but this is undesirable for reasons of sensor noise. Hence, design rule NA2 is used in practice. The arguments leading to design rule A3 show that, for the adaptive case, such an infinite notional-loop gain approach is not merely undesirable but leads to a design which cannot satisfy A3. Thus a pure model-reference approach with 1/P as the desired model and Q=0 is not feasible in practice. Although the algorithms are different, this conclusion is in accordance with those of Rohrs and colleagues[10] concerning the impracticality of model-reference adaptive control.

8.7. SUMMARY

An initial attempt has been made to unite the non-adaptive and adaptive approaches to feedback control for a particular, but important, case: a single-input single-output system with constant but uncertain parameters where, although non-adaptive control can yield the desired insensitivity, the resultant amplification of sensor noise is unacceptable. It is suggested that the non-adaptive design is a prerequisite to the adaptive design; this is in distinction to the commonly held view that the use of adaptive control avoids design. In particular, the pure model-reference version of the algorithm in this chapter, which attempts to match the closed-loop system to the reference model 1/P at all frequencies, is not a practical algorithm.

Much work remains to be done in this area. Detailed design examples are required to refine the broad outline presented in this chapter. It would seem that a similar approach could be applied to the multivariable and cascade controller configurations of the following chapters.

An interesting extension of these ideas would be to consider significantly non-minimum phase systems (with time-delay or right half-plane zeros) where these characteristics are removed from the notional system by the emulator.

References

1. Harris, C. and Billings, S., and Horowitz, I., "Plant adaptive systems versus ordinary feedback systems," IRE Trans, vol. AC-7, no. 1, pp. 48-56, Peter Peregrinus, London, 1962.

2. Horowitz, I., Synthesis of feedback systems, Academic Press, 1963.

3. Kidd, P.T., "Comments on: Self-tuning and stable adaptive control of a batch polymerization reactor'," Automatica, vol. 20, no. 4, p. 481, 1984.

4. Horowitz, I. and Sidi, M., "Synthesis of feedback systems with large plant ignorance for prescribed time-domain tolerances," International Journal of Control, vol. 16, pp. 287-309, 1972.

5. Ashworth, M.J., Feedback design of systems with significant uncertainty, Research Studies Press, Lechworth, Herts, U.K., 1982.

6. East, D,J., "A CAD procedure for optimum loop synthesis," Report RT 146/83, RMCS, Shrivenham, 1983.

7. East, D.J. and Longdon, L.W., "Sensitivity synthesis of systems subject to large parameter variations," in Proceedings of the IEE conference 'Control and its applications', University of Warwick, U.K. (IEE conference publication 194), 1981.

8. East, D.J., "Passive adaptive control of systems with large plant uncertainty," in Proceedings of the IEE workshop on ' The theory and applications of adaptive control', University of Oxford, U.K., 1983.

9. Gawthrop, P.J., "Comparative robustness of non-adaptive and adaptive control," in Proceedings of the IEE

conference "Control '85", Cambridge, U.K., 1985.

10. Rohrs, C.E., Valavani, L., Athans, M., and Stein, G., "Robustness of continuous-time adaptive control in the presence of unmodeled dynamics," Trans. IEEE, vol. AC-30, pp. 881-889, 1985.

CHAPTER 9
Cascade Control

Aims. To consider the cascade control of single-input single-output systems with a number of measurable signals available. To introduce a recursive emulator approach to cascade control.

9.1. INTRODUCTION

If self-tuning methods are to be widely used in real applications, it must be possible to use self-tuning controllers as components within a larger multi-loop control system. The current practice in the process control industry is that a control scheme for a multi-loop process is built up out of a number of simple modules rather than from one complex multi-loop algorithm. The philosophy behind this chapter is to develop a similar approach for self-tuning algorithms - they should be a simple component out of which complex control schemes may be created.

As part of this process, simple standard multi-loop configurations are under investigation. This chapter considers a standard configuration: cascade control; the next chapter considers decoupling control of two-input two-output systems. With the exception of[1], cascade control has received little attention in the context of self-tuning. Derivative generating (model-reference) type emulators (section 2.2) in cascade control are discussed in[2]; this chapter extends the results to cover all of the emulators of this book.

9.2. CASCADE SYSTEMS

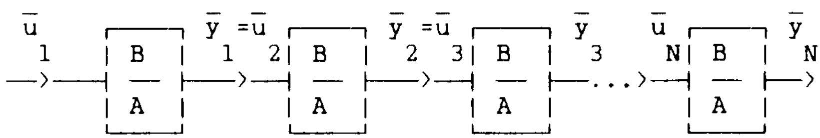


Figure 9.2.1 Cascaded systems

A class of systems to which cascade control is appropriate is given by the series connection of a number of systems of the form (Figure 9.2.1):

$$\bar{y}_i(s) = e^{-sT_i}\frac{B_i(s)}{A_i(s)}\bar{u}_i(s) + \bar{v}_i(s) \qquad (1)$$

(For simplicity, initial conditions will be ignored in this chapter). The series interconnection is specified by:

$$\bar{u}_i(s) = \bar{y}_{i-1}(s); \; i = 2..N \qquad (2)$$

The (single) output to be controlled is y_N; the (single) input available for control is $u_1(s)$. The disturbances are as described in section 1.9.

It is common in the process industry to have a number of measurements pertaining to various stages of a given process; current self-tuning methods cannot use such information. The algorithm presented in this chapter goes some way to filling this gap.

9.3. POSSIBLE CASCADE METHODS

There are a number of ways of extending the single-loop methods of earlier chapters to control the cascade systems of equations 9.2.1&2. Some of these will now be considered. For simplicity, assume $\bar{v}_i(s)=0$ for the rest of this section. For each method, advantages are indicated by "(+)" and disadvantages by "(-)"

Single-loop control

One possible strategy is to ignore the intermediate signals $\bar{y}_i(s)$ $i = 1..N-1$, and just have a single-loop self-tuning controller using y_N as output and $y_0 = u_1(s)$ as input.

(+) This requires no special algorithm.

(-) The single self-tuner must correspond to a system with order equal to the sum of the subsystem orders. This may be large.

(-) When ignoring the additional information provided by the intermediate outputs, the system is more difficult to control in terms of both phase lag and disturbance rejection.

Ignoring inner loops

A common way to implement cascade control loops is to ignore the dynamics of loops inside the one being designed. That is, having closed i-1 cascaded loops to give a system:

$$\bar{y}_{i-1}(s) = S_{i-1}(s)\bar{w}_{i-1}(s) \qquad (1)$$

(where w_i is the setpoint to the ith controller); the ith loop is designed as if $S_{i-1}(s) = 1$. The approximation is thus that the input to the ith system (the output of the i-1th system) follows the i-1th setpoint exactly:

$$\bar{w}_{i-1}(s) \cong \bar{y}_{i-1}(s) \tag{2}$$

(+) Each individual self-tuner has structure corresponding to the relevant subsystem. The order of the subsystem may be much less than that of the overall system. Thus control is easier and disturbance rejection improved.

(+) By using the additional information provided by the intermediate outputs, the system is made easier to control in terms of both phase lag and disturbance rejection.

(-) The result will only be satisfactory if the individual subsystems are ordered in terms of increasing time constant. If the dynamics of the i-1th loop are not negligible with respect to the ith loop, poor performance and even instability may result.

Taking account of inner loops

The problems encountered in the previous section may be overcome by including the dynamics of inner loops in the design of the outer loops. That is, using the notation of the previous section, the ith loop is designed on the basis of:

$$\bar{y}_i(s) = \frac{B_i(s)}{A_i(s)} S_{i-1}(s)\bar{w}_{i-1}(s) \tag{3}$$

(+) Dynamics are not neglected; the dynamics of the inner loops do not affect the accuracy or stability of the final design.

(+) By using the additional information provided by the intermediate outputs, the system is made easier to control in terms of both phase lag and disturbance rejection.

(-) The compexity of the design increases with the loop index i. Indeed, the outer loop is of the same complexity as that of single-loop control.

<u>The recursive emulator method</u>

In view of the above methods, there seems to be a need for a method which will handle cascaded systems with similar time-constants while retaining a simple structure based on N self-tuners operating on the N measured outputs. This algorithm is introduced in the next section; here its merits in with respect to the other methods are outlined:

(+) Each self-tuning emulator operates on a subsystem and is thus simple.

(+) The effect of inner loops is exactly allowed for.

(+) By using the additional information provided by the intermediate outputs, the system is made easier to control in terms of both phase lag and disturbance rejection.

(-) The reference model for each loop must be identical. This implies that each subsystem have similar dynamics.

(-) An additional level of coordination is required when compared to the cascade method ignoring inner loops.

The method presented is not the only possible, but it is felt that it strikes a balance between complexity and flexibility of use.

9.4. THE RECURSIVE EMULATOR METHOD

The aim of this method is to give a closed-loop system:

$$\bar{y}_N(s) = e^{-sNT} \frac{Z^N(s)}{P^N(s)} \bar{w}(s) \tag{1}$$

with the restrictions that:

$$\deg(P) = \deg(A_j(s)) - \deg(B_j(s)); \; T = T_j \tag{2}$$

for all j = 1..N.

To achieve this, define:

$$\phi_{i,j} = \left| e^{sT} \frac{P(s)}{Z(s)} \right|^i y_j \tag{3}$$

The emulator with i=1 corresponding to each individual system is given by:

$$\phi^{\star}_{1,j} = \frac{F_j(s)}{C_j(s)} y_j + \frac{G_j(s)}{C_j(s)} y_{j-1} \tag{4}$$

where:

$$\frac{P(s)C_j(s)}{A_j(s)} = E_j(s) + \frac{F_j(s)}{A_j(s)} \tag{5}$$

and:

$$G_j(s) = B_j(s)E_j(s) \tag{6}$$

Once again, $C_j(s)$ is <u>chosen</u> for each subsystem. The corresponding error is then:

⋆ The subscripts refer to the loop index, <u>not</u> to the emulator version

$$e_{1,j} = E_j(s)z \tag{7}$$

A recursive expression for $\phi_{i,j}$ may be obtained from these definitions as follows:

$$\phi_{i,j} = P^i(s)\phi_{1,j} \tag{8}$$

$$= P^i(s)\phi^{\star}_{1,j} + P^i(s)e_{1,j}$$

Using the above definitions, this can be further expanded as:

$$\phi_{i,j} = \frac{F_j(s)}{C_j(s)}\phi_{i-1,j} + \frac{G_j(s)}{C_j(s)}\phi_{i-1,j-1} \tag{9}$$

$$+ P^i e_{i,j}$$

There are many possible approximations to $\phi_{i,j}$ but to be useful they must have the following properties:

a) The approximation error must depend only on disturbances, not on the control signal. That is, the approximation does not affect closed-loop stability.

b) The approximation must be realisable; it must not contain derivatives of disturbance terms.

As both $\frac{F_j(s)}{C_j(s)}$ and $\frac{G_j(s)}{C_j(s)}$ are proper, a realisable emulator $\phi^{\star}_{i,j}$ may be <u>defined</u> as:

$$\phi^{\star}_{i,j} = \frac{F_j(s)}{C_j(s)}\phi^{\star}_{i-1,j} + \frac{G_j(s)}{C_j(s)}\phi^{\star}_{i-1,j-1} \tag{10}$$

The corresponding error $e_{i,j}$ is defined as:

$$e_{i,j} = \phi_{i,j} - \phi^{\star}_{i,j} \tag{11}$$

The recursive formula for the error is then:

$$e_{i,j} = P^{i}(s)e_{1,j} + \frac{F_j(s)}{C_j(s)}e_{i-1,j} \qquad (12)$$

$$+ \frac{G_j(s)}{C_j(s)}e_{i-1,j-1}$$

The recursive emulator for a 3-loop cascade system appears in Figure 9.4.1.

9.5. SELF-TUNING CASCADE CONTROL

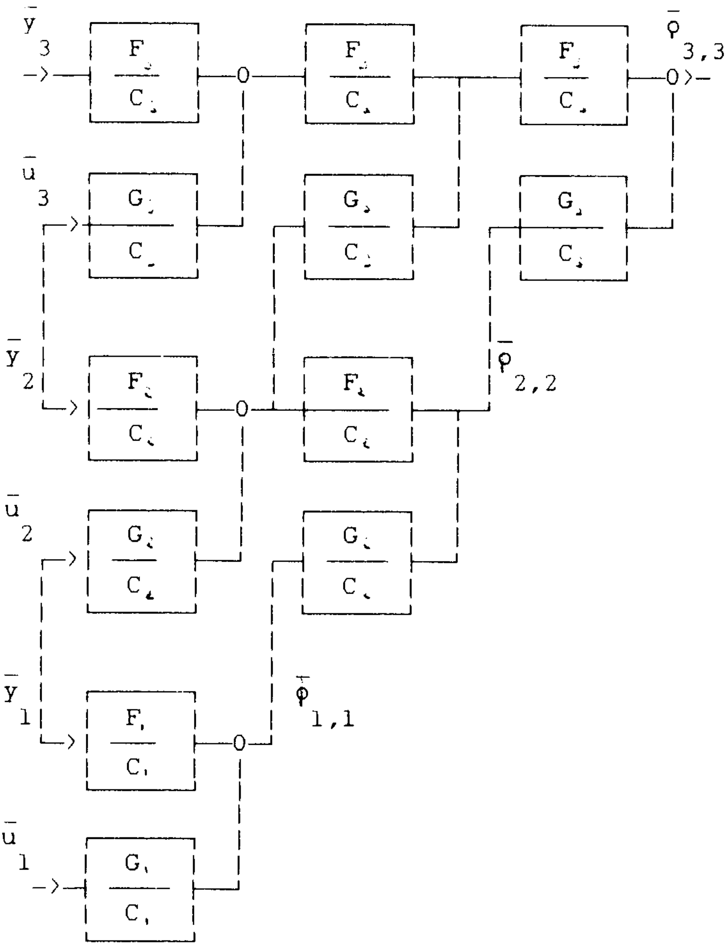
To implement the recursive emulator for an N-loop cascade control system, the N polynomial pairs $\{F_j(s), G_j(s)\}$ are required. It is proposed that these be generated (together with estimates for $\phi^{*}_{1,j}$) using N self-tuning emulators of $\phi_{1,j} = Py_j$, each operating on one of the N systems of equation 9.1.1. The control signal $\bar{u}(s)$ $(=y_0)$ may be generated in two stages:

1. Compute the emulator outputs: $\hat{\phi}^{i,j}$ which have no direct link to the control signal u, that is for i<j. This gives the N values $\hat{\phi}^{i-1,i}$ for i=1..N.

2. Letting $\hat{\phi}^{N,N} = w$, compute $\hat{\phi}^{i,i}$ for i = N-1..0 using equation 9.4.9. The control signal is then $u = \hat{\phi}^{0,0}$.

9.6. EXAMPLES

To illustrate the two non-adaptive cascade control methods, consider a double integrator system (see Figures 9.6.1&2) where the output of each integrator can be



<u>Figure 9.4.1 The recursive emulator</u>

measured. That is:

$$\bar{y}_2(s) = \frac{1}{s}\,\bar{y}_1(s);\quad \bar{y}_1(s) = \frac{1}{s}\,u_1(s) \tag{1}$$

For each control method, the objective is to give a setpoint tracking response corresponding to a critically damped system given by:

$$\frac{Z(s)}{P(s)} = \frac{1}{(1+ps)^2} \tag{2}$$

i.e.

$$Z(s) = 1;\ P(s) = (1+ps)^2 \tag{3}$$

Single-loop control

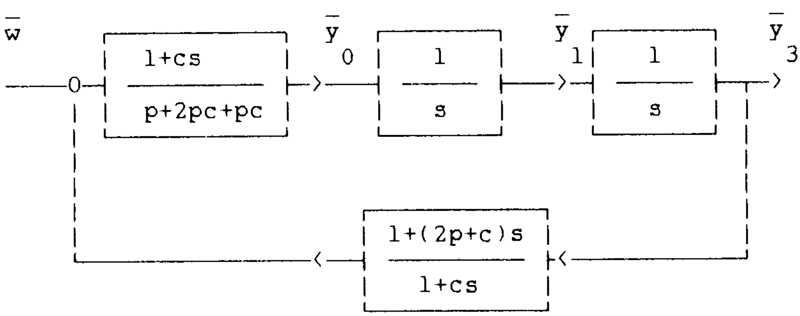


Figure 9.6.1 Single-loop control

If the intermediate variable is not used, a filter polynomial $C(s) = 1+cs$ must be used to give a realisable control law (without derivatives). The left-hand side of identity 9.4.5 becomes:

$$\frac{P(s)C(s)}{A(s)} \tag{4}$$

$$= \frac{1 + (2p+c)s + (2pc+p^2)s^2 + p^2cs^3}{s^2}$$

This gives:

$$E = (2pc+p^2) + p^2cs \tag{5}$$

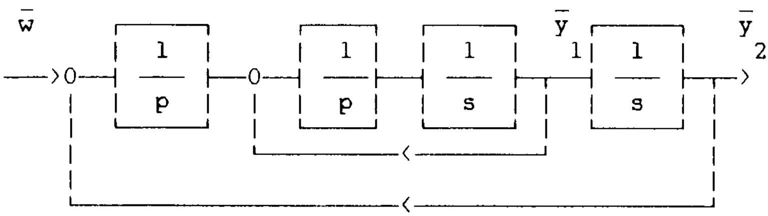
$$F = 1 + (2p+c)s \tag{6}$$

The resultant feedback control law appears in Figure 9.6.1, and may be written as:

$$u_1(s) = \frac{1+cs}{(2pc+p^2) + p^2cs} \tag{7}$$

$$[\bar{w}(s) - \frac{1 + (2p+c)s}{1+cs}\bar{y}(s)]$$

<u>Ignoring inner loops</u>



<u>Figure 9.6.2 Ignoring inner loops</u>

Choose both the inner loop controller and the outer loop controller (ignoring inner loop) to give a setpoint response:

$$\frac{1}{P(s)} = \frac{1}{1+ps} \tag{8}$$

In this case, a filter C is not required and the left-hand side of identity 9.4.5 becomes:

$$\frac{P(s)C(s)}{A(s)} = \frac{1+ps}{s} \tag{9}$$

This gives controller polynomials:

$$E(s) = p; \; F(s) = 1 \tag{10}$$

If the dynamics of this inner loop are ignored, the outer loop dynamics are, in this case, identical to the open-loop inner loop dynamics. Thus the outer loop controller is the same as the inner loop controller. This gives:

$$u_1(s) = \frac{1}{p^2}[\bar{w}(s) - \bar{y}_2(s)] - \frac{1}{p}\bar{y}_1(s) \tag{11}$$

This is shown in Figure 9.6.2. The closed-loop setpoint response is, of course, not correct. It is given by:

$$\bar{y}_2(s) = \frac{1}{1+ps+p^2s^2}\bar{w}(s) \tag{12}$$

<u>Taking account of inner loops</u>

The design of the inner loop is the same as in the previous section.

The system response, with the inner loop closed, from the inner loop setpoint to $\bar{y}_2(s)$ is then:

$$\frac{1}{s(1+ps)} \tag{13}$$

As in the previous section, the outer loop controller requires a filter C, again chosen as $C(s) = 1+cs$. The left-hand side of identity 9.4.5 becomes:

$$\frac{P(s)C(s)}{A(s)} = \frac{(1+ps)^2(1+cs)}{(1+ps)s} \tag{14}$$

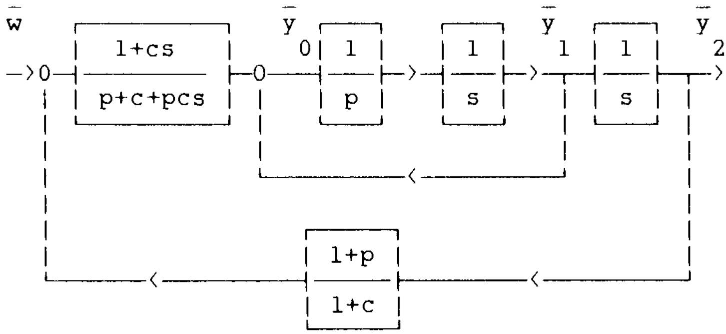


Figure 9.6.3 Taking account of inner loops

$$= \frac{1 + (p+c)s + pcs^2}{s}$$

It follows that:

$$E = (p+c) + pcs; \; F = 1+ps \tag{15}$$

The resultant feedback control law appears in Figure 9.6.3 and may be written as:

$$u_1(s) = \frac{1}{p}\Big[\frac{1+cs}{p+c + pcs}\big[\bar{w}(s) - \frac{1+ps}{1+cs}\,\bar{y}_2(s)\big] - \bar{y}_1(s)\Big] \tag{16}$$

The recursive emulator method

As the two cascaded systems are identical, the corresponding polynomial identities are the same and given by:

$$\frac{P(s)C(s)}{A(s)} = \frac{1+ps}{s} \tag{17}$$

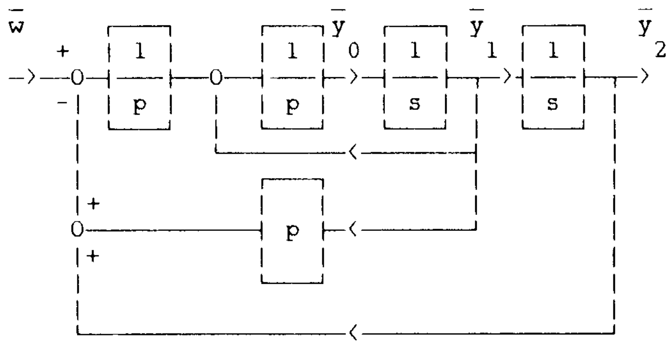


Figure 9.6.4 The recursive emulator method

This gives:

$$E_1(s) = E_2(s) = p;\ F_1(s) = F_2(s) = 1 \tag{18}$$

The three emulators are thus given by:

$$\phi^*_{1,1} = \bar{y}_1(s) + pu_1(s) \tag{19}$$

$$\phi^*_{1,2} = \bar{y}_2(s) + p\bar{y}_1(s) \tag{20}$$

$$\phi^*_{2,2} = \phi^*_{1,2} + p\phi^*_{1,1} \tag{21}$$

The resultant control law appears in Figure 9.6.4 and may be written as:

$$u_1(s) = \frac{1}{p^2}[\bar{w}(s) - 2p\bar{y}_1(s) - \bar{y}_2(s)] \tag{22}$$

This may be compared with the single-loop controller of Figure 9.6.2. In equation 22 (in the absence of disturbances), $\bar{y}_1(s) = s\bar{y}_2(s)$. This equation becomes the same as

7 if, in that equation, C=1. However, in practice, such differentiation is inadmissible.

These examples illustrate that the recursive emulator method leads to the simplest control law giving the desired closed-loop system. Moreover, the two corresponding self-tuning emulators operate on first-order systems; the first and third each require a self-tuning emulator operating on a second-order system. Finally, unlike the third example, the controller parameters for the outer loop do not depend on those for the inner loop.

References

1. Anbumani, K., Sarma, I.G., and Patnaik, L.M., Self-tuning cascade control of non-linear systems, IFAC Symposium on Theory and Applications of Digital Control, New Delhi, 1982.

2. Gawthrop, P.J., "Multi-loop self-tuning control: Cascade systems," in Preprints of the 9th IFAC triennial world congress., ed. K.J. Astrom, vol. VII, pp. 127-132, Budapest, 1984.

CHAPTER 10

Two-Input Two-Output Systems

Aims. To consider the control of two-input two-output systems using two self-tuning controllers with feedforward. To analyse the robustness of the self-tuning control in the presence of neglected loop-interaction dynamics.

10.1. INTRODUCTION

A typical process control system will involve many control loops. Often, some of these loops will involve mutually interacting systems. It follows that if self-tuning methods are to be of use in large process control systems, they must be able to perform satisfactorily in such an interactive environment. One approach to the control of a number of interacting loops forming an n-input n-output system is to use a single multivariable self-tuner. Such approaches have been reported in the literature[1,2,3].

Of course, the distinction between the two approaches is vague. Borisson[1] has shown that a multi-loop self-tuning regulator may be viewed as a number of single-loop controllers with a shared database, Morris, Nazer and Wood[3] and Peel, Morris and Tham[4] also make this point.

Nevertheless, an advantage of the one-loop philosophy is that from the start it implies that a multiloop process should be controlled using a number of autonomous (from both the hardware and software points of view) one-loop modules. This is in keeping with the current trend towards decentralised distributed control systems based on microprocessor units connected via a local area network. The particular algorithm used here is the detuned model-reference controller of section 3.10; but it would seem that the results extend to other emulator-based self-tuning controllers. As noted in chapters 3 and 6, this algorithm has PI and PID versions[5,6].

This latter approach gives rise to two distinct problems addressed in this chapter:

a) Do self-tuners, designed as if there were no loop interaction, behave satisfactorily if interaction is present?

b) Can self-tuners be modified to account for interaction and, if so, do they then behave satisfactorily?

This chapter is limited to a two-input, two-output system. The extension to n-input n-output systems with neglected dynamics in the forward path is given elsewhere[7]. For such a system, this chapter provides a theoretical analysis of each question. Both design and analysis are based on methods introduced in earlier chapters of this book in the context of single-loop control. In this chapter the detuned model-reference controller of section 3.10 is discussed; however, the main idea would seem to apply to other algorithms. The design follows a three-stage process: a notional feedback loop design, a corresponding emulator-based design and finally a self-tuning emulator design; the analysis uses the input-output methods of chapter 7. This chapter concentrates on the additional design and analysis required in the two-loop case. In the single-input single-output case (Chapter 7), the robustness problem arises from unmodelled dynamics in the transfer function relating input to output; here it arises from unmodelled, or partially modelled, interaction terms.

The analysis of the two-input two-output adaptive and non-adaptive decoupling methods of this chapter is much simplified by the use of a representation whereby interaction is modelled by system outputs being coupled to system inputs. This approach is found in certain works on the analysis of large-scale systems, for example[8,9]. This is in contrast to the usual transfer function matrix approach where interaction arises from coupling from inputs to outputs. In this chapter, the former representation is called the feedback interaction model, and the latter representation is called the feedforward interaction model. In the the case of two-input, two-output systems, these models are related via the relative gain array of Bristol[10]. These two alternative models have been discussed in the chemical engineering literature: the feedforward model has been called the P-canonical structure and the feedback model the V-canonical structure[4,11,12,13].

Robustness results are derived for four cases: with and without decoupling and with and without adaptation. This chapter is based on an internal report[14].

The chapter is organised as follows. Section 2 presents the feedback interaction model of two-input two-output systems and examines the relationship of this model to other representations. Three illustrative examples are given. Section 3 describes non-adaptive and self-tuning methods for the control of two-input two-output systems. As this self-tuning method has been discussed in earlier chapters, section 3 mainly considers the additional detail required for the two-loop case. In section 4, it is shown that the two-loop self-tuning control method is associated with a single-loop error feedback system. Section 5 presents the non-adaptive robustness results, and section 6 the corresponding adaptive results. Section 7 concludes the chapter.

10.2. THE SYSTEM

The interactive two-input two-output system considered here is displayed in Figure 10.2.1, and is described by:

$$\bar{y}_1(s) = S_{11}(s)[\bar{u}_1(s) + S_{12}(s)\bar{y}_2(s)] \tag{1}$$

$$\bar{y}_2(s) = S_{22}(s)[\bar{u}_2(s) + S_{21}(s)\bar{y}_1(s)] \tag{2}$$

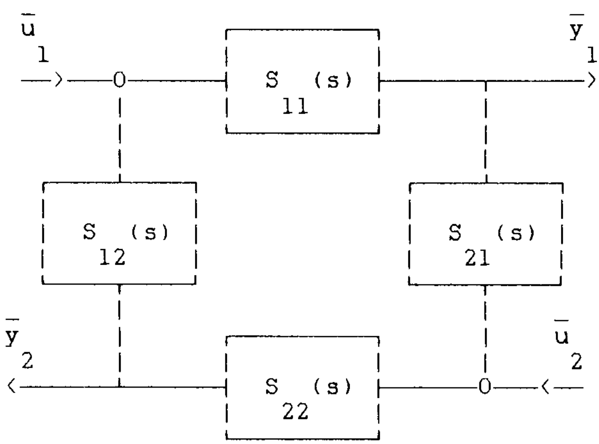


Figure 10.2.1 The open-loop system

Disturbances may be included in the algorithms and in the subsequent analysis, but for clarity and simplicity, this aspect is ignored in this chapter. Similarly, initial conditions are not treated here.

The two interacting systems have outputs $\bar{y}_1(s)$ and $\bar{y}_2(s)$. The interaction is a consequence of the transfer functions $S_{21}(s)$ and $S_{12}(s)$.

Equations 2.1 and 2.2 may be rewritten in matrix form as:

$$\underline{y} = \underline{S}_1(\underline{u} + \underline{S}_2\underline{y}) \tag{3}$$

where

$$\underline{S}_1 = \begin{vmatrix} S_{11}(s) & 0 \\ 0 & S_{22}(s) \end{vmatrix}; \quad \underline{S}_2 = \begin{vmatrix} 0 & S_{12}(s) \\ S_{21}(s) & 0 \end{vmatrix}$$

and

$$\underline{y} = \begin{vmatrix} \bar{y}_1(s) \\ \bar{y}_2(s) \end{vmatrix}; \quad \underline{u} = \begin{vmatrix} \bar{u}_1(s) \\ \bar{u}_2(s) \end{vmatrix}$$

$\underline{S}_1$ is a <u>diagonal</u> transfer function matrix, $\underline{S}_2$ is an <u>off-diagonal</u> transfer function matrix and $\underline{y}$ and $\underline{u}$ are vectors of outputs and inputs respectively. Equation 2.3 will be called the <u>feedback</u> interaction model in this chapter. This structure seems quite general (for a linear two-input two-output system), as other structures (such as coupling from input to input) can be incorporated by suitably redefining the various transfer functions.

For example, a common system model is:

$$\bar{y}_1(s) = R_{11}(s)\bar{u}_1(s) + R_{12}(s)\bar{u}_2(s) \tag{4}$$

$$\bar{y}_2(s) = R_{22}(s)\bar{u}_2(s) + R_{21}(s)\bar{u}_1(s) \tag{5}$$

This may be written in the usual transfer-function matrix form as:

$$\underline{y} = \underline{R}\underline{u} \tag{6}$$

Equation 2.6 will be called the feedforward interaction model. In this two-input two-output case, $\underline{R}$ is given in terms of S by:

$$\underline{R} = [1 - \underline{S}_1\underline{S}_2]^{-1}\underline{S}_1 \tag{7}$$

or, in terms of the individual elements, by:

$$\underline{R}_{11}(s) = [1-L_I(s)]^{-1}S_{11}(s) \tag{8}$$

$$\underline{R}_{12}(s) = [1-L_I(s)]^{-1}L_I(s)S_{21}(s)^{-1} \tag{9}$$

where the interaction loop-gain $L_I(s)$ is given by:

$$L_I(s) = S_{11}(s)S_{12}(s)S_{22}(s)S_{21}(s) \tag{10}$$

$\underline{R}_{22}(s)$ and $\underline{R}_{21}(s)$ are given by similar equations.

Similarly, $\underline{S}$ is given in terms of $\underline{R}$ by:

$$S_{11}(s) = \underline{R}_{11}(s) - \underline{R}_{12}(s)\underline{R}_{22}(s)^{-1}\underline{R}_{21}(s) \tag{11}$$

$$S_{12}(s) = S_{11}(s)^{-1}\underline{R}_{12}(s)\underline{R}_{22}(s)^{-1} \tag{12}$$

The former representation (using the $\underline{S}_1$ and $\underline{S}_2$ matrices) gives the simplest results for the analysis given here. It also arises naturally in some physical systems as demonstrated by the following example.

Example 1: Output coupled tanks

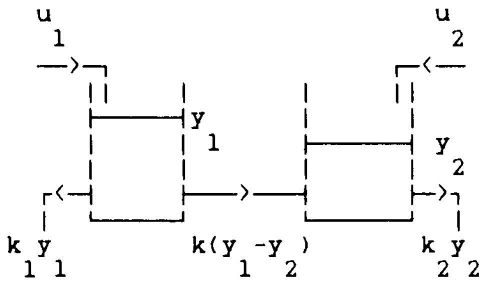


Figure 10.2.2 Output coupled tanks

The system of two coupled tanks displayed in Figure 10.2.2 will be used for motivating and illustrating the results presented here. It has been used previously by Owens[15].

Assuming each tank has unit cross-sectional area, and that the flow out of each tank is proportional to the heights and the flow between the tanks proportional to the difference in heights, it follows that:

$$y_1 = \bar{u}_1(s) - k_1\bar{y}_1(s) - k_2(\bar{y}_1(s) - \bar{y}_2(s)) \quad (13)$$

In terms of the feedback interaction model 10.2.3, this gives:

$$S_{11}(s) = S_{22}(s) = \frac{1}{s+a};\ S_{12}(s) = S_{21}(s) = k \quad (14)$$

where: $a = k_1 + k_2$ and $k = k_2$. The interaction loop gain is:

$$\frac{k^2}{(s+a)^2} \quad (15)$$

In terms of the feedforward interaction model:

$$\underline{R}_{11}(s) = \underline{R}_{22}(s) = \frac{s+a}{(s+a)^2 - k^2} \tag{16}$$

$$\underline{R}_{12}(s) = \underline{R}_{21}(s) = \frac{k}{(s+a)^2 - k^2} \tag{17}$$

□

The feedback interaction model may, as shown by the following example, be used when a feedforward interaction model arises directly from the physical problem.

Example 2: Input coupled tanks

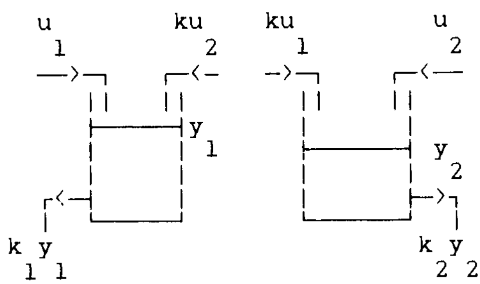


Figure 10.2.3 Input coupled tanks

Consider the two-input coupled tanks in Figure 10.2.3. The input to tank 1 is $\bar{u}_1(s) + k\bar{u}_2(s)$ and vice versa. It is readily shown that the dynamics of tank 1 are given by:

$$\bar{y}_1(s) = \frac{1}{s+a}(\bar{u}_1(s) + k\bar{u}_2(s)) \tag{18}$$

and similarly for tank 2. Thus in feedforward interaction form:

$$\underline{R}_{11}(s) = \underline{R}_{22}(s) = \frac{1}{s+a}; \; \underline{R}_{12}(s) = \underline{R}_{21}(s) = \frac{k}{s+a} \qquad (19)$$

Using equations 10.2.11&12, the feedback interaction model becomes:

$$S_{11}(s) = S_{22}(s) = \frac{1 - k^2}{s+a}; \; S_{12}(s) = S_{21}(s) = \frac{k(s+a)}{1 - k^2} \qquad (20)$$

and the interaction loop-gain is k^2.

In this case, the feedback interaction model involves <u>improper</u> terms $S_{12}(s)$ and $S_{21}(s)$.

□

<u>Example 3</u>: <u>Postlethwaite</u> & <u>MacFarlane</u>

Example 5.6 of[16] uses the transfer function matrix (G(s) in their notation):

$$R(s) = \frac{1}{1.25(s+1)(s+2)} \begin{vmatrix} s-1 & s \\ -6 & s-2 \end{vmatrix} \qquad (21)$$

After some manipulation, the feedback interaction form is described by:

$$S_{11}(s) = \frac{1}{1.25(s-2)} \qquad (22)$$

$$S_{12}(s) = 1.25s$$

$$S_{21}(s) = -7.5$$

$$S_{22}(s) = \frac{1}{1.25(s-1)}$$

This example illustrates a system in feedforward interaction form with stable diagonal elements having zeros in the right half-plane. However, in feedback interaction form, the diagonal elements are unstable with no right half-plane zeros. A small perturbation to the $\underline{R}_{12}(s)$ term would, however, give $S_{11}(s)$ and $S_{22}(s)$ with right half-plane zeros. More detailed analysis of the underlying physical system would be required to determine whether such a perturbation was physically possible.

□

Relative gain array

One measure of interaction found in the process control literature is the relative gain array of Bristol[10,17,18,19]. This provides an interesting relation between the feedback and feedforward interaction forms. For a two-input two-output system, the relative gain array is:

$$\begin{vmatrix} \lambda & 1-\lambda \\ 1-\lambda & \lambda \end{vmatrix} \tag{23}$$

Where

$$\lambda = \frac{\dfrac{y_1}{u_1} \text{ with } u_2 \text{ constant}}{\dfrac{y_1}{u_1} \text{ with } y_2 \text{ constant}} \tag{24}$$

Using the feedback interaction model of eqn. 10.2.3, and the feedforward interaction model of equation 10.2.6, it follows that:

$$\lambda = \frac{R_{11}(0)}{S_{11}(0)} = \frac{1}{1 - L_I(0)} \tag{25}$$

Example

The output coupled tank has a relative gain array with

$$\lambda = \frac{a^2}{a^2 - k^2} \tag{26}$$

and the output coupled tanks have

$$\lambda = \frac{1}{1 - k^2}. \tag{27}$$

□

10.3. A SELF-TUNING ALGORITHM

In previous chapters, a continuous-time self-tuning controller arose via the following three design methods:

1. A method based on the notional feedback loop (Chapter 3) with a possibly unrealisable element in the feedback loop to cancel out undesirable system characteristics.

2. An emulator-based design method which replaces the unrealisable feedback element in 1. by the corresponding emulator (chapter 3).

3. A self-tuning design method based on 2. which attempts to reduce sensitivity to modelling error by replacing the emulator in 2. by a self-tuning emulator (chapter 6).

In this chapter, the additional details required to apply such methods to a two-loop system are discussed.

In the single-loop design, the basic requirement was that the notional design method gave a stable closed-loop system; this required that significant system zeros were in the left half-plane. In the two-loop case, it will be seen that the feedback interaction model provides a straightforward basis for the notional feedback loop design. In particular, the zeros of the two transfer functions $S_{11}(s)$ and $S_{22}(s)$ will be found to be important. As example 3 (section 10.2) shows, these zeros may be quite different from those of the feedforward transfer functions $\underline{R}_{11}(s)$ and $\underline{R}_{22}(s)$. To emphasise the importance of $S_{11}(s)$ and $S_{22}(s)$, an analogy with the single-input single-output case is made by defining the polynomials $A_1(s)$ and $B_1(s)$ by:

$$\frac{B_1(s)}{A_1(s)} = S_{11}(s) \; \star \tag{1}$$

<u>Notional design</u>

The single-loop notional feedback loop design (chapter 7) is applied directly to each loop ignoring the interaction. Thus for loop 1:

$$\bar{u}_1(s) = \frac{Z_1(s)}{Q_1(s)}\left[\frac{1}{Z_1(s)}\bar{w}_1(s) - \bar{\phi}_1(s)\right] \tag{2}$$

where

$$\bar{\phi}_1(s) = \frac{P_1(s)}{Z_1(s)}\bar{y}_1(s) \tag{3}$$

Note that the polynomial $Z_1(s)$ plays no role at this stage; it is merely included to provide compatibility with later sections. As in chapter 3 (3.11 in particular), the

---

⋆ Here and hereafter, repetition of similar equations is avoided by writing only the equation for the first loop. Equations for the second loop are found by changing subscript "1" to "2" and vice versa.

following design rules are used:

D1. degree $(P_1(s))$ = degree $(A_1(s))$ - degree $(B_1(s))$

D2. $Z_1(s) = P_1(\varepsilon s)$

D3. $P_1(0) = 1$.

$Q_1(s)$ is a proper stable 'compensator' with proper stable inverse, $\bar{w}_1(s)$ is the desired value of the output of loop one and $P_1(s)$ is the desired closed-loop system.

The closed-loop output of loop 1 is

$$\bar{y}_1(s) = S^c_{11}(s)[\frac{1}{Z_1(s)}\bar{w}_1(s) + \bar{e}^Q(s)] \tag{4}$$

where the <u>closed</u>-<u>loop</u> transfer function $S^c_{11}(s)$ is given by:

$$S^c_{11}(s) = \frac{L_1(s)}{1+L_1(s)}\frac{Z_1(s)}{P_1(s)} \tag{5}$$

where

$$L_1(s) = S_{11}(s)\frac{P_1(s)}{Q_1(s)} \tag{6}$$

and the <u>detuning error</u> $\bar{e}^Q(s)$ by:

$$\bar{e}^Q(s) = \frac{Q_1(s)}{Z_1(s)}S_{12}\bar{y}_2(s) \tag{7}$$

The first assumption is that the two loops are stable when the interaction is zero. As in the earlier chapters, it is further assumed that the systems have sufficient stability margin for the exponentially multiplied systems to be stable.

A1. $S^c_{11}(s-\alpha)$ and $S^c_{22}(s-\alpha)$ have no right half-plane poles.

$Q_1(s)=0$ corresponds to model-reference control; in this case the usual model-reference condition that $B_1(s)$ be stable would replace condition A1. A1 is thus a less stringent condition for the suggested detuned control where $Q_1(s)\neq 0$.

The two closed loops are displayed in Figure 10.3.1.

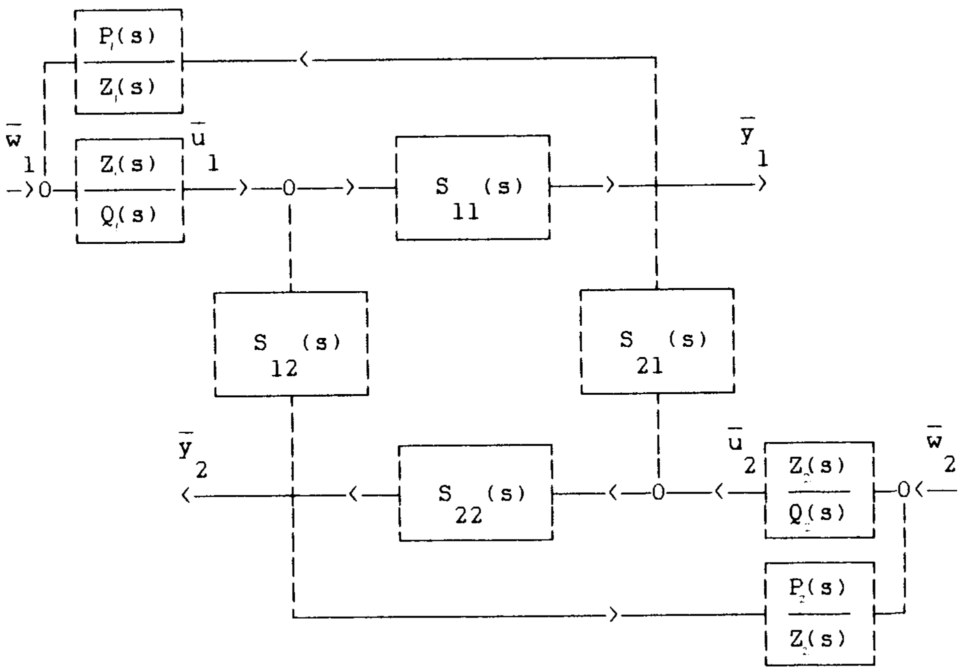


Figure 10.3.1 The notional feedback loop

If $Q_1(s)$ is small, and the resulting system is stable, the loops are approximately decoupled and:

$$\bar{y}_1(s) \simeq \frac{1}{P_1(s)} \bar{w}_1(s) \qquad (8)$$

Also, it is assumed that the coupling terms $S_{12}(s)$ and $S_{21}(s)$ have no right half-plane poles, and in addition:

A2. $S_{12}(s-\alpha)$ and $S_{21}(s-\alpha)$ have no right half-plane poles.

Using Nyquist's theorem, this two-loop notional system will be stable if the Nyquist locus of:

$$- S^c_{11}(s).S^c_{22}(s).\frac{Q_1(s)}{Z_1(s)}S_{12}(s).\frac{Q_2(s)}{Z_2(s)}S_{21}(s) \tag{9}$$

does not encircle the -1 point. If this underlying notional feedback loop is unstable, the adaptive system cannot be stable, so it is assumed that this notional feedback loop is stable:

A3. The two-loop notional feedback loop system is stable.

Example: Coupled tanks

For both input and output coupled tanks:

$$S_{11}(s) = S_{22}(s) = \frac{b}{s+a} \tag{10}$$

where b=1 for output coupled tanks and $b=1-k^2$ for input coupled tanks. If the design parameters P and Q are chosen as:

$$P(s) = 1+ps;\ Z(s) = Z^-(s) = 1+zs;\ Q(s) = qs \tag{11}$$

then the loop gains are given by:

$$L_1(s) = L_2(s) = \frac{b}{s+a}\,\frac{(1+ps)}{qs} \tag{12}$$

The notional closed-loop transfer function $S^c_{11}(s)$ is then

$$S^c_{11}(s) = \frac{1+zs}{1 + (p+\frac{qa}{b})s + \frac{q}{b}s^2} \tag{13}$$

□

The emulator

As in the single-loop case (chapter 6), the self-tuning controller is based on a low-gain emulator-based version of the notional design of the previous section. In particular, the notional feedback loop transfer function $\frac{P_1(s)}{Z_1(s)}$ is replaced by an emulator-based version using state-variable filters. In the two-loop context, loop interaction must be accounted for in the emulator design; this section concentrates on this aspect of the emulator.

The first loop of the system may be rewritten as:

$$\bar{y}_1(s) = \frac{B_1(s)}{A_1(s)}[\bar{u}_1(s) + S_{12}(s)\bar{y}_2(s)] \qquad (14)$$

where

$$S_{11}(s) = \frac{B_1(s)}{A_1(s)} \qquad (15)$$

Following the analysis of chapter 2, and replacing u by $\bar{u}_1(s) + S_{12}(s)\bar{y}_2(s)$, an emulator may be written as:

$$\bar{\Phi}_1^{\star}(s) = \frac{F_1(s)}{C_1(s)}\bar{y}_1(s) + \frac{G_1(s)}{C_1(s)Z_1(s)}[\bar{u}_1(s) + S_{12}(s)\bar{y}_2(s)]^{\star} \qquad (16)$$

where

$$\frac{P_1(s)C_1(s)}{Z_1(s)A_1(s)} = \frac{E_1(s)}{Z_1(s)} + \frac{F_1(s)}{A_1(s)} \qquad (17)$$

$$\deg(E_1(s)) = \deg(Z_1(s))-1; \ \deg(F_1(s)) = \deg(A_1(s))-1 \qquad (18)$$

⋆ The subscript ₁ refers to the loop index, not to the emulator version.

For practical reasons, it may not be possible to implement this emulator: the order of the various transfer functions may make it too complex or $S_{12}(s)$ may be improper. So three alternative design rules are proposed:

D4a The emulator is fully implemented

D4b The emulator is implemented with an approximate decoupling term $B_{12}(s)/A_1(s)$:

$$\frac{B_1(s)}{A_1(s)}S_{12}(s) \cong \frac{B_{12}(s)}{A_1(s)} \tag{19}$$

D4c The emulator is implemented with $S_{12}(s)$ replaced by zero.

The latter cases give an approximate emulator which will be denoted by $\bar{\phi}_1^a(s)$. The approximation error is given by:

$$\bar{e}^a(s) = \frac{G_1(s)}{C_1(s)Z_1(s)}\tilde{S}_{12}(s)\bar{y}_2(s) \tag{20}$$

where

$$\tilde{S}_{12}(s) = \begin{cases} 0 & \text{for D4a} \\ S_{12}(s) - \dfrac{B_{12}(s)}{B_1(s)} & \text{for D4b} \\ S_{12}(s) & \text{for D4c} \end{cases} \tag{21}$$

In all cases, the control law is given by:

$$\bar{u}_1(s) = \frac{Z_1(s)}{Q_1(s)}\left[\frac{1}{Z_1(s)}\bar{w}_1(s) - \bar{\phi}_1^a(s)\right] \tag{22}$$

The closed-loop system then becomes

$$\bar{y}_1(s) = \frac{L_1(s)}{1+L_1(s)}\,\frac{Z_1(s)}{P_1(s)}\left[\frac{1}{Z_1(s)}\bar{w}_1(s) + \bar{e}_1^Q(s) + \bar{e}_1^a(s)\right] \tag{23}$$

Example: coupled tanks

Continuing the example of the previous section:

$$A = s+a;\ B = b;\ C(s) = 1 \tag{24}$$

Identity 17 becomes:

$$\frac{(1+ps)}{(s+a)(1+zs)} = \frac{e}{(1+zs)} + \frac{f}{(s+a)} \tag{25}$$

where:

$$E_1(s) = E_2(s) = e = \frac{p-d}{1-az};\ F_1(s) = F_2(s) = f = \frac{1-ap}{1-az} \tag{26}$$

Thus, ignoring the coupling terms,

$$\bar{\phi}_1^a(s) = \frac{be}{1+zs}\bar{u}_1(s) + f\bar{y}_1(s) \tag{27}$$

$$\bar{\phi}_2^a(s) = \frac{be}{1+zs}\bar{u}_2(s) + f\bar{y}_2(s) \tag{28}$$

□

The adaptive controller

A continuous-time detuned model-reference self-tuning controller was considered in an earlier chapter. The simplest approach is to use design rule D4c and thus ignore coupling. The two-loop self-tuning controller is then merely two single-loop self-tuning controllers, one for each loop. Although this approach is simple, the presence of coupling terms not accounted for in the algorithm leads to possibly poor performance and even instability. This will be analysed in section 10.6.

The self-tuning method considered here relies on a linear-in-the-parameters representation of the emulator equation. In general, the emulator equation cannot be easily put into such a form due to the unknown denominators of $S_{12}(s)$ and $S_{21}(s)$, so design rule D4a cannot be directly used in the adaptive context. This section concentrates on design rule D4b.

Recalling the approximation in D4b, define the polynomial

$$G_{12}(s) = B_{12}(s)E_1(s) \tag{29}$$

The approximate emulator equation becomes:

$$\bar{\phi}_1^a(s) = \frac{F_1(s)}{C_1(s)}\bar{y}_1(s) + \frac{G_1(s)}{C_1(s)Z_1(s)}\bar{u}_1(s) + \frac{G_{12}(s)}{C_1(s)Z_1(s)}\bar{y}_2(s) \tag{30}$$

This may be rewritten as:

$$\bar{\phi}_1^a(s) = \underline{\bar{X}}^T(s)\underline{\theta} \tag{31}$$

where:

$$\underline{\bar{X}}^T(s) = \frac{1}{C_1(s)}[\bar{u}_1(s),\ s\bar{u}_1(s),\ \ldots;\ \bar{y}_1(s),\ s\bar{y}_1(s),\ \ldots; \tag{32}$$

$$\bar{y}_2(s),\ s\bar{y}_2(s),\ \ldots]$$

and

$$\underline{\theta}^T = [g_0,\ g_1,\ \ldots;\ f_0,\ f_1,\ \ldots;\ g'_0,\ g'_1,\ \ldots] \tag{33}$$

where g_i is the ith coefficient of $G_1(s)$, f_i is the ith coefficient of $F_1(s)$ and g'_i is the ith coefficient of $G_{12}(s)$.

As in the single-input single-output case, $\phi^{\star}$ is replaced by the output of the emulator with estimated parameters:

$$\hat{\phi} = \bar{\underline{X}}^{T}(s)\hat{\underline{\theta}} \tag{34}$$

The algorithm for generating $\hat{\underline{\theta}}$ is identical to that given in chapter 5.

The resultant error:

$$\hat{e}(s) = \bar{\phi}^{\star}(s) - \hat{\phi} \tag{35}$$

leads to the closed-loop system of equation 10.3.23 but with $\hat{e}$ replacing $\bar{e}^{a}(s)$:

$$\bar{y}_1(s) = S^{c}{}_{11}(s)\,[\frac{1}{Z_1(s)}\bar{w}_1(s) + \bar{e}^{Q}_1(s) + \hat{e}_1] \tag{36}$$

As in the single-input single-output case, the exponentially multiplied estimation error $e^{\alpha t}\hat{e}(s)$ can be considered to be the output of a system Ω_1 with input $e^{\alpha t}\bar{e}^{a}(s)$.

10.4. ERROR EQUATIONS

In this section, the various equations describing the error equations resulting from the three design methods (notional, emulator-based, and self-tuning) are gathered together. These equations appear in Figure 10.4.1.

The output of the first loop may be written as:

$$\bar{y}_1(s) = S^{c}{}_{11}(s)\,\frac{1}{Z_1(s)}\,\bar{w}_1(s) + \tilde{y} \tag{1}$$

The first term represents the system output with no error

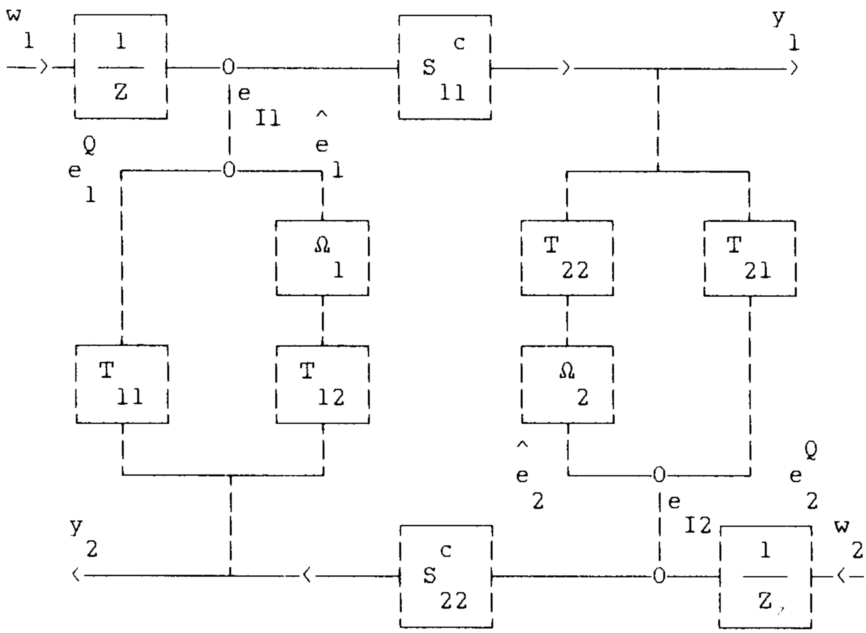


Figure 10.4.1 The error feedback system

due to interaction or estimation; the second term will be called the output error and is given by:

$$\tilde{y}_1 = S^c_{11}(s)\bar{e}^I_1(s) \quad (2)$$

where the interaction error is the sum of the detuning error and estimation error:

$$\bar{e}^I_1(s) = \bar{e}^Q(s) + \hat{e}(s) \quad (3)$$

The aim is to find stability conditions such that $\tilde{y}_1$ (and $\tilde{y}_2$) are small relative to the setpoints w_1 and w_2.

The detuning error, representing the effect of control weighting, is

$$\bar{e}^{Q}(s) = T_{11}(s)\bar{y}_{2}(s) \tag{4}$$

where

$$T_{11}(s) = \frac{Q_1(s)}{Z_1(s)}S_{12}(s) \tag{5}$$

The expression for $\hat{e}(s)$ depends on which of the three design methods is used. The three expressions are combined into one by defining:

$$\bar{\Omega}_1 = \begin{cases} 0 \text{ for the notional design} \\ 1 \text{ for the emulator-based design} \\ \Omega_1 \text{ for the self-tuning design} \end{cases} \tag{6}$$

Thus

$$\hat{e}_1 = \bar{\Omega}_1 \bar{e}^{a}_1(s) \tag{7}$$

and

$$\bar{e}^{a}_1(s) = T_{12}(s)\bar{y}_2(s) \tag{8}$$

where

$$T_{12}(s) = \frac{E_1(s)B_1(s)}{C_1(s)Z_1(s)}\tilde{S}_{12}(s) \tag{9}$$

The equations of this section appear in block-diagram form in Figure 10.4.1.

10.5. NON-ADAPTIVE ROBUSTNESS

The stability of the notional feedback loop design method has already been considered in section 3; so this section concentrates on the other non-adaptive method: that based on emulator-based control. Thus here:

$$\bar{\Omega}_1 = 1 \tag{1}$$

Hence the relation between the interaction error $\bar{e}_1^I(s)$ and the system output $\bar{y}_2(s)$ is given by:

$$\bar{e}_1^I(s) = \bar{e}^a(s)_1 + \bar{e}^Q(s) = [T_{11}(s) + T_{12}(s)]\bar{y}_2(s) \tag{2}$$

The equations of section 10.4, and Figure 10.4.1, reveal that there is a single feedback loop describing the two-loop system. This may be analysed using Nyquist's theorem as follows:

<u>Theorem 10.1</u> (<u>non-adaptive robustness</u>)

The two-loop system (eqns. 10.2.1&2) controlled using the non-adaptive controller is stable if assumptions A1 and A2 hold and if the shifted Nyquist locus of:

$$S^c_{11}(s)S^c_{12}(s)S^c_{21}(s)S^c_{22}(s) \text{ where } s = -\alpha + j\omega \tag{3}$$

where

$$S^c_{12} = T_{11}(s) + T_{12}(s) \tag{4}$$

does not encircle the -1 point.

<u>Proof</u>

From assumptions A1 and A2, and the fact that Q(s) and Z(s) are stable polynomials, all transfer functions in eqn. 10.5.3 are stable. The result then follows from Nyquists theorem. The use of $\alpha>0$ gives a certain stability margin and is included here for comparison with the results of the next section.

Remark

In fact, as discussed in chapter 4, it is not necessary (in the case of non-adaptive control) that assumptions A1 and A2 are true. But in such circumstances, the more general version of Nyquist's criterion must be used.

□

10.6. ADAPTIVE ROBUSTNESS

In the self-tuning case, the same set of equations as in the non-adaptive case describes the evolution of the error, except that:

$$\bar{\Omega}_1 = \Omega_1 \quad (1)$$

As Ω_1 is not a linear time-invariant system, Nyquist's theorem cannot be used. However, from chapter 5, Ω_1 has a gain in the L_2 sense of unity. Hence, the small gain theorem[20] may be applied. But first, the error equations must be written in a suitable form. Unlike the non-adaptive case, the presence of Ω_1 means that the parallel transfer functions cannot be amalgamated into one transfer function $S^C_{12}(s)$. Instead, the error $\bar{e}^I_1(s)$ is rewritten as follows:

$$\bar{e}^I_1(s) = \Omega_1 \bar{e}^a_1(s) + \bar{e}^Q_1(s) \quad (2)$$

where:

$$e_{Q_1}(s) = T_{11}(s)S^c{}_{22}(s)[\bar{e}^I_2(s) + \bar{w}_2(s)] \quad (3)$$

and

$$\bar{e}^a_1(s) = T_{12}(s)S^c{}_{22}(s)[\bar{e}^I_2(s) + \bar{w}_2(s)] \quad (4)$$

Similar expressions give $\bar{e}^I_2(s)$ in terms of $\bar{e}^I_1(s)$ and $\bar{w}_1(s)$. This feedback system appears in Figure 10.4.1.

As in the earlier chapter, the first step is to show that the exponentially multiplied feedback system with inputs $\bar{w}_1(s)$ and $\bar{w}_2(s)$ is L_2 stable. As $\tilde{y}_1$ is related to $\bar{e}^I_2(s)$ by a low-pass transfer function, L_∞ stability is shown for the system with $\bar{w}_1(s)$ and $\bar{w}_2(s)$ as inputs and $\tilde{y}_1$ and $\tilde{y}_2$ as outputs.

These ideas lead to the following theorem:

Theorem 10.2 (Adaptive robustness)

If the adaptive controller is designed according to design rules D1-D3 and D4b, assumptions A1-A3 are true and:

1. The forgetting factor of the self-tuning algorithm is positive: $\beta>0$

2. For some $\alpha>0$:

$$(\gamma_{11}(\alpha) + \gamma_{12}(\alpha)).(\gamma_{22}(\alpha) + \gamma_{21}(\alpha)) < 1 \quad (5)$$

where:

$$\gamma_{11}(\alpha) = \sup_\omega |T_{11}(\omega - \alpha)S^c{}_{22}(\omega - \alpha)| \quad (6)$$

and

$$\gamma_{12}(\alpha) = \sup_{\omega}|T_{12}(\omega - \alpha)S^{c}_{22}(\omega - \alpha)| \tag{7}$$

then the resultant closed-loop system is stable in the same sense as described in chapter 7.

Proof

Firstly, each block in Figure 10.4.1 is premultiplied by $e^{-\alpha t}$ and postmultiplied by $e^{\alpha t}$. As the gain of $\Omega_1 < 1$, then the gain of $\Omega_1 T_{12}(s)S^{c}_{22}(s)$ < the gain of $T_{12}(s)S^{c}_{22}(s)$. As $T_{11}(s)S^{c}_{22}(s)$ and $T_{12}(s)S^{c}_{22}(s)$ are linear transfer functions, then their gain (in the L_2 sense) is given by the expressions for γ_{11} and γ_{12}. The same statement holds with 1 and 2 interchanged. As in chapter 7, the L_2 stability of the exponentially multiplied system follows from the small gain theorem[20].

Using the results of chapter 7, the fact that the exponentially multiplied system with inputs $\bar{w}_1(s)$ and $\bar{w}_2(s)$ and outputs $\bar{e}^{I}_1(s)$ and e_{I2} is L_2 stable, together with the fact that $\tilde{y}_1$ and $\tilde{y}_2$ are related to $\bar{e}^{I}_2(s)$ and e_{I2} via low-pass transfer functions, give the required result.

□

To illustrate these results, the transfer functions T_{ij} are derived for the two coupled tank examples.

Example: Output coupled tanks

Using the same control parameters as in section 3, it follows that in the case of output coupled tanks b=1 and

$$T_{11}(s)S^{c}_{22}(s) = T_{22}(s)S^{c}_{11}(s) = \frac{Q_1(s)}{Z_1(s)}S_{12}(s)S^{c}_{22}(s) \tag{8}$$

$$= \frac{kqs}{1 + (p+qa)s + qs^2}$$

If the coupling term is estimated (as it can be here), then

$$T_{12}(s)S^C_{22}(s) = T_{21}(s) = 0 \tag{9}$$

On the other hand, if no attempt is made to identify the coupling term, then $\tilde{S}_{12}(s) = S_{12}(s)$ and:

$$T_{12}(s)S^C_{22}(s) = T_{21}(s)S^C_{11}(s) \tag{10}$$

$$= \frac{E_1(s)B_1(s)}{C_1(s)Z_1(s)}\tilde{S}_{12}(s)S^C_{22}(s)$$

$$= \frac{ek}{1 + (p+qa)s + qs^2}$$

□

Example: Input coupled tanks

Using the same control parameters as in section 3, it follows that in the case of input coupled tanks $b=1-k^2$ and

$$T_{11}(s)S^C_{22}(s) = T_{22}(s)S^C_{11}(s) \tag{11}$$

$$= \frac{Q_1(s)}{Z_1(s)}S_{12}(s)S^C_{22}(s)$$

$$= \frac{kqs(s+a)}{1 + (p(1-k^2) + qa)s + qs^2}$$

The coupling term cannot be estimated so that $\tilde{S}_{12}(s) = S_{12}(s)$ and:

$$T_{12}(s)S^{c}_{22}(s) = T_{21}(s)S^{c}_{11}(s) \tag{12}$$

$$= \frac{E_1(s)B_1(s)}{C_1(s)Z_1(s)}\tilde{S}_{12}(s)S^{c}_{22}(s)$$

$$= \frac{ek(s+a)}{1 + [p + qa/(1-k^2)]s + [q/(1+k^2)]s^2}$$

The robustness conditions are harder to satisfy for the input coupled tanks, as the improper interaction terms ($S_{12}(s)$ and $S_{21}(s)$) lead to $T_{11}(s)S^{c}_{22}(s)$ and $T_{22}(s)S^{c}_{11}(s)$ having non-zero gain (k) at high frequencies, and this gain is independent of the weighting factor q if $q \neq 0$.

□

10.7. SUMMARY

Using a particular representation for two-input two-output systems, standard input-output methods have been used to derive frequency-domain conditions to ensure that a continuous-time least-squares based self-tuning algorithm is stable in the face of unmodelled interaction dynamics. Because of the particular structure chosen, the stability analysis is based on a single-loop feedback system. As in the single-input single-output case, both adaptive and non-adaptive stabilities are based on the frequency-domain properties of certain transfer functions.

The n-input n-output case is discussed elsewhere[7]. However, as the error equations no longer form a single-loop feedback system, this results in a more complex criterion. The two-input, two-output system considered in this chapter is thus an important special case which deserves separate analysis.

References

1. Borisson, U., "Self-tuning regulator for a class of multivariable systems," Automatica, vol. 15, pp. 209-215., 1979.

2. Koivo, H.N., "A multivariable self-tuning controller," Automatica, vol. 16, pp. 351-366, 1980.

3. Morris, A.J., Nazer, Y., and Wood, R.K., "Multivariate self-tuning process control," Optimal control: applications and methods, vol. 3, no. 4, pp. 363-388, 1982.

4. Peel, D., Morris, A.J., and Tham, M.T., Univariate self-tuning control in a distributed control environment, 2, pp. 563-568, IEE (Conference publication 232), International conference 'Control 85', Cambridge, 1985.

5. Gawthrop, P.J., "Self-tuning PI and PID Controllers," in Proceedings of the IEEE conference on "Applications of Adaptive and Multivariable Control", Hull, 1982.

6. Gawthrop, P.J., "Self-tuning PID controllers: Algorithms and implementation," IEEE Transactions on Automatic Control., vol. AC-31, no. 3, 1986.

7. Gawthrop, P.J., "Robust self-tuning control of n-input n-output systems," in Preprints of the 7th IFAC Symposium on Identification and System Parameter Estimation, York, U.K., 1985.

8. Vidyasagar, M., Input-output analysis of large-scale interconnected systems, Springer, Berlin, 1981.

9. Huseyin, O, Sezer, M.E., and Siljak, D.D., "Robust decentralised control using output feedback," IEE Proceedings, vol. 129 Pt.D., no. 6, pp. 310-314, 1982.

10. Bristol, E.H., "On a new measure of interaction for multivariable process control," IEEE Transactions, vol. AC-10., 1966.

11. Bhalodia, M. and Weber, T.W., "Feedback control of a two-input, two-output interacting process," Ind. Eng. Chem. Process Des. Dev., vol. 18, pp. 599-607, 1979.

12. Waller, K.V.T., "Decoupling in distillation," AIChE J., vol. 20, pp. 592-594, 1974.

13. Yang, C-H. and Ward T.J., "Decoupling control," AIChE J., vol. 20, pp. 1215-1217, 1974.

14. Gawthrop, P.J., "Robustness of self-tuning controllers. PartII: Two-input two-output systems.," Report CE/T/12, School of Engineering and Applied Sciences, Univ. of Sussex., 1985.

15. Owens, D.H., Multivariable and optimal systems, Academic, 1981.

16. Postlethwaite, I. and MacFarlane, A.G.J., A complex variable approach to the analysis of linear multivariable systems, Springer, Berlin, 1979.

17. Shinskey, F.G., Process control systems, McGraw-Hill, 1979.

18. Deshpande, P.B. and Ashe, R.H., Elements of computer process control with advanced control applications, Instrument Society of America, 1981.

19. Wang, S. and Munro, N., "A complete proof of Bristol's relative gain array," Trans. Inst. M C, vol. 4, no. 1, pp. 53-56, 1982.

20. Desoer, C.A. and Vidyasagar, M., Feedback systems : Input-output properties, Academic Press, London, 1975.

Keyword Index

Symbol Index