A UNIFIED SOLUTION TO CONSTRAINED CONFIGURATION CONTROL LAW DESIGN

by

Andrew F. Potvin

Submitted to the

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

in partial fulfillment of the requirements

for the degree of

BACHELOR OF SCIENCE

and

MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

October, 1991

Signature of Author ______________________________

Department of Electrical Engineering and Computer Science,
September, 1991

Certified by ______________________________

Associate Professor Munther A. Dahleh
Thesis Supervisor

Certified by ______________________________

K. Dean Minto
General Electric Supervisor

Accepted by ______________________________

Campbell L. Searle, Chairman
Committee on Graduate Students

A Unified Solution to Constrained Configuration Control Law Design

by

Andrew F. Potvin

Submitted to the Department of Electrical Engineering

in October, 1991 in partial fulfillment of the

requirements for the degrees of

Bachelor of Science and Master of Science in

Electrical Engineering

ABSTRACT

This thesis implements a previously proposed algorithm for constrained configuration control law design (cccld). The algorithm assumes an H_2 or mixed H_2/H_∞ cost formulation. After reconfiguring the controller as a linear fractional transformation (LFT) interconnecting a fixed dynamic system and a tunable static system, the design is recast as a (possibly block decentralized) static output feedback (SOF) problem. Necessary equations for optimal SOF control are known, but satisfying the equations presently requires iterative numerical algorithms. Finally one recovers the constrained controller as a function of the optimal static gain.

This thesis considers the entire cccld method summarized above including (1) problem setup and formulation, (2) data reduction to SOF, (3) solution of SOF, and (4) practical applications. Previously underdeveloped steps of the algorithm are resolved, and the entire method is coded into MATLAB m-files. Integrating the weighting techniques, LFT decomposition, and SOF software this thesis discusses an application of the cccld procedure to a gas turbine engine.

Thesis supervisor: Munther A. Dahleh

Title: Associate Professor of Electrical Engineering

ACKNOWLEDGEMENT

I wish first of all to thank those who had the greatest technical impact on this thesis: my advisors Dr. K. Dean Minto and Prof. Munther A. Dahleh. Dean provided me with day–to–day direction, advise, consultation, latitude, ..., during my research time at GE CR&D. Munther provided me with valuable alternative perspectives, always with an eye on the "bigger picture."

I would also like to thank the staff of the Control Systems Laboratory at CR&D for making my stay enjoyable, especially Dr. Jim Wilson (for introducing me to CR&D happy hours), the Dead Brain Society, and the noontime runners. I greatly appreciate Dr. Tim Johnson for arranging my internship at CR&D and Gordon Fiddes for the opportunity to participate in the VI-A program. Thanks Karl and Mike for letting me crash at your place during January.

A special thanks go to my parents who have always encouraged me in all my endeavors. I don't tell you often enough how much your loving support means to me.

Contents

List of Figures

Chapter 1

Introduction

This thesis contributes to the development of a unified software solution to a class of constrained configuration control law design (cccld) problems. The solution is based on a recent pair of articles by Bernstein, Haddad, and Nett [49, 5] which propose and partially develop a methodology for synthesizing optimal controllers for multivariable LTI systems while incorporating arbitrary constraints on controller structure. The methodology assumes a standard output feedback configuration as shown below. Controller order is assumed given, while controller parameters may either be tunable for optimization or fixed apriori. In this thesis, optimality is determined by minimizing a closed loop H_2 cost subject to an H_∞ bound.

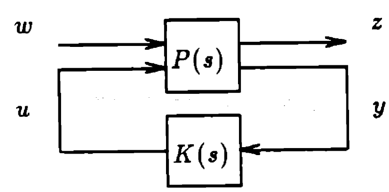


Figure 1.1: Standard output feedback configuration.

The standard output feedback configuration of Figure 1.1 is also known as a linear fractional transformation (LFT) of K in P [53]and is sometimes written $F_\ell(P, K)$ [13, Ch.1.4.1]. In this setup, w and z represent vectors of external inputs and outputs while u and y represent vectors of controlled and sensed variables. The algebra of manipulating two (block) input, two (block) output (TITO) systems such as P is well understood for both frequency and state space representations [49, 42, 52]; often used TITO operations have been mechanized in the MATLAB[1] toolbox ISICLE© [42].

The constrained configuration control law design solution implemented here begins by incorporating weighting functions into the control loop, and then posing an optimal H_2 control problem (or an optimal H_2 problem subject to a maximum H_∞ cost) which is presented in a standard output feedback configuration. After reconfiguring the controller as a linear fractional transformation

[1] MATLAB is a trademark of The MathWorks, Inc.

(LFT) interconnecting a fixed dynamic system and a tunable (possibly block decentralized) static system, the design is recast as a static output feedback (SOF) problem. Necessary equations for optimal SOF control are known, but satisfying the equations presently requires iterative numerical algorithms. Finally one recovers the constrained controller as a function of optimal static gain.

Remark 1 *Neither the formulation, reconfiguration, or solution changes the difficulty of the cccld problem: as will be discussed later, the problem remains non-convex. Indeed even the question of existence of admissable (stabilizing) controllers for the cccld problem remains unsolved.*

This thesis considers the entire cccld method summarized above including (1) problem setup and formulation, (2) data reduction to SOF, (3) solution of SOF, and (4) practical applications. Given a command tracking configuration and constrained controller, this thesis proposes a weighting scheme, similar to LQG weighting, which provides intuition and mechanisms for loop shaping. This thesis provides an LFT decomposition for a large class of constrained configuration controllers. It explores a number of practical considerations related to the feedback descent algorithm, a numerical technique used to satisfy the SOF optimality equations. Integrating the weighting techniques, LFT decomposition, and SOF software this thesis discusses an application of the cccld procedure to a gas turbine engine.

The remainder of this chapter is organized as follows. First §1.1 reviews some unconstrained control solutions including the familiar LQG result and the recent H_∞ result of Doyle, Glover, Khargonekar, and Francis [14]. Next §1.2 presents the constrained configuration control law design problem in more detail. It motivates the problem primarily as a necessary alternative to generating unconstrained controllers. Finally §1.3 outlines the remaining chapters of this thesis.

1.1 Unconstrained full order control solutions

Before presenting the cccld problem in §1.2, this section summarizes some unconstrained optimal control solutions, beginning with the familiar LQG result originating from [27, 28]. Specifically, if a system $G(s)$ is given by

$$\dot{x} = Ax + Bu + w_1 \tag{1.1}$$

$$z = E_1 x + E_2 u \tag{1.2}$$

$$y = Cx + w_2 \tag{1.3}$$

$$V_i = \mathcal{E}(w_i w_i'),\ i = 1, 2 \tag{1.4}$$

where

$$R_1 = E_1' E_1 \qquad R_2 = E_2' E_2 \qquad E_1' E_2 = 0 \tag{1.5}$$

consider designing a compensator, $K(s)$ operating from y to u to minimize the cost

$$\begin{aligned} J &= \lim_{t\to\infty} \mathcal{E}(z'z) \\ &= \lim_{t\to\infty} \mathcal{E}(x'R_1x + u'R_2u) \end{aligned}$$

where $\mathcal{E}$ represents expected value. It is well known that the optimal solution to this problem (unique to within a similarity transform) is a linear compensator of the same order as G

$$\dot{x}_c = A_c x_c + B_c y \tag{1.6}$$

$$u = C_c x_c \tag{1.7}$$

where the compensator's state space matrices are given by

$$A_c = A - BR_2^{-1}B'P - QC'V_2^{-1}C \tag{1.8}$$

$$B_c = QC'V_2^{-1} \tag{1.9}$$

$$C_c = -R_2^{-1}B'P \tag{1.10}$$

and where P and Q are solutions to the Ricatti equations

$$0 = A'P + PA + R_1 - PBR_2^{-1}B'P \tag{1.11}$$

$$0 = AQ + QA' + V_1 - QC'V_2^{-1}CQ. \tag{1.12}$$

Note that the Ricatti equations are decoupled: (1.11) is known as the (linear quadratic) regulator equation and (1.12) is known as the (Kalman) filter equation. In the full state feedback case where $C = I$, the optimal compensator collapses to merely a static gain

$$u = C_c y \tag{1.13}$$

where C_c is calculated as in (1.10) which requires only solving the regulator equation. In summary, the LQG control synthesis method has the following advantages: it always yields a stable closed loop system (under stabilizability and detectability assumptions); solutions may be solved for in closed form; and closed loop performance objectives may be stated in terms of desired loop shapes.

Because the optimal compensator for this problem possesses dynamic order equal to that of the original system, it is referred to as a full order (unconstrained) compensator. It is also well known that assuming the original system has n states and p outputs, one may design a controller (a Luenberger observer) of order $n - p$ which asymptotically reproduces the dynamics of the original system. For this reason, controllers of order less than $n-p$ are referred to as low order. [34, pp.300-9] [33]

In recent notes, Dailey [12] shows the full order quadratic cost formulation and optimal solution to have an equivalent H_2 minimization interpretation. Namely the compensator given by (1.6)-(1.12) minimizes the 2-norm of the closed loop transfer function from u to y (*i.e.* it satisfies $\min_K \| T_{uy} \|_2$). Dailey also summarizes full order solutions to more general problems than those reviewed in this section.

In their seminal paper, Doyle, Glover, Khargonekar, and Francis [14] showed the above LQG results to be just the asymptotic case of the full order (unconstrained) H_∞ control solution. In other words, given the system G as above consider designing a compensator to satisfy

$$\| T_{wz} \|_\infty \leq \gamma$$

where T_{wz} is the closed loop transfer function from w to z and $\gamma > 0$ is a real scalar. Assuming such a controller exists, it will have the form (1.6)-(1.7) where the compensator's state space matrices are given by

$$A_c = A - BR_2^{-1}B'X_\infty(I - \gamma^{-2}Y_\infty X_\infty)^{-1} - Y_\infty C'V_2^{-1}C + \gamma^{-2}Y_\infty R_1 \tag{1.14}$$

$$B_c = (I - \gamma^{-2}Y_\infty X_\infty)^{-1}Y_\infty C'V_2^{-1} \tag{1.15}$$

$$C_c = -R_2^{-1}B'X_\infty \tag{1.16}$$

$$0 = A'X_\infty + X_\infty A + R_1 - X_\infty(BR_2^{-1}B' - \gamma^{-2}V_1)X_\infty \tag{1.17}$$

$$0 = AY_\infty + Y_\infty A' + V_1 - Y_\infty(C'V_2^{-1}C - \gamma^{-2}R_1)Y_\infty \tag{1.18}$$

and where one requires

$$X_\infty, Y_\infty \geq 0 \tag{1.19}$$

$$\rho(X_\infty Y_\infty) \leq \gamma^2 \tag{1.20}$$

where $\rho(\cdot)$ is the spectral radius. Note that the Ricatti equations are again decoupled: (1.17) is known as the (H_∞) regulator equation and (1.18) is known as the (H_∞) filter equation. Notice that the gain B_c include both Y_∞ and X_∞ terms: alternate realizations would introduce the coupling in either the gain C_c or the Ricatti equations. As in the H_2 case, if $C = I$ the solution collapses to (1.13) where C_c is calculated as in (1.16) which only requires solution to the regulator equation.

To generate an optimal H_∞ controller — that resulting in the lowest possible γ — one first searches to find the smallest γ which satisfies (1.17)-(1.20) and then solves (1.14)-(1.16). Given the problem formulation above, upper and lower bounds on γ may be computed [9]. Besides possessing similar advantages to the LQG method, H_∞ control provides robustness guarantees.

One advantage of the above problem formulations is that the optimal controllers possess closed form solutions. Such a characteristic contributes to the use of full order (unconstrained) optimal control results as an aid in control design. However control system designers do not always have the liberty of using full order designs as detailed in the next section.

1.2 CCCLD problem overview

Constraints on controller structure arise naturally in real world control design, especially as plants and plant models become increasingly complex. Cost, complexity, and reliability concerns force engineers to design controllers as simply as possible while still meeting performance requirements. Some popular constraints imposed on controllers include requiring them to be of multivariable PI type, to have certain fixed poles, or to be of fixed dynamic order.

Although rigorous theory and closed loop formulas exist for generating full order optimal output feedback controllers as detailed above, no closed form solutions have been found for generating optimal low order controllers (for a discussion, see [24, 4]). In the general low order unconstrained H_2 problem, Hyland and Bernstein [23, 24] show that the separation principle no longer applies. Instead the filter and regulator equations couple through an idempotent matrix. Since solving coupled Ricatti equations presently requires complex iterative numerical algorithms, other approaches have been developed for synthesizing optimal low order controllers. Some methods rely on model reduction. For example, one might attempt to reduce the order of an optimal compensator designed for the full order plant; similarily one might first reduce the order of the plant and then generate an optimal controller for the low order plant; or one might use a combination of plant and controller reduction. However, as shown by Hyland and Bernstein [23], none of these techniques results in an optimal low order controller for the full order plant; indeed, the control generated can even fail to guarantee stability of the full order plant. More recently, Mustafa has developed H_∞ balanced truncation theory which yields a sufficient condition for stability under reduced order control [47, 48, 45, 46]. Though promising, the theory does not yet allow weighting to specify performance objectives without also introducing added states to the controller: thus if one wishes to generate an r^{th} order controller for an n^{th} order plant weighted by n_w states, one must actually generate an $(r-n_w)^{th}$ order compensator for the weighted system (which is of order $n+n_w$). As explained below, the cccld method proposed by Bernstein, Haddad, and Nett (BHN) does not reflect weighting states back to the compensator. Another limitation of H_∞-balanced truncation theory is that it cannot accommodate constrained configuration control (only fixed order control).

As mentioned above, the cccld solution implemented here reformulates the problem in an optimal SOF framework. In all, the methodology involves six independent steps which are at various stages of development. The steps are:

1. Choose control configuration and controller constraints,
2. Formulate optimal H_2/H_∞ problem,
3. Extract controller parameters,
4. Formuate equivalent SOF problem,
5. Solve SOF problem, and
6. Recover controller.

In the first step, the designer examines the plant and performance specifications and then chooses a control framework and adopts controller constraints consistent with meeting performance objectives. One assumes the compensator is of fixed dynamic order, but that it may otherwise be arbitrarily constrained. In the second step, the designer introduces weighting functions and views the problem in the standard output feedback framework of Figure 1.1. This thesis develops a weighting scheme for command tracking control configurations with integral action which provides intuition and a mechanism for loop shaping.

A block diagram interpretation of the next two steps is depicted in Figure 1.2. Step three

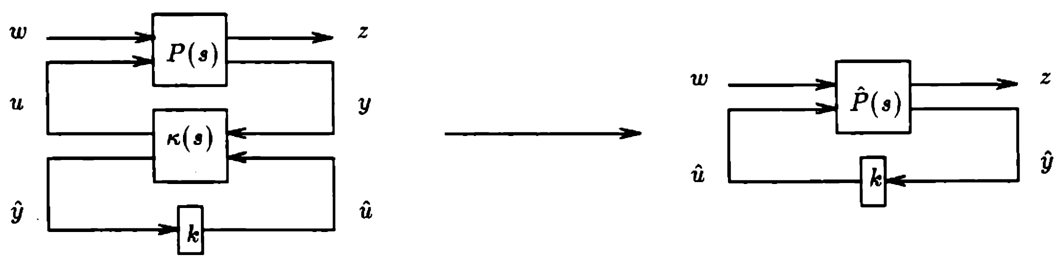


Figure 1.2: Static output feedback formulation.

decomposes the partially tunable compensator K as an LFT of a fixed dynamic system $\kappa(s)$ and a static system k tunable (most generally along its block diagonal) for optimization. Then step four combines the systems P and κ to yield a system $\hat{P}(s)$ thus transforming the dynamic control problem into a SOF problem. BHN prove the existence of such a decomposition if the compensator is at most an affine function of its unknown parameters. They also provide decompositions for a number of specific control structures. This thesis extends BHN's work in this area by developing a general LFT decomposition technique which may be automated in software.

The fifth step employs numerical techniques to compute solutions to the SOF problems by satisfying optimality conditions which have previously been derived. This thesis enhances a recently developed technique for solving the optimal design problem and discusses issues which affect the techniques' convergence characteristics. Finally, one recovers the optimal constrained controller by recombining the fixed dynamic system with the (now optimized) parameter matrix.

Some methods of constrained configuration control law design use the same problem formulation as above but a different solution approach. One particularly relevent work is that of Mercadal [39] which approaches the design of decentralized controllers for large scale systems as a fixed architecture LQG problem (see §3.4.2 for more details regarding controller constraints). Instead of formulating an equivalent static output feedback problem, Mercadal uses techniques similar to other optimal projection work [3] and sets up a Lagrangian to derive first order necessary conditions which also satisfy the constraints on the controller's structure. The constraints enter the equations as projections of the full order LQG solution (*i.e.* the equations are coupled through an idempotent projection matrix). Of course for a given problem, the associated SOF necesary equations and the optimal projection equations are merely two different ways to write the same equations. However, the optimal projection equations lend themselves to solution via homotopy techniques [54] and Mercadal does indeed develop a homotopy algorithm to solve the decentralized control problem. Homotopy methods have the advantage that initialization concerns (see §4.3) are eliminated because one begins iterating with the full order controller (for which closed form solutions exist), and in theory homotopy algorithms will converge to a global minimum should one exist. However, Mercadal reports that homotopy methods tend to be unreliable for optimal projection applications. In addition, he uses a different homotopy algorithm for fixed order (but otherwise unconstrained) control than for decentralized control, and would require yet another for true static output feedback problems. One advantage of formulating all problems in the SOF framework is that only one software solution is required (or possibly two — see Remark 2) and one instead varies the weighting scheme depending on the application.

1.3 Thesis outline

Now that the problem of optimal constrained configuration control law design has been framed, §2 reviews the problem history and various solution formulations. As mentioned above, one finds that although no closed form solutions are known, necessary equations for optimality have been derived which are amenable to iterative search algorithms. The work of BHN is explained in more detail and a numerical algorithm for computing solutions which meet the SOF optimality conditions is summarized.

Towards the goal of creating a unified software solution to cccld, §3 presents a technique whereby a partially tunable dynamic system may be decomposed as an LFT of a fixed dynamic system and a tunable static system. Thus a dynamic design problem is recast as a static output feedback problem. Given the SOF framework, §4 discusses practical details related to a computer implementation of the algorithm in §2. Considerations include such things as guarantees of monotonically decreasing cost at each iteration and requirements for initialization of search.

With SOF formulation and numerical machinery well in hand, §5 applies the methodology to generate optimal H_2 and H_2/H_∞ PI controllers for a GE gas turbine engine. In an attempt to develop intuition and mechanisms for loop shaping in this design framework, §5 proposes a weighting scheme for cccld which is similar to full order LQG and H_∞ weighting techniques. Another PI controller for the same system, generated via the method of Edmunds [15], allows evaluation of the new methodology's strengths and weaknesses through comparing closed loop system performance, problem formulation complexity, solution computation time, and performance specification adjustment capability. Finally, §6 reiterates thesis contributions and results and provides recommendations for further research.

Working software is the only acceptable proof.

Chapter 2

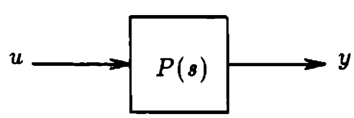
Theoretical Background

This chapter reviews notational and mechanical tools and previous theoretical results as a foundation for the chapters to follow. First §2.1 reviews state space notation and two–input two–output (TITO) operators. Next §2.2 summarizes early results in the area of optimal control by static or fixed-order output feedback. Then §2.3 contains more recent optimal control results that incorporate disturbances, an infinity cost constraint, and controller configuration constraints. One finds in §2.2 and §2.3 that although necessary equations for optimal low order and static output feedback are known, no closed form solutions to the equations presently exist. Thus §2.4 provides a summary of one algorithm developed to iteratively solve the equations of §2.2 and §2.3.

2.1 State space and TITO notation

We assume reader familiarity with basic single (block) input, single (block) output (SISO) notation where a linear system may be represented in the following ways:

Block diagram



Frequency domain

$$P(s) = C(sI - A)^{-1}B + D$$

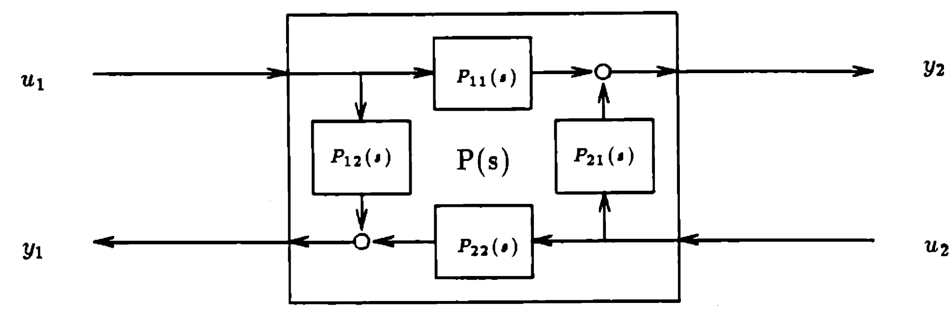
State space

$$\begin{aligned} P(s) \quad \longleftrightarrow \quad & \dot{x} = Ax + Bu \\ & y = Cx + Du \\ \longleftrightarrow \quad & \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \end{aligned} \tag{2.1}$$

The packed matrix construct is a notational convenience and always includes explicit state partitioning. The explicit partitioning acts to distinguish a <u>system</u> represented by a packed matrix construct from a gain represented by a partitioned matrix; the importance of understanding and representing the difference will become apparent in §3.

As in [49] this notation readily extends to two-block input two-block output (TITO) systems as follows:

Block diagram



Frequency domain

$$
\begin{aligned}
P(s) &= \begin{bmatrix} P_{11}(s) & \vdots & P_{12}(s) \\ \cdots & \cdot & \cdots \\ P_{21}(s) & \vdots & P_{22}(s) \end{bmatrix} \\
P_{kl}(s) &= C_k(sI - A)^{-1}B_l + D_{kl}
\end{aligned}
$$

State space

$$
P(s) \longleftrightarrow \begin{array}{rcllll} \dot{x} & = & Ax & +B_1u_1 & +B_2u_2 \\ y_1 & = & C_1x & +D_{11}u_1 & +D_{12}u_2 \\ y_2 & = & C_2x & +D_{21}u_1 & +D_{22}u_2 \end{array}
$$

$$
\longleftrightarrow \left[\begin{array}{c|ccc} A & B_1 & \vdots & B_2 \\ \hline C_1 & D_{11} & \vdots & D_{12} \\ \cdots & \cdots & \cdot & \cdots \\ C_2 & D_{21} & \vdots & D_{22} \end{array}\right] \tag{2.2}
$$

where the packed matrix construct is again a notational convenience. To avoid confusion, we explicitly show both the state and input/output partitioning of packed matrices and transfer functions.

Note the generality of this representation in that each of the inputs or outputs may be vectors of signals. One treats mixed representations or systems with less than two (block) inputs or two (block) outputs by assuming that the appropriate input or output is "empty," *i.e.* not present. First introduced in [43] and then redefined in [49], the empty matrix is defined algebraically as follows.

Definition 1 (Empty matrix operator) *The empty matrix operator is denoted by the symbol* [ ] *and is defined by the following properties:* $\forall M$

$$\begin{aligned} &[\,]\,M = M\,[\,] = [\,],\\ &M + [\,] = [\,] + M = M,\\ &[\,[\,]\,M] = [M\,[\,]\,] = \begin{bmatrix} M \\ [\,] \end{bmatrix} = \begin{bmatrix} [\,] \\ M \end{bmatrix} = M. \end{aligned} \tag{2.3}$$

The empty transfer function may be defined in exactly the same way if one assumes the dependence on s has been dropped for convenience. We use the symbol $[\,]$ as both the empty matrix and the empty transfer function with the meaning to be inferred from the context.

Before reviewing how TITO interconnections are defined, one requires definitions of two more matrix operators: the star product operator and the linear fractional transformation (LFT) operator.

Definition 2 (Matrix star product operator [53]) *Given the two partitioned matrices,*

$$A^{(n_1+n_2)\times(m_1+m_2)} = \begin{bmatrix} A_{11}^{n_1\times m_2} & A_{12}^{n_1\times m_2} \\ A_{21}^{n_2\times m_1} & A_{22}^{n_2\times m_2} \end{bmatrix}, \qquad \mathcal{A} = \begin{bmatrix} \mathcal{A}_{11}^{m_2\times n_2} & \mathcal{A}_{12}^{m_2\times m_3} \\ \mathcal{A}_{21}^{n_3\times n_2} & \mathcal{A}_{22}^{n_3\times m_3} \end{bmatrix} \tag{2.4}$$

the star product operator $*$ *is defined by*

$$a = A * \mathcal{A} = \begin{bmatrix} A_{11} + A_{12}\mathcal{A}_{11}(I - A_{22}\mathcal{A}_{11})^{-1}A_{21} & A_{12}(I - \mathcal{A}_{11}A_{22})^{-1}\mathcal{A}_{12} \\ \mathcal{A}_{21}(I - A_{22}\mathcal{A}_{11})^{-1}A_{21} & \mathcal{A}_{22} + \mathcal{A}_{21}A_{22}(I - \mathcal{A}_{11}A_{22})^{-1}\mathcal{A}_{12} \end{bmatrix} \tag{2.5}$$

where one assumes the stated inverses exist and verifies that the dimensions are consistent.

In the case where $\mathcal{A}_{11}$ is the only non-empty submatrix of $\mathcal{A}$, then only the upper-left submatrix of $a = A * \mathcal{A}$ will be non-empty and we instead write $a = a_{11} = A \diamond \mathcal{A}$.

Definition 3 (Matrix LFT operator) *Given the two partitioned matrices,*

$$A^{(n_1+n_2)\times(m_1+m_2)} = \begin{bmatrix} A_{11}^{n_1\times m_2} & A_{12}^{n_1\times m_2} \\ A_{21}^{n_2\times m_1} & A_{22}^{n_2\times m_2} \end{bmatrix}, \qquad \mathcal{A} = \begin{bmatrix} \mathcal{A}_{11}^{m_2\times n_2} & [\,] \\ [\,] & [\,] \end{bmatrix} \tag{2.6}$$

the LFT operator $\diamond$ *is defined by*

$$a = a_{11} = A \diamond \mathcal{A} = A_{11} + A_{12}\mathcal{A}_{11}(I - A_{22}\mathcal{A}_{11})^{-1}A_{21}. \tag{2.7}$$

With the definitions above, SISO systems may be viewed as a subset of TITO systems where the transfer functions P_{12}, P_{21}, and P_{22} and the matrices B_2, D_{12}, C_2, D_{21}, and D_{22} are empty.

The fundamental TITO building block is the cascade interconnection shown in Figure 2.1 from which all TITO and SISO interconnections may be formed. Formally, given the two TITO systems

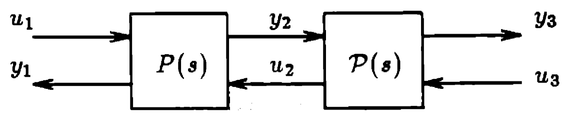


Figure 2.1: TITO interconnection.

$P(s)$ and $\mathcal{P}(s)$ with representations

$$P(s) = \left[\begin{array}{c:c} P_{11}(s) & P_{12}(s) \\ \hdashline P_{21}(s) & P_{22}(s) \end{array}\right] \longleftrightarrow \left[\begin{array}{c|c:c} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ \hdashline C_2 & D_{21} & D_{22} \end{array}\right] \tag{2.8}$$

$$\mathcal{P}(s) = \left[\begin{array}{c:c} \mathcal{P}_{11}(s) & \mathcal{P}_{12}(s) \\ \hdashline \mathcal{P}_{21}(s) & \mathcal{P}_{22}(s) \end{array}\right] \longleftrightarrow \left[\begin{array}{c|c:c} \mathcal{A} & \mathcal{B}_1 & \mathcal{B}_2 \\ \hline \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \\ \hdashline \mathcal{C}_2 & \mathcal{D}_{21} & \mathcal{D}_{22} \end{array}\right] \tag{2.9}$$

the cascade interconnection of $P(s)$ with $\mathcal{P}(s)$ results in the system $p(s)$ with a transfer function given by the star product of $P(s)$ with $\mathcal{P}(s)$. The transfer function star product operator is defined in the same way as it is for regular matrices [18], namely

$$p = P * \mathcal{P} = \begin{bmatrix} P_{11} + P_{12}\mathcal{P}_{11}(I - P_{22}\mathcal{P}_{11})^{-1}P_{21} & P_{12}(I - \mathcal{P}_{11}P_{22})^{-1}\mathcal{P}_{12} \\ \mathcal{P}_{21}(I - P_{22}\mathcal{P}_{11})^{-1}P_{21} & \mathcal{P}_{22} + \mathcal{P}_{21}P_{22}(I - \mathcal{P}_{11}P_{22})^{-1}\mathcal{P}_{12} \end{bmatrix} \tag{2.10}$$

where the functional dependence on s has been dropped to streamline the notation. Similarily, after some algebra manipulation it can be shown that the state space representation of $p(s)$ may be given in terms of star products of appropriately partitioned matrices. Specifically,

$$p(s) \longleftrightarrow \left[\begin{array}{c|c} \begin{bmatrix} A & B_2 \\ C_2 & D_{22} \end{bmatrix} * \begin{bmatrix} \mathcal{D}_{11} & \mathcal{C}_1 \\ \mathcal{B}_1 & \mathcal{A} \end{bmatrix} & \begin{bmatrix} B_1 & B_2 \\ D_{21} & D_{22} \end{bmatrix} * \begin{bmatrix} \mathcal{D}_{11} & \mathcal{D}_{12} \\ \mathcal{B}_1 & \mathcal{B}_2 \end{bmatrix} \\ \hline \begin{bmatrix} C_1 & D_{12} \\ C_2 & D_{22} \end{bmatrix} * \begin{bmatrix} \mathcal{D}_{11} & \mathcal{C}_1 \\ \mathcal{D}_{21} & \mathcal{C}_2 \end{bmatrix} & \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} * \begin{bmatrix} \mathcal{D}_{11} & \mathcal{D}_{12} \\ \mathcal{D}_{21} & \mathcal{D}_{22} \end{bmatrix} \end{array}\right] \tag{2.11}$$

where the state partitioning is marked and the input/output partitioning is given by the definition of the star product in (2.5) above.

It should be clear that the LFT interconnection shown in Figure 1.1 is just a specialization of the star product described above where

$$\mathcal{P}(s) = \begin{bmatrix} K(s) & \vdots & [\,] \\ \cdots & \cdot & \cdots \\ [\,] & \vdots & [\,] \end{bmatrix} \longleftrightarrow \left[\begin{array}{c|cc} A_c & B_c & \vdots \; [\,] \\ \hline C_c & D_c & \vdots \; [\,] \\ \cdots & \cdots & \cdot \; \cdots \\ [\,] & [\,] & \vdots \; [\,] \end{array}\right] \tag{2.12}$$

and $P(s)$ remains as in (2.9). The interconnected system p has the transfer function

$$p(s) = \begin{bmatrix} p_{11}(s) & \vdots & [\,] \\ \cdots & \cdot & \cdots \\ [\,] & \vdots & [\,] \end{bmatrix} = \begin{bmatrix} P_{11}(s) & \vdots & P_{12}(s) \\ \cdots & \cdot & \cdots \\ P_{21}(s) & \vdots & P_{22}(s) \end{bmatrix} * \begin{bmatrix} K(s) & \vdots & [\,] \\ \cdots & \cdot & \cdots \\ [\,] & \vdots & [\,] \end{bmatrix}$$

which one abbreviates as $p(s) = p_{11}(s) = P(s) \diamond K(s) = F_\ell(P, K)$. From (2.10) it is clear that

$$P \diamond K = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} \tag{2.13}$$

where the dependence on s has again been dropped for notational streamlining. Additionally, the state space representation of $F_\ell(P, K)$ is given by

$$F_\ell(P, K) \longleftrightarrow \left[\begin{array}{c|c} \begin{bmatrix} A & B_2 \\ C_2 & D_{22} \end{bmatrix} * \begin{bmatrix} D_c & C_c \\ B_c & A_c \end{bmatrix} & \begin{bmatrix} B_1 & B_2 \\ D_{21} & D_{22} \end{bmatrix} * \begin{bmatrix} D_c & [\,] \\ B_c & [\,] \end{bmatrix} \\ \hline \begin{bmatrix} C_1 & D_{12} \\ C_2 & D_{22} \end{bmatrix} * \begin{bmatrix} D_c & C_c \\ [\,] & [\,] \end{bmatrix} & \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \diamond D_c \end{array}\right] \tag{2.14}$$

which follows directly from (2.11).

2.2 Early static output feedback formulations

This section reviews the work of Athans, Levine, and Johnson [32, 25] from the early 1970's. Article [32] derives necessary equations for the optimal static output feedback regulator for LTI systems. Article [25] derives necessary equations for optimal regulation of LTI systems via fixed order but otherwise unconstrained compensation. Here we review the results of [32] in §2.2.1 and [25] in §2.2.2 and include references to related works.

2.2.1 Optimal static output feedback

Levine and Athans [32] derive a set of necessary equations for the optimal state regulator via static output feedback for deterministic systems with known initial state covariance and quadratic cost. Specifically, assume a system given by the equations

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \\ u &= -Fy \end{aligned} \tag{2.15}$$

with the goal of designing F to minimize the cost

$$J = \frac{1}{2}\int_0^\infty \mathcal{E}\{x'Qx + u'Ru\}dt \tag{2.16}$$

subject to known initial state covariance, X. For well-posed problems, Q will be positive semi-definite symmetric and R will be positive definite symmetric.

With the above formulation the necessary equations for optimality are:

$$F = R^{-1}B'KLC'(CLC')^{-1} \tag{2.17}$$

$$0 = (A - BFC)L + L(A - BFC)' + X \tag{2.18}$$

$$0 = K(A - BFC) + (A - BFC)'K + Q + C'F'RFC \tag{2.19}$$

with (2.18) and (2.19) recognized as Lyapunov equations. Equation (2.18) is known as the state Lyapunov equation with solution L, and (2.19) is known as the cost Lyapunov equation with solution K. Derivation of the optimality equations typically involves setting up a Lagrangian and employing matrix differentiation properties [1]. Various derivations of Levine and Athans' main result appear in [29], [38], and [22].

Note that F may be eliminated from (2.17)-(2.19) by substituting (2.17) into (2.18) and (2.19) thus yielding two coupled equations: one a modified Lyapunov equation, and the other a modified Ricatti equation. Thus we refer to equations of the type (2.17)-(2.19) as coupled Ricatti/Lyapunov equations. Because no closed form solutions presently exist to solve coupled Ricatti/Lyapunov equations, one must adopt an iterative search algorithm to solve (2.17)-(2.19). Towards this end, §2.4 summarizes one recently developed algorithm which has been implemented in MATLAB.

Finally note that although equations of the type (2.17)-(2.19) are necessary for optimality, they are not sufficient. Thus both local and global cost minima will satisfy the coupled Ricatti/Lyapunov equations. See §4 for further discussion of this fact and others concerning the practical implementation of the algorithm of §2.4.

2.2.2 Optimal fixed-order dynamic feedback

The work of Johnson and Athans [25] parallels that of Levine and Athans, expanding their results to dynamic compensation. Specifically, assume the configuration shown in Figure 2.2, with the fixed order but otherwise unconstrained controller given by A_c, B_c, C_c, D_c and the plant given by A, B, C. With the following definitions

$$F = \begin{bmatrix} D_c & C_c \\ B_c & A_c \end{bmatrix}, \hat{A} = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}, \hat{B} = \begin{bmatrix} B & 0 \\ 0 & I \end{bmatrix}, \hat{C} = \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}, \hat{x} = \begin{bmatrix} x \\ x_c \end{bmatrix},$$

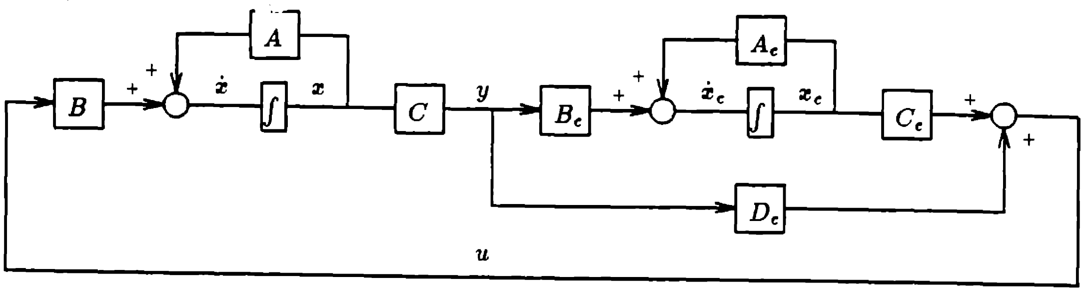


Figure 2.2: Fixed order compensator of Johnson and Athans.

$$\hat{R} = \begin{bmatrix} R_1 & 0 \\ 0 & R_2 \end{bmatrix}, \hat{Q} = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix}, W = \begin{bmatrix} X & 0 \\ 0 & 0 \end{bmatrix}, \text{ and } T = \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix}.$$

one may write a cost function and optimality equations in a manner analogous to §2.2.1. Specifically defining the cost

$$\begin{aligned} J &= \frac{1}{2}\int_0^\infty \mathcal{E}\{\hat{x}'\hat{Q}\hat{x} + u'\hat{R}u\}dt \\ &= \frac{1}{2}\int_0^\infty \mathcal{E}\{x'Qx + y'[B_c'R_2B_c + D_c'R_1D_c]y + x_c'[A_c'R_2A_c + C_c'R_1C_c]x_c\}dt \end{aligned}$$

where X is again the covariance of the initial plant state, one may write optimality equations

$$\begin{aligned} F &= \hat{R}^{-1}\hat{B}'KL\hat{C}(0.5 * \hat{C}(L + TLT)\hat{C}')^{-1} \\ 0 &= (\hat{A} - \hat{B}F\hat{C})L + L(\hat{A} - \hat{B}F\hat{C})' + W \\ 0 &= K(\hat{A} - \hat{B}F\hat{C}) + (\hat{A} - \hat{B}F\hat{C})'K + \hat{Q}. \end{aligned}$$

So the solution is again characterized by three equations (which logically resemble those of Levine and Athans) and three unknowns, F,K, and L. One recognizes F as a static output feedback gain for the system defined by $\hat{A}$, $\hat{B}$, and $\hat{C}$. Necessary equations for variations on this problem may be found in [2] and [60].

One might expect the low order optimal control problem to have multiple solutions based on the fact that a given system has infinitely many state space realizations resulting from state similarity transforms. Indeed [8] verifies that the optimal equations above are invariant to similarity transforms on either the compensator or the plant. Additionally, multiple solutions to the optimal equations may exist which are not equivalent through a similarity transform [23]. The non-uniqueness of minima becomes significant in developing search techniques for solving the optimal equations as will be discussed in §4.

2.3 Advanced static output feedback formulation

This section primarily restates the results of recent work by Bernstein, Haddad, and Nett (BHN). In [49, 5], BHN extend earlier work by incorporating arbitrary constraints on the compensator to be designed, by accommodating disturbances in plant state and plant output, and by introducing an infinity norm bound on a particular closed loop transfer function. We do not restate their main result (see [5, Theorem 3.1]) but rather a particular specialization (see [5, Case 1, p.2509]) whose necessary equations for optimality resemble those discussed in the previous section. Specifically given the system equations

$$\begin{aligned} \dot{x} &= Ax + D_1 w + Bu \\ z_2 &= E_1 x + E_2 u \\ z_\infty &= E_{1\infty} x + E_\infty w + E_{2\infty} u \\ y &= Cx \\ u &= Fy \end{aligned} \tag{2.20}$$

we wish to minimize the 2-norm of the transfer function from w to z_2 (T_{wz_2}) while ensuring the infinity norm of the transfer function from w to z_∞ (T_{wz_∞}) remains upper bounded by γ, *i.e.*

$$\| T_{wz_\infty} \|_\infty \leq \gamma. \tag{2.21}$$

Similarities to Levine and Athans become pronounced by defining

$$\begin{aligned} \hat{A} &= A + BFC && (2.22) \\ V &= D_1(I - \gamma^{-2} E_\infty' E_\infty)^{-1} D_1' \\ \alpha^2 R = R_2 &= E_2' E_2 && (2.23) \\ \beta^2 R = R_{2\infty} &= E_{2\infty}' E_{2\infty} && (2.24) \\ M &= I - \gamma^{-2} E_\infty E_\infty' \\ \tilde{R} &= (E_1 + E_2 FC)'(E_1 + E_2 FC) \\ \tilde{R}_\infty &= (E_{1\infty} + E_{2\infty} FC)' M^{-1} (E_{1\infty} + E_{2\infty} FC) \\ \tilde{R}_{01\infty} &= E_\infty' M^{-1} (E_{1\infty} + E_{2\infty} FC) \\ P_a &= B'K + E_2' E_1 + \gamma^{-2} E_{2\infty}' M^{-1} (E_{1\infty} LK + E_\infty D_1' K) && (2.25) \end{aligned}$$

where one requires that control weights, R_2 and $R_{2\infty}$, be scalar multiples of each other (an imposition which results in greatly simplified optimality equations, but which is not necessary). In this case the necessary equations for optimality are

$$F = -R^{-1} P_a LC' (\alpha^2 CLC' + \gamma^{-2} \beta^2 CLKLC')^{-1} \tag{2.26}$$

$$0 = (\hat{A} + \gamma^{-2} D_1 \tilde{R}_{01\infty}) L + L(\hat{A} + \gamma^{-2} D_1 \tilde{R}_{01\infty})' + \gamma^{-2} L \tilde{R}_{\infty} L + V \quad (2.27)$$

$$0 = (\hat{A} + \gamma^{-2} D_1 \tilde{R}_{01\infty} + \gamma^{-2} L \tilde{R}_{\infty})' K + K(\hat{A} + \gamma^{-2} D_1 \tilde{R}_{01\infty} + \gamma^{-2} L \tilde{R}_{\infty}) + \tilde{R} \quad (2.28)$$

which restates [5, Eq. (4.1)-(4.3)] with minor streamlining of notation. Again the solution involves three equations and three unknowns in a coupled Ricatti/Lyapunov framework. As such, the equations of (2.26)-(2.28) are amenable to solution via a variation of the feedback descent method discussed in the next section. Note that if the infinity norm constraint in the above formulation is eliminated by letting $\gamma \to \infty$ then one recovers the equations of Levine and Athans with the added feature of crossweighting between control input and state: *i.e.* $E_1'E_2$ may be nonzero. If $C = I$, $E_1'E_2 = 0$, and $\gamma \to \infty$ one recovers the full state feedback (LQR) result of §1.2. Similarily if $C = I$, $E_1'E_2 = 0$, and $\alpha = 0$ one recovers the H_∞ full state feedback solution.

This section concludes by mentioning two pertinent articles which predate BHN. First the idea of extracting a partially constrained dynamic compensator's parameters for independent optimization has origins in Wenk and Knapp [60]. Second, as explained more thoroughly in [5], the structure of the optimal equations is intimately tied to optimal projection theory ([3, 24]). Indeed, they derive both the full order LQG, H_∞, and H_2/H_∞ control laws as well as the optimal static output feedback equations as specializations of optimal projection results.

2.4 Solution of coupled Ricatti/Lyapunov equations

Returning now to the problem of finding solutions to coupled Ricatti/Lyapunov equations, this section summarizes the work of Beseler, Chow, and Minto [7] who recently developed the feedback descent (FD) method, an iterative algorithm for efficiently solving equations of the type found in Levine and Athans or BHN.

The FD method uses Newton's technique to approximate the objective, $J(F)$ around the current point in the search F_k with a quadratic function, and then minimizes this approximation. To get the quadratic approximation, expand J in a Taylor series about F_k and discard higher order terms so that

$$J(F) \approx J(F_k) + \nabla J(F_k)(F - F_k) + \frac{1}{2}(F - F_k)' J_{FF}(F_k)(F - F_k) \quad (2.29)$$

where $J_{FF}(F_k)$ is the Hessian which is required to be positive definite. Using matrix calculus [1], and taking derivatives with respect to F yields

$$\nabla J'(F) = \nabla J'(F_k) + J_{FF}(F - F_k).$$

Recalling that first order necessary conditions require $\nabla J(F^*) = 0$ implies

$$F^* = F_k - J_{FF}(F_k)^{-1} \nabla J'(F_k) \quad (2.30)$$

where $J_{FF}(F_k) > 0$ ensures that F^* is well defined.

In practice the FD algorithm alters (2.30) to better minimize $J(F)$ along $-J_{FF}(F_k)^{-1}\nabla J'(F_k)$ via employing a line search with a step size α

$$F_{k+1} = F_k - \alpha J_{FF}(F_k)^{-1}\nabla J'(F_k) \tag{2.31}$$

where α is chosen to guarantee that the cost does not increase from one iteration to the next. If the quadratic approximation (2.29) is good — if the present iteration is in the neighborhood of a minimum — then one expects $\alpha \to 1$.

Another way of obtaining (2.30) begins by applying a small perturbation

$$F \leftarrow F + \Delta F \tag{2.32}$$

to the necessary equation $\nabla J(f^*) = 0$. Thus

$$\nabla J(F + \Delta F) = \nabla J'(F) + J_{FF}(F)\Delta F + o(||\Delta F||) = 0$$

and so

$$\Delta F \approx -J_{FF}(F)^{-1}\nabla J'(F) \tag{2.33}$$

which yields the same gradient as in (2.30).

As found in previous sections, the first order necessary conditions for optimal control depend not only on the unknown set of control parameters found in matrix F but also on solutions (K,L) to auxilary coupled Lyapunov equations, re: (2.17)-(2.19) and (2.26)-(2.28). Thus one instead begins with a set of equations generally represented by

$$\mathcal{N}_i(K, L, F; M_j) = 0, \qquad i = 1, 2, 3 \tag{2.34}$$

where $\mathcal{N}$ denotes that the equations may be nonlinear in K, L, and F and contain other matrices given by the M_j's. For the Levine and Athans formulation, other matrices are A, B, C, Q, R, and X. In BHN's formulation, the M_j's are the matrices of the system equations (2.20). As in (2.32) above one introduces the perturbations

$$K \leftarrow K + \Delta K, \qquad L \leftarrow L + \Delta L, \qquad F \leftarrow F + \Delta F \tag{2.35}$$

into (2.34). Expanding the equations and dropping the second order and higher perturbation terms, one obtains a system of coupled matrix equations represented generally by

$$\mathcal{A}_i(\Delta K, \Delta L, \Delta F; K, L, F, M_j) = 0, \qquad i = 1, 2, 3 \tag{2.36}$$

where $\mathcal{A}$ denotes that the equations are affine in the Δ terms and are also functions of the matrices K, L, F, and M_i's. In other words, each of the three equations will be of the form

$$Y_1 \Delta K Y_2 + Y_3 \Delta L Y_4 + Y_5 \Delta F Y_6 + Y_7 = 0 \tag{2.37}$$

where the $Y_i = \mathcal{N}(K, L, F, M_j)$.

Remark 2 *In cases where the problem requires formulation as a decentralized static output feedback problem, the feedback descent algorithm becomes more complex than as presented here although the underlying ideas and techniques are the same. For a detailed explaination of the feedback descent algorithm in the case of decentralized feedback, consult Beseler's thesis. [6]*

Since the Δ terms of (2.36) must be found at each iteration, it is imperative that they be determined in an efficient manner. The main difference between the feedback descent (FD) algorithm and others employing Newton expansions (see for example [37, 58]) is the scheme used to solve Newton's equations; others use a congugate gradient technique, the FD algorithm finds a solution directly using matrix inversion. Specifically, the FD algorithm employs the **vec** operator and Kronecker products defined as in [10].

Definition 4 (vec operator) *Given the matrix A, the vector*

$$a = \mathbf{vec}A$$

is formed by stacking the columns of A. Similarily, one recovers A by simply unstacking $\mathbf{vec}A$.

Definition 5 (Kronecker product operator) *Given the $n \times m$ matrix A and the $p \times q$ matrix B, the Kronecker product of A and B is denoted as $A \otimes B$ and is given by the $np \times mq$ matrix*

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1m}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \cdots & a_{nm}B \end{bmatrix}$$

where a_{ij} is the ij^{th} element of A.

Continuing now with the FD algorithm, by exploiting the following property of the **vec** operator

$$vec(XYZ) = (Z' \otimes X)\mathbf{vec}Y,$$

one may write (2.36) as a single linear equation

$$H\xi = b \tag{2.38}$$

where

$$\xi = \begin{bmatrix} \mathbf{vec}\Delta K \\ \mathbf{vec}\Delta L \\ \mathbf{vec}\Delta F \end{bmatrix} \quad (2.39)$$

and where H and b are functions of K, L, F, and the M_j's. For the sample equation of (2.37), one finds

$$H = [\, Y_2' \otimes Y_1 \;\; Y_4' \otimes Y_3 \;\; Y_6' \otimes Y_5 \,]$$

and $b = \mathbf{vec} Y_7$.

One immediately observes that as a result of introducing Kronecker products, the size of the linear equation is large as compared to the original problem: $O(3n^2)$ vs $O(n)$. Thus a direct inversion may be computationally costly. Luckily for many applications, system matrices of the type found in Levine and Athans (2.15) or BHN (2.20) tend to be sparsely populated and thus sparse matrix techniques may be used to significantly reduce computational requirements. Further computational advantage may be gained in some problems by inverting a number of smaller matrices instead of a single large matrix.

To recap, this thesis proposes synthesizing constrained configuration controllers via posing an H_2 minimization (possibly with an H_∞ bound constraint) which is solved by satisfying first order necessary equations. These equations are in turn solved via the quasi–Newton feedback descent algorithm so that the overall 2–norm minimization is a truely second order technique. For clarity, the implementation of the FD algorithm is summarized as follows:

1. Initialization
   (a) set $k = 0$,
   (b) select an initial stabilizing gain F_0 for the system.
2. Iterative search
   (a) solve (2.34) for K and L and form H,
   (b) extract ΔF from (2.39) after solving for (2.38),
   (c) determine α^* to minimize $J(F)$ along ΔF,
   (d) $F_{k+1} \leftarrow F_k + \alpha * \Delta F$, $k \leftarrow k + 1$,
   (e) repeat search until convergence is achieved.

Given a general unstable system, the question of whether stabilizing static gains exist (and how to find one, should one exist) remains a topic of current research. Questions about the iterative search

— such as what are the particulars of (2a), what constitutes a good line search in (2c), and what are the stopping conditions in (2e) — will be discussed in §4. For now note that because of step (2c), the updated control law at each iteration will be admissable and non-inferior to the control law for the previous iteration.

Think globally; minimize locally.

Chapter 3

Controller Decomposition

This chapter contains the primary theoretical contribution of the thesis: a technique for decomposing a large class of arbitrarily tunable dynamic system into an LFT of a fixed dynamic system and a static, possibly block decentralized, parameter matrix. The need for a decentralized structure arises because methods for solution of SOF problems deal with only two types of formulations: fully centralized or block decentralized. The decomposition proposed here may be thought of as a two stage process. First §3.1 provides the particulars of an LFT decomposition of the controller to the SOF framework where the parameter matrix may not yet be in the required form. This intermediate parameter matrix may then be reduced to (block) decentralized form as in §3.2. The machinery used to form the LFT decomposition is familiar from μ analysis [51], but the application to control parameter optimization is new. The reduction procedure is also new, and generally results in a SOF problem which will be more fully centralized than the LFT used in μ applications (see Remark 8). This is an advantage since decentralized problems require more elaborate optimality conditions and greater computational time.

Next, §3.3 provides details of how the decomposition and reduction procedures apply to some simple controller structures. Then §3.4 provides LFT decompositions for more complicated controllers. Finally, §3.5 extends the decomposition result to include controllers with repeated parameters in their state space matrices. Here a further expansion of the multiplicative reduction procedure produces a SOF gain k with minimal repetition of parameters. Determining such a representation is of paramount importance in problems with repeated parameters because no known solutions exist for optimal SOF with repeated parameters.

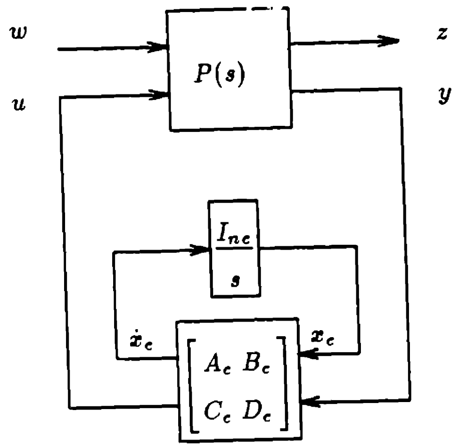


Figure 3.1: Morton and McAfoos interconnection.

3.1 LFT Decomposition to SOF Framework

The decomposition procedure proposed is most easily derived through a series of illustrations. One begins by assuming that the compensator has an n_c^{th} order state space representation of the form

$$
\begin{aligned}
K(s) \quad \longleftrightarrow \quad & \dot{x}_c = A_c x_c + B_c y \\
& u = C_c x_c + D_c y \\
\longleftrightarrow \quad & \left[\begin{array}{c|c} A_c & B_c \\ \hline C_c & D_c \end{array}\right]
\end{aligned}
\tag{3.1}
$$

and that it is connected in an LFT with the plant as in Figure 1.1. With a decomposition as shown in Figure 3.1, the packed matrix construct of (3.1) may be viewed as a regular gain matrix mapping the composite vector $[x_c' \; y']'$ to the vector $[\dot{x}_c' \; u']'$ [44]. The key step to producing the desired LFT interconnection is to assert that this regular gain matrix may be written as

$$
\begin{bmatrix} A_c & B_c \\ C_c & D_c \end{bmatrix} = \begin{bmatrix} \bar{a} & \bar{b} \\ \bar{c} & \bar{d} \end{bmatrix} + \begin{bmatrix} \hat{a} & \hat{b} \\ \hat{c} & \hat{d} \end{bmatrix} = \bar{k} + \hat{k} \tag{3.2}
$$

where $\bar{k}$ contains only zeros and those compensator parameters which are considered fixed or non-variable, and $\hat{k}$ contains only zeros and those parameters which are tunable. It is important to remember that neither the partitioned matrix $\bar{k}$ nor $\hat{k}$ is considered an independent dynamic system at this point. Indeed a well defined system need not result from either.

Remark 3 *Contrast the assumptions about parameter optimization (3.1)-(3.2) with those of model uncertainty typical of μ applications where one assumes a system with uncertainties may be modeled as*

$$\begin{aligned} \dot{x} &= \left(A_0 + \textstyle\sum_{i=1}^{k} \delta_i A_i\right) x + \left(B_0 + \textstyle\sum_{i=1}^{k} \delta_i B_i\right) u \\ y &= \left(C_0 + \textstyle\sum_{i=1}^{k} \delta_i C_i\right) x + \left(D_0 + \textstyle\sum_{i=1}^{k} \delta_i D_i\right) u. \end{aligned} \tag{3.3}$$

where the $\delta_i \in \Re[-1,1]$. Packard [51] shows how this system may be decomposed via an LFT into a dynamic system and a static gain where each uncertainty δ_i appears q_i times along the diagonal, and where

$$q_i = \textbf{rank} \begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix}. \tag{3.4}$$

Remark 4 *It is a fact that the class of systems given by (3.1)-(3.2) is contained within that given by (3.3). Indeed, the necessary specialization of the system given by (3.3) is*

$$\begin{aligned} \bar{k} &= \begin{bmatrix} A_0 & B_0 \\ C_0 & D_0 \end{bmatrix} \\ \hat{k} &= \sum_{i=1}^{n_c+p} \sum_{j=1}^{n_c+m} k_{ij} E_{ij} = \begin{bmatrix} \hat{a} & \hat{b} \\ \hat{c} & \hat{d} \end{bmatrix} \end{aligned}$$

where the E_{ij} are elementary matrices (i.e. contain zeros in all but their ij^{th} element which equals one). In this case, the decomposition of Packard results in a SOF gain matrix with the non-zero k_{ij} along the diagonal.

One may recognize that the summation of (3.2) can be used to produce a matrix LFT. Mathematically this is equivalent to realizing that

$$\begin{bmatrix} A_c & B_c \\ C_c & D_c \end{bmatrix} = \bar{k} + I \cdot \hat{k}(I - \hat{k} \cdot 0)^{-1} I \tag{3.5}$$

which is a matrix LFT equation as in (2.7). Thus one may write

$$\begin{bmatrix} A_c & B_c \\ C_c & D_c \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \bar{a} & \bar{b} \\ \bar{c} & \bar{d} \end{bmatrix} & I \\ I & 0 \end{bmatrix} \diamond \begin{bmatrix} \hat{a} & \hat{b} \\ \hat{c} & \hat{d} \end{bmatrix} = \begin{bmatrix} \bar{k} & I \\ I & 0 \end{bmatrix} \diamond \hat{k} \tag{3.6}$$

where the identity matrices have appropriate dimensions. Figure 3.2 provides a block diagram interpretation of the summation and LFT suggested by (3.2)-(3.6). One may verify that the transfer

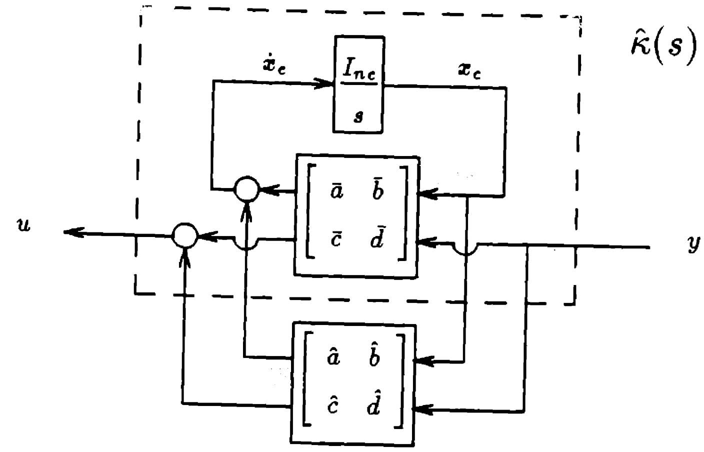


Figure 3.2: LFT Decomposition.

function, $K(s) = \hat{\kappa}(s) \diamond \hat{k}$, where

$$\hat{\kappa}(s) \longleftrightarrow \left[\begin{array}{c|c:cc} \bar{a} & \bar{b} & I & 0 \\ \hline \bar{c} & \bar{d} & 0 & I \\ \hdashline I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \end{array}\right].$$

As desired the dynamic system $\hat{\kappa}(s)$ contains no tunable parameters while the static gain matrix $\hat{k}$ contains only tunable parameters or zeros. Note also that the dynamic TITO system $\hat{\kappa}(s)$ and the original system $K(s)$ share the same dynamic order. The following theorem summarizes the development to this point.

Theorem 1 (LFT decomposition to SOF framework) *A constrained structure compensator, $K(s)$, satisfying equations (3.1)-(3.2) can be decomposed into an LFT of two systems, a TITO dynamic system $\hat{\kappa}(s)$ and a SISO static gain matrix $\hat{k}$, such that*

$$K(s) = \hat{\kappa}(s) \diamond k \tag{3.7}$$

where

$$\hat{\kappa}(s) \longleftrightarrow \left[\begin{array}{c|c:cc} \bar{a} & \bar{b} & I & 0 \\ \hline \bar{c} & \bar{d} & 0 & I \\ \hdashline I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \end{array}\right], \qquad \hat{k} = \begin{bmatrix} \hat{a} & \hat{b} \\ \hat{c} & \hat{d} \end{bmatrix} \tag{3.8}$$

Remark 5 *Using Equation 3.2 the compensator's state and dynamic equations (3.1) can be recovered easily from the decomposition.*

Remark 6 *Before considering how to reduce the static gain matrix of Theorem 1 to block diagonal form, consider the applicability of Theorem 1 to general MIMO controllers. Specifically, how general is the assumption that all elements of a controller's state space realization can be considered to be at most the sum of a fixed scalar and a tunable scalar? To address this issue, consider designing controller in either the time or frequency domain. In the time domain a designer works directly with the state space matrices and so it is only logical that the elements of the controller's state space matricies should either be constrained apriori or left tunable for optimization and thus Theorem 1 generally applies. On the other hand, when designing in the frequency domain, the elements of a controller's transfer function matrix are presumably composed of ratios of polynomials where the coefficients of the polynomials are either constrained or tunable for optimization. However it is well known that transfer function coefficients map directly into individual elements of canonical form state space representations, with the remaining elements of the state space matrices constrained to either one or zero [26, Ch.6]. Thus Theorem 1 may be applied in this case too.*

3.2 Multiplicative Reduction to Block Decentralized Structure

This section details a reduction procedure whereby the controller decomposition of Theorem 1 is converted to (block) decentralized form. The procedure may be viewed as having two steps:

1. the matrix $\hat{k}$ is multiplicatively decomposed as $\hat{k} = E_1 k E_2$ where the tunable submatrices of $\hat{k}$ lie along the block diagonal of k and where the columns of E_1 and rows of E_2 are elementary vectors; and

2. the dynamic TITO system $\hat{\kappa}(s)$ is augmented with the matrices E_1 and E_2 to form $\kappa(s)$ such that $K(s) = \kappa(s) \diamond k$ as depicted in Figure 1.2.

Before presenting a theorem for decomposing $\hat{k}$ to a block diagonal form, the following operators and notation will allow results to be stated in a compact form.

Definition 6 (Matrix delta function) *Given the $n \times m$ matrix, A, define the matrix operator*

$$\begin{aligned} \delta_A &= \delta(A) = \begin{cases} [\,] & \text{if } A = 0 \\ I_m & \text{otherwise} \end{cases} \\ \delta_{A'} &= \delta(A') = \begin{cases} [\,] & \text{if } A = 0 \\ I_n & \text{otherwise} \end{cases} \end{aligned} \tag{3.9}$$

where [] is the empty matrix as defined in (2.3).

Notation 1 *Given the matrix A, partitioned as A_{ij}, let*

$$\begin{aligned} \delta_{ij} &= \delta(A_{ij}) \\ \delta'_{ij} &= \delta(A'_{ij}) \end{aligned} \tag{3.10}$$

where $\delta(A_{ij})$ is as defined in (3.9).

Definition 7 (Block diagonal operator) *In the spirit of MATLAB's "diag" operator, define*

$$\begin{aligned} blk\_diag(M_1, M_2, \cdots, M_n) &= \begin{bmatrix} M_1 & & & \\ & M_2 & & \\ & & \ddots & \\ & & & M_n \end{bmatrix} \\ blk\_diag(M_1, [\,], M_2) &= blk\_diag(M_1, M_2) \end{aligned}$$

where the M_i are regular matrices or scalars and where blank entries imply zero.

As motivation for the next operator, consider stacking block diagonal matrices. For example,

$$\begin{bmatrix} blk\_diag(M_1, M_2, M_3) \\ blk\_diag(M_4, M_6, M_6) \end{bmatrix} = \begin{bmatrix} M_1 & 0 & 0 \\ 0 & M_2 & 0 \\ 0 & 0 & M_3 \\ M_4 & 0 & 0 \\ 0 & M_5 & 0 \\ 0 & 0 & M_6 \end{bmatrix} \tag{3.11}$$

assuming dimensions are conformable and none of the M_i are empty. But if some of the M_i are empty, the left hand side of (3.11) is ill defined. The next operator thus defines one way of stacking block diagonal matrices where some of the component matrices are empty.

Definition 8 (Stack diagonal operator) *When stacking block diagonal matrices, dimensional conformity must be maintained. Thus define the* stack_diag *operator such that*

$$\begin{aligned} stack\_diag(\; & M_{11}, \ldots, M_{1(i-1)}, & M_{1i}, & \quad M_{1(i+1)}, \ldots, M_{1p}; \; \ldots \\ & M_{21}, \ldots, M_{2(i-1)}, & [\,], & \quad M_{2(i+1)}, \ldots, M_{2p}; \; \ldots) \; = \end{aligned}$$

$$\left[\begin{array}{ccccc}
M_{11} & 0 & 0 & 0 & 0 \\
 & \ddots & & & \\
0 & M_{1(i-1)} & 0 & 0 & 0 \\
0 & 0 & M_{1i} & 0 & 0 \\
0 & 0 & 0 & M_{1(i+1)} & 0 \\
 & & & & \ddots \\
0 & 0 & 0 & 0 & M_{1p} \\
 & & \vdots & & \\
M_{21} & 0 & 0 & 0 & 0 \\
 & \ddots & & & \\
0 & M_{2(i-1)} & 0 & 0 & 0 \\
0 & 0 & 0 & M_{2(i+1)} & 0 \\
 & & & & \ddots \\
0 & 0 & 0 & 0 & M_{2p} \\
 & & \vdots & &
\end{array}\right]$$

where one assumes the matrix dimensions are conformable. Notice that

$$stack\_diag(M_1, M_2, M_3;\ M_4, M_5, M_6) = \left[\begin{array}{c} blk\_diag(M_1, M_2, M_3) \\ blk\_diag(M_4, M_6, M_6) \end{array}\right]$$

if none of the M_i are empty.

Notice that the *stack_diag* operator may still be ill defined if there exists a j such that $M_{ij} = []$ for all i. One could expand the definition of *stack_diag* so that it is well defined in this case too (for example by dropping the j^{th} block of columns altogether), but such complications are not necessary in our case.

Using the above operators and notation, a reduction procedure is summarized in the following theorem.

Theorem 2 (Block diagonal matrix reduction) *Given a matrix A partitioned according to $A_{ij}^{n_i \times m_j}$ $(i = 1, \ldots, n; j = 1, \ldots, m)$ satisfying:*

1. *no (block) row or (block) column of A may be identically zero, and*

2. *each submatrix A_{ij} is either fully tunable or identically equal to zero,*

then A may be reduced to block diagonal form A_d with the fully tunable submatrices placed along the block diagonal. One such reduction may be written as

$$A = E_1 A_d E_2 \tag{3.12}$$

where

$$\begin{aligned} E_1 &= blk\_diag([\delta'_{11} \cdots \delta'_{1m}], [\delta'_{21} \cdots \delta'_{2m}], \cdots, [\delta'_{n1} \cdots \delta'_{nm}]) \\ A_d &= blk\_diag(A_{11}\delta_{11}, \cdots, A_{1m}\delta_{1m}, \cdots, A_{n1}\delta_{n1}, \cdots, A_{nm}\delta_{nm}) \\ E_2 &= stack\_diag(\delta_{11}, \cdots, \delta_{1m}; \; \cdots \; \delta_{n1}, \cdots, \delta_{nm}) \end{aligned}$$

and where the columns of E_1 and the rows of E_2 are recognized as elementary vectors. Thus E_1 (E_2) has full column (row) rank and a left (right) inverse given by its transpose E'_1 (E'_2).

Proof: Simply verify (3.12) realizing that if submatrix $A_{ij} = 0$ then A_d will have fewer rows and fewer columns than it would if A_{ij} is fully tunable. Similarily, E_1 (E_2) will have fewer columns (rows) if $A_{ij} = 0$. A specific example of the reduction technique is

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & 0 \end{bmatrix}$$

where the a_{ij} are scalars. Thus

$$\begin{aligned} \delta_{11} &= \delta'_{11} = 1 \\ \delta_{12} &= \delta'_{12} = 1 \\ \delta_{21} &= \delta'_{21} = 1 \\ \delta_{22} &= \delta'_{22} = [\,] \\ E_1 &= blk\_diag([\,1\;1\,], [\,1\;[\,]\,]) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ A_d &= blk\_diag(a_{11}\cdot 1, a_{12}\cdot 1, a_{21}\cdot 1, a_{22}\cdot [\,]) = \begin{bmatrix} a_{11} & & \\ & a_{12} & \\ & & a_{21} \end{bmatrix} \\ E_2 &= stack\_diag(1, 1;\; 1, [\,]) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}. \end{aligned}$$

Now verify

$$A = E_1 A_d E_2 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & & \\ & a_{12} & \\ & & a_{21} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & 0 \end{bmatrix} \qquad \text{QED.}$$

Before applying Theorem 2 to the result of Theorem 1, some preliminary operations must be performed on $\hat{k}$ to ensure that it satisfies the requirements of Theorem 2. Specifically, zero rows and columns of $\hat{k}$ must be eliminated and the resulting matrix correctly partitioned. Fortunately, both tasks are trivially solved. The first step eliminates the identically zeros rows and columns of $\hat{k}$ via

$$\hat{k} = \hat{E}_1 \bar{k} \hat{E}_2 \tag{3.13}$$

where the column of $\hat{E}_1$ will be elementary vectors e_i with indices given by the non-zero rows of $\hat{k}$, and the rows of $\hat{E}_2$ will be elementary vectors e_i' with the indicies given by the non-zero rows of $\hat{k}$. Thus $\hat{E}_1$ ($\hat{E}_2$) has full column (row) rank and a left (right) inverse given by its transpose $\hat{E}_1'$ ($\hat{E}_2'$). Therefore,

$$\bar{k} = \hat{E}_1' \hat{k} \hat{E}_2'$$

where the dimensions of $\bar{k}$ are equal to or smaller than those of $\hat{k}$. The second step partitions $\bar{k}$ in the following manner:

- Construct a matrix T of the same dimensions as $\bar{k}$ such that

$$T_{ij} = \begin{cases} 1 & \text{if } \bar{k}_{ij} \text{ is tunable} \\ 0 & \text{if } \bar{k}_{ij} \text{ is not tunable} \end{cases} \tag{3.14}$$

  and consider T_i to be the i^{th} column of T and T_i' to be the i^{th} row of T;

- if $T_i \neq T_{i+1}$ then partition $\bar{k}$ between its i^{th} and $(i+1)^{th}$ column; and

- if $T_i' \neq T_{i+1}'$ then partition $\bar{k}$ between its i^{th} and $(i+1)^{th}$ row.

Now via Theorem 2, the partitioned matrix $\bar{k}$ may be written

$$\bar{k} = \bar{E}_1 k \bar{E}_2 \tag{3.15}$$

where k is now in the required form: tunable along its block diagonal. Combining the results of the two steps yields

$$\begin{aligned} \hat{k} &= \hat{E}_1 \bar{E}_1 k \bar{E}_2 \hat{E}_2 \\ &= E_1 k E_2 \end{aligned}$$

where $E_1 = \hat{E}_1 \bar{E}_1$ and $E_2 = \bar{E}_2 \hat{E}_2$.

Remark 7 *The columns (rows) of E_1 (E_2) will be elementary vectors. This is easily shown by observing that columns (rows) of E_1 (E_2) will be a linear combination of the elementary columns (rows) of $\hat{E}_1$ ($\hat{E}_2$) with the coefficients of the linear combination given by the elements of the elementary columns (rows) of $\tilde{E}_1$ ($\tilde{E}_2$).*

Remark 8 *The two step process may be combined into one by partitioning $\hat{k}$ without first producing $\tilde{k}$ and then defining a multiplicative decomposition of $\hat{k}$ similar to Theorem 2. However this method sometimes produces a decentralized structure for k when it may be possible to produce a fully populated k. For example, consider the following matrix and two possible reductions*

$$[\,A\ \ 0\ \ B\,] \;=\; I[\,A\ \ B\,]\begin{bmatrix} I & 0 & 0 \\ 0 & 0 & I \end{bmatrix} \tag{3.16}$$

$$\;=\; [\,I\ \ I\,]\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}\begin{bmatrix} I & 0 & 0 \\ 0 & 0 & I \end{bmatrix} \tag{3.17}$$

and verify that the method of Theorem 2 produces the reduction in (3.16) which is preferred.

At this point the dynamics portion of the compensator, $\hat{\kappa}(s)$, must be augmented with the matrices E_1 and E_2. If preferred, one can picture the matrix reduction included in the LFT decomposition of Figure 3.2 and can rewrite (3.5)-(3.6) as

$$\begin{bmatrix} A_c & B_c \\ C_c & D_c \end{bmatrix} = \bar{k} + E_1 k(I - 0\cdot k)^{-1}E_2 = \begin{bmatrix} \bar{k} & E_1 \\ E_2 & 0 \end{bmatrix} \circ k \tag{3.18}$$

from which one now constructs $K(s) = \kappa(s) \circ k$ where

$$\kappa(s) \longleftrightarrow \left[\begin{array}{c|c:c} \bar{a} & \bar{b} & E_{11} \\ \hline \bar{c} & \bar{d} & E_{12} \\ \cdots & \cdots & \cdots \\ E_{21} & E_{22} & 0 \end{array}\right] \tag{3.19}$$

where the E_{ij} are appropriate partitions of E_i and where k is as defined in (3.15).

3.3 Illustrative examples

This section presents simple examples of the decomposition/reduction procedure found above. The procedure is first applied to extract the parameters of a PI controller, a simple yet pertinent example since the problem of §5 utilizes PI control. Next the procedure is applied to extract the parameters of a three–block controller. This section closes with the decomposition/reduction of a three-input two-output decentralized lead/lag controller with integration and feedforward. Advantages of employing the algorithm developed above become readily apparent in decomposing this controller, as recasting the controller into a static output feedback problem by hand with an ad hoc technique would be a cumbersome and tedious experience.

3.3.1 PI Controller

Consider a design problem where one restricts the compensator to have a multivariable PI structure. In this case $K(s)$ has the form

$$K(s) = K_p + \frac{K_i}{s}$$

and a particular state space realization is

$$K(s) \longleftrightarrow \left[\begin{array}{c|c} 0 & I \\ \hline K_i & K_p \end{array}\right]$$

which may then be decomposed as in Figure 3.1. Rewriting the regular gain matrix as the summation

$$\begin{bmatrix} 0 & I \\ K_i & K_p \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ K_i & K_p \end{bmatrix} = \bar{k} + \hat{k} \tag{3.20}$$

allows for the application of Theorem 1. One finds

$$\hat{\kappa}(s) \longleftrightarrow \left[\begin{array}{c|c:cc} 0 & I & I & 0 \\ \hline 0 & 0 & 0 & I \\ \hdashline I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \end{array}\right]$$

and then proceeds to reducing the static gain matrix, $\hat{k}$. After eliminating the zero rows of $\hat{k}$ as in (3.13) via

$$\begin{bmatrix} 0 & 0 \\ K_i & K_p \end{bmatrix} = \begin{bmatrix} 0 \\ I \end{bmatrix} [K_i \; K_p] \begin{bmatrix} 0 & I \\ 0 & I \end{bmatrix}$$

one produces $K(s) = \kappa(s) \diamond k$ where

$$\kappa(s) \longleftrightarrow \left[\begin{array}{c|c:c} 0 & I & 0 \\ \hline 0 & 0 & I \\ \hdashline I & 0 & 0 \\ 0 & I & 0 \end{array}\right], \qquad k = [K_i \quad K_p]. \tag{3.21}$$

3.3.2 3 Block Problem

Now consider a design problem where the compensator's A, B, and C matrices are fully tunable while its D matrix is set apriori. In this case K has the form

$$K(s) = C(sI - A)^{-1}B + D \longleftrightarrow \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

and the summation of (3.2) yields

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & D \end{bmatrix} + \begin{bmatrix} A & B \\ C & 0 \end{bmatrix} = \bar{k} + \hat{k} \tag{3.22}$$

from which one produces $\hat{\kappa}(s)$. The result is

$$\hat{\kappa}(s) \longleftrightarrow \left[\begin{array}{c|c:cc} 0 & 0 & I & 0 \\ \hline 0 & D & 0 & I \\ \hdashline I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \end{array}\right]$$

but in this case one cannot reduce the static gain matrix $\hat{k}$ to a fully populated form. Instead one converts it to block decentralized form via Theorem 2 which yields the reduction

$$\begin{bmatrix} A & B \\ C & 0 \end{bmatrix} = \begin{bmatrix} I & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} A & & \\ & B & \\ & & C \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & I \\ I & 0 \end{bmatrix}$$

and thus $K(s) = \kappa(s) \diamond k$ where

$$\kappa(s) \longleftrightarrow \left[\begin{array}{c|c:ccc} 0 & 0 & I & I & 0 \\ \hline 0 & D & 0 & 0 & I \\ \hdashline I & 0 & & & \\ 0 & I & & & \\ I & 0 & & & \end{array}\right], \quad k = \begin{bmatrix} A & & \\ & B & \\ & & C \end{bmatrix}$$

and blank entries imply zero.

3.3.3 Decentralized lead/lag with integration and feedforward

Now consider a design problem where one wishes to design a three-input two-output compensator and structure restricted to have direct feedforward from the first input to the outputs and decentralized lead/lag with integration from the second and third inputs to the outputs. In other words

$$K(s) \quad = \quad C(sI-A)^{-1}B+D = \begin{bmatrix} N_1 & \frac{s+c_1}{s^2+\tau_1^{-1}s} & \\ N_2 & & \frac{s+c_2}{s^2+\tau_2^{-1}s} \end{bmatrix} \tag{3.23}$$

$$\longleftrightarrow \quad \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{cccc|ccc} 0 & 1 & & & 0 & 0 & 0 \\ 0 & -\tau_1^{-1} & & & 0 & 1 & 0 \\ & & 0 & 1 & 0 & 0 & 0 \\ & & 0 & -\tau_2^{-1} & 0 & 0 & 1 \\ \hline c_1 & 1 & 0 & 0 & N_1 & 0 & 0 \\ 0 & 0 & c_2 & 1 & N_2 & 0 & 0 \end{array}\right] \tag{3.24}$$

where one recognizes the state space representation of $K(s)$ to be in canonical form. This control structure is examined because it is actually used to control the plant of §5 at various operating points. As optimization software for the decentralized case is still under development, no optimal controllers of this structure will be generated in this thesis. However an LFT decomposition of the controller can nonetheless be generated using the same algorithm as above. Eliminating the intermediate steps due to space considerations, one finds

$$\kappa(s) \quad \longleftrightarrow \quad \left[\begin{array}{cccc|ccc:cccccc} 0 & 1 & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & & & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ & & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hdashline 0 & 1 & 0 & 0 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & & & & & & \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & & & & & & \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & & & & & & \end{array}\right]$$

$$k = \begin{bmatrix} -\tau_1^{-1} & & & & & \\ & -\tau_2^{-1} & & & & \\ & & c_1 & & & \\ & & & N_1 & & \\ & & & & c_2 & \\ & & & & & N_2 \end{bmatrix}$$

from which one verifies that indeed $K(s) = \kappa(s) \diamond k$.

3.4 Further examples and considerations

This section shows how the results of §3.1 and §3.2 may be used to encompass a wide range of control structures. The examples provided reinforce the idea that any control loop may be viewed in the standard output feedback configuration of Figure 1.1. As a result, the techniques of the previous sections may be applied to recast a large class of control loops into a static output feedback framework, a result which might be overlooked because previous examples only decomposed single–block–input single–block–output controllers. This section begins by reproducing a previous result in §3.4.1, which shows that a fixed order control problem may be recast as a fully centralized SOF problem by using a canonical form representation of the controller. Next §3.4.2 provides a decomposition for decentralized controllers in large scale systems. Then, §3.4.3 converts a two degree-of-freedom (2–DOF) problem into an equivalent SOF problem. The 2–DOF problem is shown to be a specialization of a more general controller which will be used in the next section to motivate the study of repeated parameter problems.

3.4.1 Canonical form representations

In [30], Kramer and Calise exploit canonical form representations of fixed order controllers to recast a disturbance rejection problem into an equivalent fully centralized SOF problem. Here this fact is shown to be a trivial extension of the results of previous sections. Indeed, by using canonical form realizations, any fixed order control problem may be recast as a fully centralized SOF problem. Verification of this claim requires only examining the structure of MIMO canonical form realizations. [26, Ch.6]

Definition 9 (MIMO canonical forms) *Given an n^{th} order, m–input, p–output compensator, with m controllability indices k_i satisfying*

1. $\sum_{i=1}^{m} k_i = n$, *and*

2. $k_i > k_{i+1}$,

then one controllable canonical form realization will be

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{cccc|c} A_{11} & A_{12} & \cdots & A_{1m} & \\ \vdots & \vdots & \ddots & \vdots & blk\_diag(e_{k_1}, \cdots, e_{k_m}) \\ A_{m1} & A_{m2} & \cdots & A_{mm} & \\ \hline x & x & \cdots & x & \\ \vdots & \vdots & \ddots & x & 0 \\ x & x & \cdots & x & \end{array}\right] \tag{3.25}$$

where

$$A_{ii} = \begin{bmatrix} x & \cdots & \cdots & x \\ & I_{k_i-1} & & 0_{(k_i-1)\times 1} \end{bmatrix}_{k_i \times k_i} \quad \text{and} \quad A_{ij} = \begin{bmatrix} x & \cdots & x \\ & 0_{(k_i-1)\times k_j} & \end{bmatrix}_{k_i \times k_j} \quad \forall i \neq j \tag{3.26}$$

and where the e_{k_i} are elementary column vectors of length k_i whose top element is 1. Similarily, one observable canonical form will be given by the triple (A', C', B'), and the k_i will then be referred to as observability indices.

Remark 9 *For SISO systems, the controllable canonical form reduces to the familiar*

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{ccccc|c} x & x & \cdots & x & x & 1 \\ 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & & 0 & 0 & 0 \\ \vdots & & \ddots & & \vdots & \vdots \\ 0 & 0 & & 1 & 0 & 0 \\ \hline x & x & \cdots & x & 0 & 0 \end{array}\right]. \tag{3.27}$$

It should now be immediately obvious that the LFT decomposition and reduction procedures of §3.1 and §3.2 can recast the controllable (observable) canonical form compensator above into an LFT of a fixed dynamic system and a fully centralized parameter matrix $k_{(m+p)\times n}$ $(k_{n\times(p+m)})$. Specifically, assuming the controllability canonical form realization above, one finds that $\hat{k}$ of (3.2) will be

$$\hat{k} = \begin{bmatrix} \begin{bmatrix} x & \cdots & x \\ & 0_{(k_1-1)\times n} & \\ \vdots & \vdots & \vdots \\ x & \cdots & x \\ & 0_{(k_p-1)\times n} & \end{bmatrix} & 0_{n\times m} \\ C & 0 \end{bmatrix} \tag{3.28}$$

where C is fully tunable as in (3.26). After eliminating zero rows and columns of $\hat{k}$ as in (3.13), one finds

$$k = \begin{bmatrix} A_{k_i \times n} \\ C \end{bmatrix} \tag{3.29}$$

where the $A_{k_i \times n}$ are the fully tunable rows of A. We prefer the computer perform the calculations necessary for the dynamics portion of the compensator $\kappa(s)$ (or see [30] for details).

Compare this fully centralized structure to the block decentralized structure obtained for fixed order problems not exploiting canonical form realizations (*i.e.* the three block problem of §3.3.2).

3.4.2 Decentralized controllers for large scale systems

Recent work by Mercadal [39] focuses on generating decentralized controllers for large scale systems. Such controllers are characterized by r distinct subcontrollers which may be aggregated into a single state space realization given by

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \longleftrightarrow \left[\begin{array}{ccc|ccc} A_1 & & & B_{11} & \cdots & B_{1m} \\ & \ddots & & \vdots & \ddots & \vdots \\ & & A_r & B_{r1} & \cdots & B_{rm} \\ \hline C_{11} & \cdots & C_{1r} & & & \\ \vdots & \ddots & \vdots & & & \\ C_{p1} & \cdots & C_{pr} & & & \end{array}\right] \tag{3.30}$$

where $B_{ij} \in \Re^{(n_i \times 1)}$ and $C_{ij} \in \Re^{(1 \times n_i)}$. Each B_{ij} and C_{ij} will be fully tunable or identically equal to zero — thus system inputs may be determined by more than one subcontroller and system outputs may be seen by more than one subcontroller.

Since (3.30) may be decomposed as in (3.2), one can recast the decentralized controller into an LFT of a fixed dynamic system and a decentralized parameters matrix. Because the LFT result will depend on which of the B_{ij} and C_{ij} are tunable for optimization, no general solution is provided. However, consider the case where the entire B and C matrices are tunable for optimization and define

$$B = \begin{bmatrix} B_1 \\ \vdots \\ B_r \end{bmatrix} \quad \text{and} \quad C = [\, C_1 \; \cdots \; C_r \,] \quad \text{where} \quad B_i = [\, B_{i1} \; \cdots \; B_{im} \,] \quad \text{and} \quad C_i = \begin{bmatrix} C_{1i} \\ \vdots \\ C'_{pi} \end{bmatrix}.$$

In this case, one readily sees that

$$k = blk\_diag(A_1, B_1, A_2, B_2, \ldots, A_r, B_r, C_1, \ldots, C_r) \tag{3.31}$$

but allows a computer to manipulate of the dynamics portion of the controller.

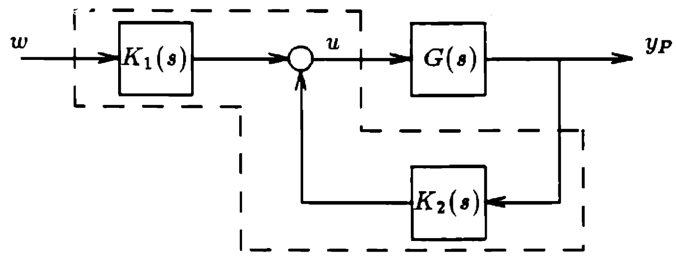


Figure 3.3: 2-DOF controller.

Remark 10 *Note that the decomposition of this decentralized controller is NOT as fully centralized as possible in the sense that a different reduction procedure could have produced a static output gain which has less diagonal matrix blocks than (3.31). For example, one could produce*

$$k = blk\_diag(A_1, A_2, \ldots, A_r, B, C) \tag{3.32}$$

via a reduction similar to that in §3.3.2. Software enhancements can certainly be included to check for decentralized regions in the unreduced gain $\hat{k}$.

3.4.3 2–DOF problem

Consider the 2-DOF control loop shown in Figure 3.3 where the controllers $K_i(s)$ are given by

$$K_i(s) \longleftrightarrow \left[\begin{array}{c|c} a_i & b_i \\ \hline c_i & d_i \end{array}\right]$$

and the plant, $G(s)$ is given by

$$G(s) \longleftrightarrow \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]. \tag{3.33}$$

Now let the single controller, $K_{2-DOF}(s)$, be defined by the dashed boundary as shown. In this case simple algebra yields

$$K_{2-DOF}(s) = [K_1(s) \ \ K_2(s)]$$

where the input to $K_{2-DOF}(s)$ is the vector $[w' \ y_P']'$ and its output is u. At the same time, the boundary defines a TITO system $P(s)$ such that $P(s)$ and $K_{2-DOF}(s)$ may be connected in the standard output feedback configuration. Assuming P has the external output $[u' \ y_P']'$, one can verify

$$P(s) = \begin{bmatrix} 0 & \vdots & I \\ 0 & \vdots & G(s) \\ \cdots & \cdot & \cdots \\ I & \vdots & 0 \\ 0 & \vdots & G(s) \end{bmatrix} \longleftrightarrow \left[\begin{array}{c|cc} A & 0 \ \vdots & B \\ \hline 0 & 0 \ \vdots & I \\ C & 0 \ \vdots & D \\ \cdots & \cdots \ \cdot & \cdots \\ 0 & I \ \vdots & 0 \\ C & 0 \ \vdots & D \end{array}\right]. \tag{3.34}$$

Now,

$$K_{2-DOF}(s) \longleftrightarrow \left[\begin{array}{cc|cc} a_1 & 0 & b_1 & 0 \\ 0 & a_2 & 0 & b_2 \\ \hline c_1 & c_2 & d_1 & d_2 \end{array}\right] \tag{3.35}$$

which may be decomposed as in previous sections once a designer specifies which controller parameters are to be left tunable for optimization.

Remark 11 *From an optimization standpoint, one may have to include extra optimality equations to ensure that the controller $K_1(s)$ is stable, but the question of how the configuration may be reduced to a static output feedback framework has been answered. As a partial answer to the question of ensuring the stability of $K_1(s)$, notice that the closed loop system T_{wy_P} will have an "A" matrix of the form*

$$\begin{bmatrix} * & * \\ 0 & a_1 \end{bmatrix}$$

for an appropriate state ordering. Thus any set of parameters which stabilize the closed loop system will also guarantee that $K_1(s)$ is stable.

Note the generality of a standard output feedback formulation with P as in (3.34) in that it allows the controller direct access to w and y_P. It follows that the most general control loop can be formulated in the standard output feedback framework with P as in (3.34) and K given by

$$K(s) = [\ K_{wu}(s) \ \ K_{yu}(s)\ ] \longleftrightarrow \left[\begin{array}{c|cc} A_c & B_1 & B_2 \\ \hline C_c & D_1 & D_2 \end{array}\right]. \tag{3.36}$$

In other words, all control problems may be formulated as specializations of this most general controller; for example, the 2–DOF example above has a specialization given by (3.35). As before, if all the elements of the state space matrices are assumed to be at most the sum of a fixed value and a tunable parameter, this general controller problem may be recast as a SOF problem.

Remark 12 *Vidyasager [59] calls the most general controller a "two parameter compensator." He refers to the 2–DOF problem as an "infeasible implementation of the two parameter compensator" because it requires $K_{wu} = K_1$ to be stable (i.e. not all general controllers have a 2–DOF implementation). Here the 2–DOF problem is assumed to arise from structural constraints imposed on the controller; it is not expected to implement the more general controller.*

As an example of decomposing another specialization of the general controller, consider a command tracking loop with PI and feedforward control as explored by Grace [19]. In this case,

$$K(s) = Cs^{-1}(w - y_P) + Fw - Dy_P \longleftrightarrow \left[\begin{array}{c|cc} 0 & I & -I \\ \hline C & F & -D \end{array}\right].$$

which may be decomposed using the techniques of §3.1 and §3.2 as $K(s) = \kappa(s) \circ k$ where

$$\kappa(s) \longleftrightarrow \left[\begin{array}{c|cc:c} 0 & I & -I & 0 \\ \hline 0 & 0 & 0 & I \\ \hdashline I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \end{array}\right], \qquad k = [\; C \;\; F \;\; -D \;]. \tag{3.37}$$

3.5 Redundant parameter elimination

From the results above, one might attempt framing all control problems as specializations of the general controller above. For example one might now ponder the implications of using the general controller to realize a PI controller. In this case the k of (3.37) becomes

$$k = [\; C \;\; D \;\; -D \;]$$

(*i.e.* $F = D$ and so $K(s) = (Cs^{-1} + D)(w - y_P)$) and one describes k as possessing repeated parameter D with multiplicity 2. At this juncture, the designer confronts having to manipulate symbolics in an effort to decrease the multiplicity of D. Computationally this requires reducing k further as

$$k = I[\; C \;\; D \;] \begin{bmatrix} I & 0 & 0 \\ 0 & I & -I \end{bmatrix} = E_1[\; C \;\; D \;]E_2 \tag{3.38}$$

and then augmenting $\kappa(s)$ in (3.37) with E_1 and E_2 to recover (3.21).

In the absence of software capable of symbolic manipulation, the designer retreats to formulating the problem as in §3.3.1. For present purposes, the example motivates further consideration of the repeated parameter phenomena. In the above problem, the repeated parameter arose because

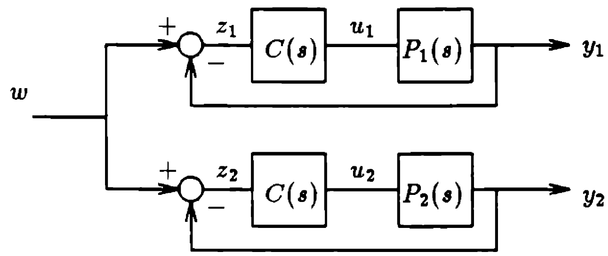


Figure 3.4: Multi-plant simultaneous stabilization problem.

posing the problem in the general control framework overparameterized the controller, but through a reduction similar to that of §3.2, all but a single occurrance of the parameter is eliminated. In general though, how and why do repeated parameters occur in controller formulations? And when and how might a reduction decrease the multiplicity of repeated parameters? In problems where repeated parameters arise, the answers to such questions become all important since optimization techniques (such as the one in §2.4) become significantly more complicated (bordering on intractible) when they must also guarantee repeated parameters. [19]

As a partial answer to the first question, consider for a moment the model of system uncertainty discussed in Remark 3. If instead of defining model uncertainties the δ_i of (3.3) define the parameters to be optimized in a constrained control problem, one finds that the controller has repeated parameters δ_i of multiplicity q_i where the q_i are given by (3.4). However a controller given by (3.3) has little physical interpretation. Figure 3.4 provides a more practical example of how repeated parameters might arise. In this case, the same controller is required to stabilize multiple plants in the presence of external disturbances [41]. If

$$C(s) \longleftrightarrow \left[\begin{array}{c|c} a & b \\ \hline c & d \end{array}\right]$$

then

$$K(s) = \begin{bmatrix} C(s) & \\ & C(s) \end{bmatrix} \longleftrightarrow \left[\begin{array}{cc|cc} a & & b & \\ & a & & b \\ \hline c & & d & \\ & c & & d \end{array}\right]$$

where K has input $[z_1' \;\; z_2']'$ and output $[u_1' \;\; u_2']'$. In this case that the repeated parameters cannot be eliminated via any multiplicative reduction as in §3.2.

Remark 13 *By now it should be obvious to the reader that, if all the parameters of $C(s)$ are* *tunable for optimization, the decomposition reduction procedure of §3.1 and §3.2 will yield* $k = blk\_diag(a, b, a, b, c, d, c, d)$.

Remark 14 *Parameters may be rearranged along the block diagonal by pre– and post–multiplying by permutation matrices. For example,*

$$k_p = P_1 k P_2 \tag{3.39}$$

where

$$\begin{aligned} k_p &= blk\_diag(a, a, b, b, c, c, d, d) \\ k &= blk\_diag(a, b, a, b, c, d, c, d) \end{aligned}$$

and

$$P_1 = \begin{bmatrix} \delta_{a'} & 0 & 0 & 0 & & & & \\ 0 & 0 & \delta_{a'} & 0 & & & & \\ 0 & \delta_{b'} & 0 & 0 & & & & \\ 0 & 0 & 0 & \delta_{b'} & & & & \\ & & & & \delta_{c'} & 0 & 0 & 0 \\ & & & & 0 & 0 & \delta_{c'} & 0 \\ & & & & 0 & \delta_{d'} & 0 & 0 \\ & & & & 0 & 0 & 0 & \delta_{d'} \end{bmatrix}, \quad P_2 = \begin{bmatrix} \delta_{a} & 0 & 0 & 0 & & & & \\ 0 & 0 & \delta_{a} & 0 & & & & \\ 0 & \delta_{b} & 0 & 0 & & & & \\ 0 & 0 & 0 & \delta_{b} & & & & \\ & & & & \delta_{c} & 0 & 0 & 0 \\ & & & & 0 & 0 & \delta_{c} & 0 \\ & & & & 0 & \delta_{d} & 0 & 0 \\ & & & & 0 & 0 & 0 & \delta_{d} \end{bmatrix}.$$

Since a permutation matrix is also orthogonal (i.e. $P_i^{-1} = P_i'$*) one can write*

$$\begin{aligned} k &= P_1' P_1 k P_2 P_2' \\ &= P_1' k_p P_2'. \end{aligned}$$

Augmenting the plant with P_1' *and* P_2'*, one may thus order the parameters as desired along the block diagonal.*

Consider now framing the simultaneous stabilization problem above as a specialization of the general controller so that its boundaries are now defined by $[w' \;\; y_1' \;\; y_2']'$ and $[u_1' \;\; u_2']'$. Assuming P

outputs $[u_1' \ u_2' \ y_1' \ y_2']'$ one finds

$$P(s) = \begin{bmatrix} 0 & \vdots & I & 0 \\ 0 & \vdots & 0 & I \\ 0 & \vdots & P_1(s) & 0 \\ 0 & \vdots & 0 & P_2(s) \\ \cdots & \cdot & \cdots & \cdots \\ I & \vdots & 0 & 0 \\ 0 & \vdots & P_1(s) & 0 \\ 0 & \vdots & 0 & P_2(s) \end{bmatrix}, \quad K(s) = \begin{bmatrix} C(s) & -C(s) & 0 \\ C(s) & 0 & -C(s) \end{bmatrix} \longleftrightarrow \left[\begin{array}{cc|ccc} a & & b & -b & 0 \\ & a & b & 0 & -b \\ \hline c & & d & -d & 0 \\ & c & d & 0 & -d \end{array}\right] \tag{3.40}$$

and one may decompose K as $\hat{\kappa}(s) \diamond \hat{k}$ as in §3.1. This $\hat{k}$ may then be decomposed as

$$\hat{k} = E_1 k E_2 = \begin{bmatrix} I & & & \\ & I & & \\ & & I & \\ & & & I \end{bmatrix} \begin{bmatrix} a & & & & & & & \\ & b & & & & & & \\ & & a & & & & & \\ & & & b & & & & \\ & & & & c & & & \\ & & & & & d & & \\ & & & & & & c & \\ & & & & & & & d \end{bmatrix} \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & 0 & I & -I & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & -I \\ I & 0 & 0 & 0 & 0 \\ 0 & 0 & I & -I & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & -I \end{bmatrix} \tag{3.41}$$

where all the identities are assumed to be of the appropriate dimension. As before, $\hat{\kappa}(s)$ can be augmented with E_1 and E_2 to produce $\kappa(s)$ such that $K(s) = \kappa(s) \diamond k$.

Even in the general control framework, the repeated parameters continue to have multiplicity 2, but the problem has provided some insight into how to minimize the number of repeated parameters in a SOF framework. For example, notice that the columns (rows) of E_1 (E_2) need no longer be elementary vectors as was the case for problems without repeated parameters.

With intuition gained in the previous example, one might now pose the following questions: Given that one has defined the boundaries of some controller K and produced the unreduced decomposition $K(s) = \hat{\kappa}(s) \diamond \hat{k}$ (presumably with repeated parameters in $\hat{k}$) can the reduction of $\hat{k}$ still be automated? And is there a minimal repetition decomposition? The answer to both these questions is "yes;" the following theorem summarizes the reduction for the scalar parameter case.

Theorem 3 (Minimal repetition reduction) *Given the $n \times m$ matrix A with elements A_{ij} satisfying:*

(a) each element A_{ij} is either fully tunable or identically equal to zero, and

(b) a tunable element of A may be repeated elsewhere in A, but all tunable elements of A are given by the parameters $\rho_1, \ldots, \rho_r$,

form the matrices A_i, $i = 1, \ldots, r$ as $A_i = A|_{\rho_j=0,\forall j\neq i}$. Then

1. *A may be reduced to*

$$A = E_1 A_d E_2 \tag{3.42}$$

where A_d is a diagonal matrix with the (repeated) parameter(s) ρ_i appearing exactly $\text{rank}(A_i)$ *times along the diagonal of A_d, and*

2. *A_d will be a "minimal repetition reduction" in the sense that no other multiplicative reduction of A exists such that the multiplicity of ρ_i will be less than* $\text{rank}(A_i)$.

Proof: One implementation of part (1) may be found below. A proof of part (2) begins with a reduction of A as in (3.42) where the multiplicity of parameter ρ_i is given by m_i. Now assume that there exists an i such that $m_i < \text{rank}(A_i)$. Without loss of generality (see Remark 14), let the parameters be ordered along the diagonal of A_d and partition E_1, E_2, and A_d as

$$\begin{aligned} A &= E_1 A_d E_2 = [\, E_{11} \mid \ldots \mid E_{1r} \,] \; blk\_diag(\rho_1 I_{m_1}, \ldots, \rho_r I_{m_r}) \; [\, E'_{21} \mid \ldots \mid E'_{2r} \,]' \\ &= \rho_1 E_{11} E'_{21} + \ldots + \rho_r E_{1r} E'_{2r}. \end{aligned}$$

Recognizing that $A_i = \rho_i E_{1i} E'_{2i}$ implies that $\text{rank}(A_i) = m_i$ and so (2) is true by contradiction. QED

One implementation of part (1) of the above theorem begins with the matrices A_i, $i = 1, \ldots, r$. The reduction decomposes each as

$$A_i = \rho_i E_{1i} I_{m_i} E'_{2i}$$

and then constructs

$$\begin{aligned} A &= E_{11} \rho_1 I_{m_1} E'_{21} + \ldots + E_{1r} \rho_r I_{m_r} E'_{2r} \\ &= [\, E_{11} \mid \ldots \mid E_{1r} \,] \; blk\_diag(\rho_1 I_{m_1}, \ldots, \rho_r I_{m_r}) \; [\, E'_{21} \mid \ldots \mid E'_{2r} \,]' \\ &= E_1 A_d E_2. \end{aligned}$$

The reduction of each A_i proceeds as follows:

1. Eliminate all zeros columns and rows of the matrix,

2. Eliminate all repeated rows or columns of the matrix,

3. Form E_{1i} and E'_{2i}

The first step proceeds exactly as in (3.13) The second step is similar to (3.13) except that $\tilde{E}_1$ will have (possibly negative) elementary rows, and $\tilde{E}_2$ will have (possibly negative) elementary columns. For example

$$\begin{bmatrix} a & a & -a \\ a & 0 & -a \\ a & a & -a \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a & a \\ a & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}.$$

At this point, one has

$$A_i = \rho_i \hat{E}_1 E_i \hat{E}_2$$

where E_i will be composed of elements which are either 1, -1, or 0, and will have either full column rank, full row rank, or both. If E_i has dimensions $p \times q$ and $p < q$ then let $E_{1i} = \hat{E}_1$, $m_i = p$, and form $E'_{2i} = E_i \hat{E}_2$. Otherwise, let $E_{2i} = \hat{E}_2$, $m_i = q$, and form $E_{1i} = \hat{E}_1 E_i$.

Remark 15 *An analogous theorem may be developed for problems with repeated matrix parameters instead of repeated scalar parameters, thus producing a block diagonal k. As in Theorem 2, special care must be taken to ensure that the identity matrices which compose E_1 and E_2 are appropriately dimensioned.*

The proof is in the picture.

Chapter 4

Achieving Optimality: Computational Issues

This chapter discusses the implementation of the feedback descent (FD) algorithm introduced in §2.4 which may be used to iteratively solve coupled Ricatti/Lyapunov equations. Whereas §2.4 presented the algorithm from a theoretical perspective, this chapter explores more practical considerations including line search technique, special cases, and coding details. First §4.1 reviews the derivation of the FD algorithm for problems formulated as in Levine and Athans (2.15). Next §4.2 investigates the line search developed to accompany the FD algorithm. Issues addressed by the search include avoiding unstable gain regions and handling small step sizes. Finally §4.3 addresses initialization and convergence considerations. All the algorithms outlined in this chapter have been coded in MATLAB and incorporated into the GE proprietary toolbox ISICLE© [42].

4.1 Feedback descent algorithmic summary

This section assumes a system of equations with associated optimality equations as in Levine and Athans (2.15,2.17-2.19) and an initial stabilizing gain for the system, F_0. For decentralized problems or problems formulated as in BHN, an FD algorithm may be derived in a fashion similar to that below.

Following the theoretical roadmap of §2.4, first substitute the Newton expansions $F + \Delta F$, $K + \Delta K$, and $L + \Delta L$ for F, K, and L in (2.17)-(2.19). Next multiply out the equations and eliminate terms of order Δ^2 or higher. Cancelling the constant terms which sum to zero in the second and third equation and substituting for F in the first yields the three coupled equations [7]

$$
\begin{aligned}
&B'\Delta KLC' + (B'K - B'KLC'(CLC')^{-1}C)\Delta LC' - R\Delta FCLC' = (B'K - RFC)LC' \\
&0 = \Delta K(A - BFC) + (A - BFC)'\Delta K - (B'K - RFC)'\Delta FC - C'\Delta F'(B'K - RFC) \\
&0 = \Delta L(A - BFC)' + (A - BFC)\Delta L - B\Delta FCL - LC'\Delta F'B'
\end{aligned}
$$

which are affine in the Δ terms. Now the **vec** operator may be utilized, in conjunction with Kronecker products, to transform these three equations to a single linear equation with unknowns **vec**ΔK, **vec**ΔL, **vec**ΔF, and **vec**$\Delta F'$. A further simplification combines the **vec**ΔF and **vec**$\Delta F'$ terms to yield

$$
\begin{aligned}
H\xi &= \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & 0 & \hat{A}_{23} \\ 0 & A_{21}^T & \hat{A}_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ 0 \\ 0 \end{bmatrix} = b \qquad (4.1) \\
x_1 &= \mathbf{vec}\Delta K \\
x_2 &= \mathbf{vec}\Delta L \\
x_3 &= \mathbf{vec}\Delta F \\
b_1 &= \mathbf{vec}(RFC - B'K)LC' \\
A_{11} &= CL \otimes B' \\
A_{12} &= C \otimes [B'K - B'KLC'(CLC')^{-1}C] \\
A_{13} &= -(CLC') \otimes R \\
A_{21} &= (A - BFC)' \otimes I_n + I_n \otimes (A - BFC)' \\
\hat{A}_{23} &= \mathcal{T}(-C' \otimes (B'K - RFC)', -(B'K - RFC)' \otimes C') \\
\hat{A}_{33} &= \mathcal{T}(-LC' \otimes B, -B \otimes LC')
\end{aligned}
$$

where the operator $\mathcal{T}$ is defined as follows.

Definition 10 ($\mathcal{T}$ operator) *Given*

$$A\mathbf{vec}X + B\mathbf{vec}X' = C$$

form $\mathcal{T}$ such that

$$\mathcal{T}(A, B)\mathbf{vec}X = C$$

where the numerical details of $\mathcal{T}$ may be found in [6, App.1].

Now x_3 may be solved for directly as

$$x_3 = \mathbf{vec}\Delta F = (-A_{11}A_{21}^{-1}\hat{A}_{23} - A_{12}A_{21}^{-T}\hat{A}_{33} + A_{13})^{-1}b_1 \qquad (4.2)$$

where computational advantage is gained by inverting smaller matrices. Further computational advantage results from employing sparse matrix algorithms to perform all inversions. At each step, one recovers ΔF by unstacking x_3.

Having obtained the gradient direction (ΔF), the FD algorithm performs a line search along the gradient to minimize the cost at each iteration (*i.e.* it generates $F_{k+1} = F_k + \alpha^* \Delta F$ where α^* minimizes the cost). The line search may be summarized as follows:

1. If one assumes the direction of the gradient to be correct then $\alpha_{min} = 0$. Further since a good approximation (2.29) finds $\alpha \rightarrow 1$, set $\alpha_{max} = 2$.

2. Set $\alpha = (\alpha_{max} + \alpha_{min})/2$.

3. If the differential step size $(\alpha_{max} - \alpha_{min})$ is small or the associated cost differentials $(J(F + \alpha_{max}\Delta F) - J(F + \alpha_{min}\Delta F)$ and $J(F + \alpha\Delta F) - J(F + \alpha_{min}\Delta F))$ are small then break the loop and set α^* equal to the step with the lowest associated cost.

4. If the cost at step size α (*i.e.* $J(F + \alpha\Delta F)$) is less than the cost at either α_{min} or α_{max}, the line search fits the three points and their costs to a quadratic curve. The minimum of this curve $\hat{\alpha}$ (which lies between α_{min} and α_{max}) is expected to be α^*.

   (a) If $J(F + \hat{\alpha}\Delta F)$ is less than or equal to $J(F + \alpha\Delta F)$ then set $\alpha^* = \hat{\alpha}$.

   (b) Else if $J(F + \hat{\alpha}\Delta F) > J(F + \alpha\Delta F)$ and $\hat{\alpha} > \alpha$, then set $\alpha_{max} = \hat{\alpha}$ and return to step (2).

   (c) Else since $J(F + \hat{\alpha}\Delta F) > J(F + \alpha\Delta F)$ and $\hat{\alpha} < \alpha$, set $\alpha_{min} = \hat{\alpha}$ and return to step (2).

5. Else if the cost at step size α_{max} is less than the cost at either α_{min} or α, then the line search chooses $\alpha^* = \alpha_{max}$.

6. Else α^* is assumed to be between α_{min} and α. Thus set $\alpha_{max} = \alpha$ and return to step (2).

The quadratic fit condition is inserted to increase line search efficiency over a brute force binary search which would only terminate at step (3). The checks executed in steps (4a)–(4c) effectively guarantee that cost monotonically decreases from one iteration to the next. How the line search recovers should it encounter an unstable region (*i.e.* if $A - BFC$ has positive eigenvalues) is deferred to the following section.

The FD algorithm conceeds higher computational cost in gradient calculation than do other algorithms which solve the equations of Athans and Levine. However, due to the quality of the FD gradient, the algorithm requires fewer total iterations as opposed to other methods. According to [6], the FD algorithm generally exibits lower total computational cost than other methods.

4.2 Line search technique

This section considers program execution after a gradient has been determined. Specifically two previously omitted details of the line search algorithm are discussed: First, what happens if the maximum step allowed ($\alpha_{max} = 2$) produces an unstable system, and second what happens if the line search returns with a very small or zero minimizing step size?

Instability may be encountered in two ways: first, the maximum step size ($\alpha_{max} = 2$) may yield an unstable system, and second, the minimizing step size at each iteration, α^*, may yield an unstable system. In the case where the maximum step size ($\alpha_{max} = 2$) is unstable, the software retreats geometrically until a stable maximum step is found. A logical retreating ratio seems to be 0.6 (so that the program next checks $\alpha_{max} = 2(0.6) = 1.2$ for stability). This ratio is chosen as a trade off between the expectation of the approximation yielding a minimum cost at $\alpha = 1$ (so one would not suggest a ratio less than 0.5), and the idea that heading toward instability is a generally undesirable thing (so one should choose a small ratio in order to get to the next iteration — and hopefully a better gradient — more quickly). Once the maximum stable step size has been determined, the line search proceeds as before.

Now even though step sizes of α_{min} and α_{max} yield stable systems, no assumptions about the points in between can be made; stability is not a convex property of a SOF gain. [50, Ch.8.5] [17] Thus, as with guaranteed cost decreases at each iteration, stability at each iteration is guaranteed simply by checking for stability at the step size which minimizes cost (α^*) prior to exiting the line search routine. If instability is found, the line search reinitializes with a maximum step size equal to the present (unstable) step size. A similar approach is used in the H_2/H_∞ software to guarantee the closed loop infinity norm constraint at each iteration.

In some instances (for example if the gradient points to a nearby unstable region), the line search routine returns a small step size. Since experience with Newton techniques would suggest a step size of about one in the neigborhood of a solution, one suspects some sort of error — either the quadratic Taylor expansion is not accurate enough for this iteration or some numerical computation errors have produced a poor gradient. Regardless, such pathological behavior requires a fix. Thus our software inspects the cost along a first order gradient direction. The Descent Anderson-Moore (DAM) gradient direction [36] is less sophisticated but more easily calculated than the FD gradient direction, and comes at no added computational cost since it is actually a component of the FD gradient. If the DAM gradient also yields a small step size, then our software inspects the cost in the direction opposite to the FD gradient. The logic here is that through numerical round–off error some gradient element(s) may possess a sign error. Experience shows that the algorithm sometimes finds significant cost decreases in the direction opposite to the FD gradient. If all the directions yield a small step size, the software chooses that which possesses the largest ratio of step size to minimum cost. If all three directions yield a step size of zero, the loop is immediately terminated.

4.3 Initialization and convergence considerations

One conspicuous open problem remaining which would greatly increase the value of the FD algorithm if solved is that of finding an initial stabilizing gain. Much research on these questions has been performed (see the references in [21] for a springboard to the vast body of literature on the topic), but the questions remain unanswered except in a few special cases. Presently, one must adopt or develop some ad hoc technique in an attempt to find a stabilizing SOF gain. One often sucessful technique for finding an initial stabilizing gain F_0 is to attempt

$$F_0 = F_{LQR}C^+ \tag{4.3}$$

where F_{LQR} is the optimal linear quadratic regulator for the controllable pair (A, B), and C^+ is the pseudo–inverse of the observation matrix C. In the case of the FD software, if the simple right inverse formula (4.3) fails to create a stabilizing gain, a shift algorithm is envoked [6, §4.1.4]. Briefly stated the shift algorithm solves the optimal SOF problem for a stable system formed by shifting the poles of the unstable system to the left (accomplished simply by subtracting $\lambda_k I$ from the unstable system $(A - BF_kC)$ where λ_k is the distance the poles have been shifted). The resulting optimal gain for the shifted system is used to initiate another shifted problem (where $\lambda_{k+1} \leq \lambda_k$) until a gain is found which stabilizes the unshifted system or the problem is abandoned as unsolvable. In the case of the H_2/H_∞ software, not only must an initial stabilizing solution be found, but the solution must also satisfy the infinity norm bound constraint on T_{wz_∞} (2.21).

Now that all special run time phenomena have been discussed — and assuming an initial stabilizing gain for the system can be found — the only remaining task is developing termination conditions for the software. Theory suggests that the algorithm sucessfully finds a minimum if the cost gradient $(\partial J/\partial F)$ is zero. Thus one establishes a small cost gradient as one loop termination condition. However, as may be expected after encountering the special cases of the previous sections, one cannot rely on the cost gradient as the sole means of loop termination. Occasions do exist where a large cost gradient nonetheless results in a small step size and/or very little decrease in actual cost (J). Thus in addition to terminating the loop if the norm of the cost gradient is small, the software also terminates the loop if no significant cost decrease has been observed over a period of five iterations. Combined with the cost decreases guaranteed at each iteration by steps (4a)–(4c) in §4.1, and the stability guarantees as discussed in §4.2, this loop termination condition guarantees that the FD software eventually converges.

A minimum in the hand is worth two in an infinite loop.

Chapter 5

Application: LV100 Gas Turbine Engine

As an example of an end–to–end constrained configuration control law design (cccld) procedure, this chapter summarizes a PI control synthesis strategy for the LV100 gas turbine engine. First, §5.1 introduces the LV100 engine and specifies control objectives. Next, §5.2 details the formulation of the cccld problem in the H_2 framework, applying the LFT decomposition results of §3 to recast the problem in a static output feedback framework. In addition, §5.2 discusses techniques for weight selection consistent with meeting system performance objectives. Similarily, §5.3 details the formulation of the cccld problem in the H_2/H_∞ framework. Finally, §5.4 relates the compensator solutions returned by the various problem formulations. As a basis for comparison, another PI controller for the LV100 engine is generated via the method of Edmunds [15]. A quantitative comparison of performance characteristics follows as well as a qualitative comparison of the control design techniques.

5.1 The LV100 engine model and performance specifications

The LV100, a next generation tank engine currently under development at GE, is presented in schematic form in Figure 5.1. Key features to note from this figure include the two engine spools connected to the turbines and the recuperator inserted in the airflow path. The high pressure turbine drives the compressor while the low pressure turbine couples to the vehicle transmission. The recuperator improves thermodynamic efficiency. From a controls point of view, the objective is to vary main fuel flow (W_f) and the variable area turbine nozzle (V_{ATN}) to regulate the speeds of the gas generator spool (N_g) and the power output to the vehicle (N_p) while ensuring that internal constraints on temperatures and stall margins are not violated. This thesis addresses the regulation aspect of the control problem. Internal temperature and stall margin constraints tend to be peak constraints for which ℓ^1 control methods are more appropriate [11].

At the operating point of interest, the LV100 may be modeled as a five state, two input, two output minimum phase system with inputs $u = [\ W_f\ V_{ATN}\ ]'$ and outputs $y_P = [\ N_g\ T_6\ ]'$. One

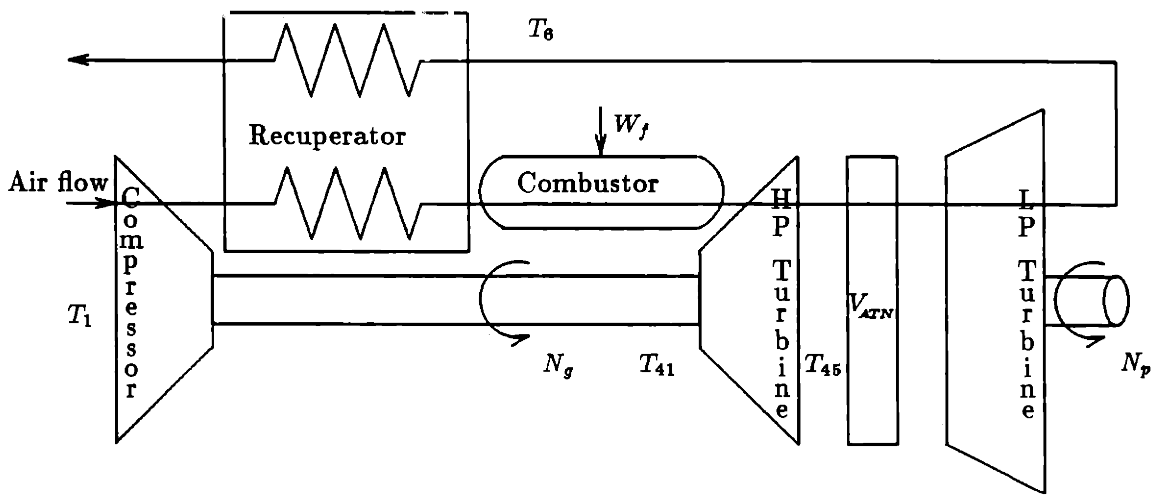


Figure 5.1: LV100 schematic diagram.

normalized state space representation of the LV100 gas turbine plant finds

$$A_p = \begin{bmatrix} -1.4122 & -0.0552 & 0 & 42.9536 & 6.3087 \\ 0.0927 & -0.1133 & 0 & 4.2204 & -0.7581 \\ -7.8467 & -0.2555 & -3.3333 & 300.4167 & -4.4894 \\ 0 & 0 & 0 & -25.0000 & 0 \\ 0 & 0 & 0 & 0 & -33.3333 \end{bmatrix}, \quad B_p = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$C_p = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad D_p = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

with the state vector $x_p = [\ N_g\ N_p\ T_6\ x_{W_f}\ x_{V_{ATN}}\ ]'$ where x_{W_f} and $x_{V_{ATN}}$ are states associated with the actuation of W_f and V_{ATN} respectively. Analysis of the engine's open loop singular values shown in Figure 5.2 reveals that the LV100 model exibits second order roll-off at frequencies above 10r/s. Placing the engine and PI controller in a standard command tracking loop as in Figure 5.3, one expects the achievable closed loop bandwidth to be limited by the plant roll-off. This thesis therefore attempts a design achieving a 1r/s closed loop bandwidth for command tracking which corresponds to about a 2.2s step response rise time using standard rules of thumb. Now consider the problem formulations required to produce PI controllers which satisfy the performance objectives.

Remark 16 *Assuming the command tracking loop of Figure 5.3 outputs performance variables yp*

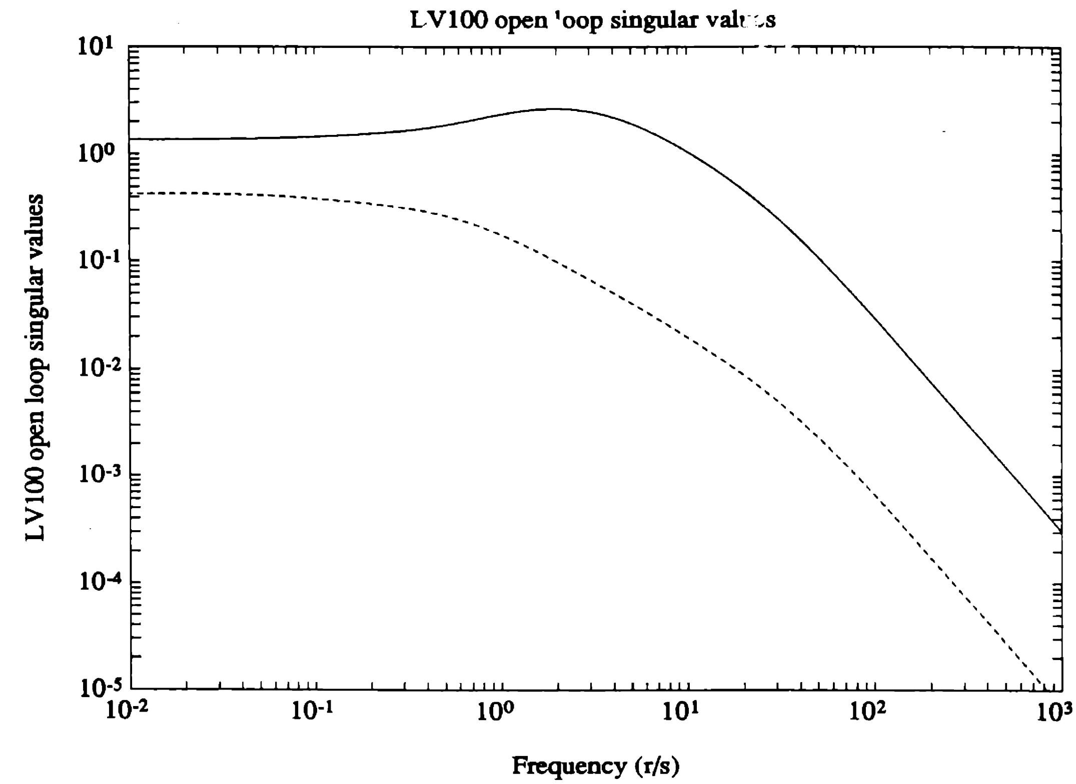


Figure 5.2: Open loop singular values of LV100 gas engine.

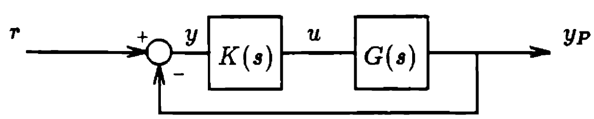


Figure 5.3: Standard command tracking loop

and u, it may be represented as $P_{CTL} * K$ *where*

$$P_{CTL}(s) \longleftrightarrow \begin{bmatrix} 0 & \vdots & G(s) \\ 0 & \vdots & I_2 \\ \cdots & \cdot & \cdots \\ I_2 & \vdots & -G(s) \end{bmatrix} \tag{5.1}$$

and where the inputs to P_{CTL} *are* $[\ r'\ u'\ ]$.

5.2 H_2 problem formulation

The goal of this section is to formulate a command tracking problem consistent with the requirements of Levine and Athans and consistent with performance objectives. In other words, one wishes to

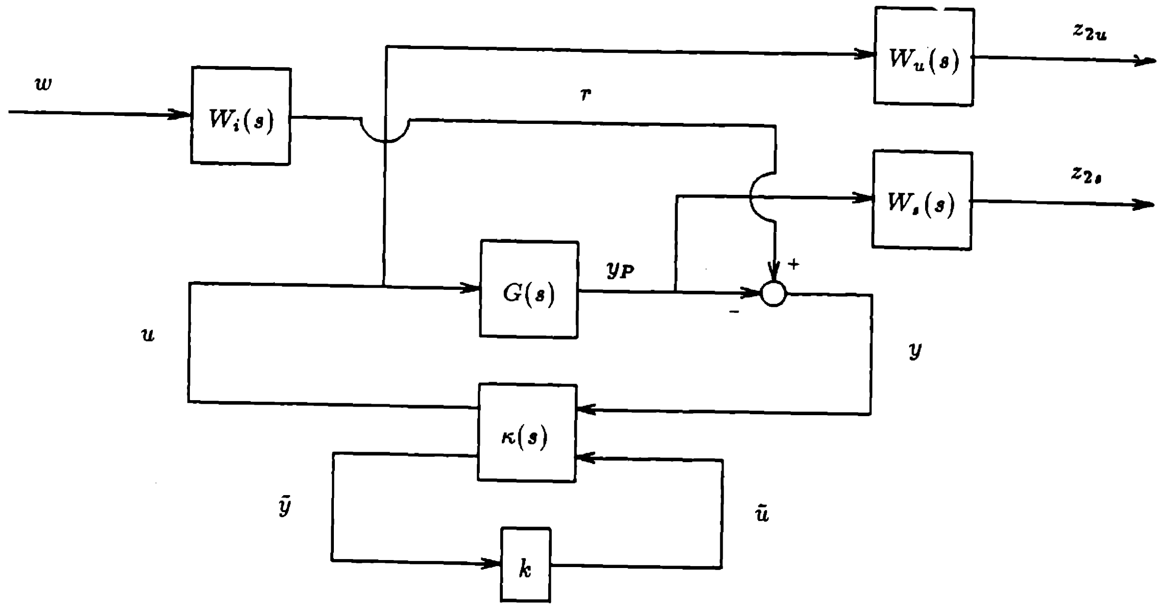


Figure 5.4: Weighted command tracking loop.

develop a technique for appropriately choosing Q, R, and P in (2.17)-(2.19) so that the optimization problem has significance with respect to the performance objectives.

Towards this end, this thesis proposes augmenting the control loop with the command weighting function $W_r(s)$, the controls weighting function $W_u(s)$, and the complementary sensitivity weighting function $W_s(s)$ as shown in Figure 5.4. The reasons for establishing such weights and the actual weighting functions used for the LV100 example will be discussed after examining the structure of the resulting static output feedback control configuration. Recall that §3.3 shows how to perform an LFT decomposition of a PI controller.

5.2.1 Control structure analysis

Following the roadmap of §1.3 and Figures 1.1–1.2, one first reformulates Figure 5.4 as a standard output feedback configuration with the TITO system given by

$$\begin{bmatrix} z_{2s} \\ z_{2u} \\ \cdots \\ y \end{bmatrix} = P(s) \begin{bmatrix} w \\ \cdots \\ u \end{bmatrix} \tag{5.2}$$

$$= \; W_o(s)\, P_{CTL}(s)\, W_i(s) \begin{bmatrix} w \\ \cdots \\ u \end{bmatrix} \tag{5.3}$$

$$= \begin{bmatrix} W_s(s) & & \\ & W_u(s) & \\ & & I_2 \end{bmatrix} \begin{bmatrix} 0 & G(s) \\ 0 & I_2 \\ I_2 & -G(s) \end{bmatrix} \begin{bmatrix} W_r(s) & 0 \\ 0 & I_2 \end{bmatrix} \begin{bmatrix} w \\ \cdots \\ u \end{bmatrix} \tag{5.4}$$

$$= \begin{bmatrix} 0 & \vdots & W_s(s)\, G(s) \\ 0 & \vdots & W_u(s) \\ \cdots & \cdot & \cdots \\ W_r(s) & \vdots & -G(s) \end{bmatrix} \begin{bmatrix} w \\ \cdots \\ u \end{bmatrix}. \tag{5.5}$$

Recall that for H_2 problems in general there can be no direct feedthrough from the external input w to the performance variables z_{2s} and z_{2u}, as otherwise the cost would be infinite. Presently, the strictly proper nature of $G(s)$ ensures that there is no direct feedthrough from w to z_{2s} or from u to y_P. In order to produce zero feedthrough from w to z_{2u} while allowing for a non–strictly proper controller (*i.e.* direct feedthrough from y_P to u) requires that the command weighting function, $W_r(s)$, be strictly proper. Assuming this, a state space representation of the TITO system $P(s)$ will be of the form

$$\begin{bmatrix} \dot{x} \\ \hline z_2 \\ \cdots \\ y \end{bmatrix} = \left[\begin{array}{c|ccc} a & b_1 & \vdots & b_2 \\ \hline c_1 & 0 & \vdots & d_{12} \\ \cdots & \cdots & \cdot & \cdots \\ c_2 & 0 & \vdots & 0 \end{array}\right] \begin{bmatrix} x \\ \hline w \\ \cdots \\ u \end{bmatrix} \tag{5.6}$$

where $x = [\; x_r' \; x_p' \; x_s' \; x_u' \;]'$ and $z_2 = [\; z_{2s}' \; z_{2u}' \;]'$.

Decomposing the PI controller as in §3.3.1 and then combining $P(s)$ with the dynamics portion of the compensator, $\kappa(s)$ using the star product operator produces a static output feedback configuration where the TITO plant, $\hat{P}(s)$, has the form

$$\begin{bmatrix} \dot{x}_t \\ \hline z_2 \\ \cdots \\ \tilde{y} \end{bmatrix} = \left[\begin{array}{c|ccc} A & D_1 & \vdots & B \\ \hline E_1 & 0 & \vdots & E_2 \\ \cdots & \cdots & \cdot & \cdots \\ C & 0 & \vdots & 0 \end{array}\right] \begin{bmatrix} x_t \\ \hline w \\ \cdots \\ \tilde{u} \end{bmatrix} \tag{5.7}$$

with $x_t = [\; x' \; x_c' \;]'$. The zero transmissions arise as a result of the star product, but they also make sense intuitively: there is still no transmission from w to z_2 as before; there is no transmission from w to $\tilde{y}$ because $W_r(s)$ is assumed strictly proper; and there is no transmission from $\tilde{u}$ to $\tilde{y}$ because that is a property of the LFT decomposition (3.19).

One may recognize (5.7) as resembling the formulation of Bernstein, Haddad, and Nett (see Eq. 2.20) without the infinity constraint. Next, one might expect to form $Q = E_1'E_1$, $R = E_2'E_2$,

$P = D_1 D_1'$ as weights for the system given by the triple (A, B, C). Obviously, Q, R, and P are symmetric and positive semidefinite, and R will be positive definite as long as E_2 has full column rank. However, a well–posed problem also requires the pair (A, E_1) to be observable [25], and it is easily shown that in the present case this requirement is violated, the unobservable states being associated with the compensator's integrators due to the fact that $\bar{a} = 0$ and $\bar{c} = 0$ for a PI controller (see 3.20)). Thus in addition to providing the actual frequency weights used in the LV100 design problem, the next section considers what further requirements must be placed on Q, R, and P in order to create a well–posed problem and how to iteratively adjust the weights to perform loop shaping.

5.2.2 Controller weight selection

With the structural insight gained above, it is time to discuss techniques and requirements for choosing the frequency weighting functions and define the weighting matrices Q, R, and P. As mentioned above, the command weighting function $W_r(s)$ must be strictly proper to guarantee a well–posed problem. At the same time, the dynamics of $W_r(s)$ should not effect optimization results. Thus choose a first order lag such as

$$W_r(s) = \frac{10}{s+10} I_2 \tag{5.8}$$

which has a break frequency sufficiently higher than the desired bandwidth of the system.

With the complementary sensitivity weighting function $W_s(s)$ one wishes to accomplish two things: weight the plant output with the inverse of the desired closed loop response and emphasize low frequency behavior and penalize high frequency behavior. Thus choose

$$W_s(s) = \frac{10(s+1)^2}{(s+10)} \begin{bmatrix} \rho & 0 \\ 0 & 1 \end{bmatrix} \tag{5.9}$$

where the $(s+1)^2$ factor is the inverse of the desired frequency response and ρ is a parameter used for loop shaping initially set to one. Note that the $10/(s+10)$ term, while emphasizing low frequencies, also ensures that the $W_s(s)G(s)$ is strictly proper from u to z_{2s}. Incidently, the fact that $W_s(s)$ is not realizable by itself a problem since the closed loop transfer function $P(s)$ (5.5) depends on $W_s(s)G(s)$ which is strictly proper.

Finally one chooses the controls weighting function

$$W_u(s) = \frac{s+1}{s+10} I_2 \tag{5.10}$$

to penalize high frequency control.

Recalling that $x_t = [\, x_r' \; x' \; x_s' \; x_u' \; x_c \,]'$ it is evident that, with the weights posed above. the system $\hat{P}$ (5.7) has 13 states.

With the above frequency weighting resulting in a formulation of $\hat{P}$ as in (5.7), form $P = B_1 B_1'$. This effectively eliminates r from the formulation and guarantees that P is positive semidefinite and symmetric. Similarily form $R = rE_2'E_2$ where r is a parameter used for loop shaping initially set to one. Note that R will be positive definite as required as long as E_2 is has full column rank. As mentioned above, defining $Q = E_1'E_1$ results in an ill posed problem because the controller states are not observable: indeed

$$E_1 = [\, E_{t-c} \; 0_c \,]$$

where 0_c is a zero matrix with as many columns as there are integrator states (2 in our case). Thus consider forming

$$E_{1a} = \begin{bmatrix} E_{t-c} & 0_c \\ 0_{t-c} & q_c I_2 \end{bmatrix} \tag{5.11}$$

where q_c is a parameter used for loop shaping and is initially set to one. Next form $Q = E_{2a}'E_{2a}$.

Remark 17 *Notice that E_2 may be augmented with rows of zeros without affecting it's column rank or problem formulation. In formulations which include cross weighting between system state and control, one performs such an augmentation to ensure dimensional compatibility. Such is the case with BHN's H_2/H_∞ formulation discussed in the next section.*

Now that the problem formulation is complete and includes parameters available for loop shaping, one may use the LFT decomposition technique descibed in §3 and the feedback descent software described in §4 to generate controllers to meet certain performance specifications. First increase q_c until bandwidth constraints are met and then tune ρ and r to adjust such things as overshoot and cross channel coupling. For the LV100 design problem at hand, parameter values of

$$\begin{aligned} q_c &= \sqrt{1000} \\ r &= 0.2 \\ \rho &= 0.15 \end{aligned} \tag{5.12}$$

produced a controller which met the performance objectives set in §5.1 (see §5.4 for more details).

5.3 H_2/H_∞ problem formulation

As with the H_2 formulation, the H_2/H_∞ problem (2.20), (2.26)-(2.28) may be augmented with frequency weights so that the optimization problems acquires significance in relation to the performance objectives. Recalling that the optimization will minimize the 2-norm of T_{wz_2} while ensuring that

$\| T_{wz_\infty} \|_\infty$ remains upper bounded by some constant γ, one expects to formulate the H_2 portion of the problem exactly as in §5.3 (with the additional consideration of Remark 17).

Assuming system uncertainties can be modeled as a bounded function from w to z_∞, then one may impose an H_∞ constraint on the control design, which — if satisfied — will guarantee system robustness in the face of those uncertainties [62]. In our case, high frequency unmodeled dynamics are assumed present in the loop complementary sensitivity T_{wy_P}; uncertainty at low frequency is assumed small. Thus for the H_∞ portion of the control design, a constraint will be imposed on T_{wy_P}. Thus set

$$E_{1\infty} = [\, 0_2 \;\; C_P \;\; 0_{2\times 6} \,]$$

and $E_\infty = E_{2\infty} = 0_2$.

The first remaining concern is whether the weightings as mentioned in the previous two items are consistent with the requirements of BHN. Specifically, are the control weights for LQG and H_∞ multiples of each other (*i.e.* do (2.23)–(2.24)

$$\begin{aligned} R_2 &= \alpha^2 R \\ R_{2\infty} &= \beta^2 R \end{aligned}$$

hold). In our case, $R_{2\infty} = 0$ so that $\beta = 0$, $\alpha = 1$. Another remaining concern is finding an initial stabilizing gain F_0 which also produces a closed loop infinity norm less than the desired γ.

5.4 Results

For comparative purposes we generate, via the method of Edmunds, another PI controller for the LV100 engine. For a more thorough explanation of Edmunds method, we direct the interested reader to either Edmunds paper [15] or to [35, Ch. 7]. For our present needs, summarizing the basic design philosophy of Edmunds' method will suffice. In general, it produces a controller, $K_{LS}(s)$, that yields the best closed loop approximation to a desired response $H_D(s)$ in a least squares sense. Specifically, assume a model matching configuration as shown in Figure 5.5 with the desired response $H_D(s)$ and poles of $K_{LS}(s)$ chosen by the designer. Specifying a number of frequency points of interest, one can generate an overconstrained linear equation with the numerator coefficients of the compensator unknown. Of course as with any control method, the designer must still know a great deal about the plant to be controlled and about the closed loop system's acheivable performance given the controller constraints. In the case of Edmunds' method, choosing an $H_D(s)$ with unachievable bandwidth generally produces lousy results. One advantage of Edmunds' method for low order control design is the relative ease with which problems may be formulated once achievable performance is known.

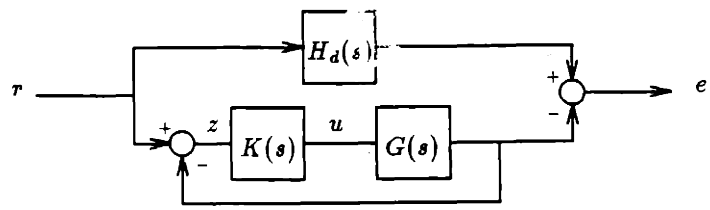


Figure 5.5: Model matching configuration of Edmunds.

However disadvantage include 1) closed loop stability is NOT guaranteed, 2) compensator poles must be chosen apriori, and 3) approximations are NOT guaranteed at frequencies other than those chosen (although one can argue using "smoothness"). Still in many cases Edmunds method produces satisfactory results; in our case we use it as a baseline to which control designs produced by H_2 SOF and H_2/H_∞ methods are measured.

5.4.1 Edmunds' controller

Employing Edmunds method with a desired response of

$$H_D(s) = \frac{1}{s^2 + 1.6s + 1} I_2 \tag{5.13}$$

approximated at frequency points logarithmically evenly spaced between 0.01r/s and 100r/s yields the PI controller given by

$$K_{LS}(s) \longleftrightarrow \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 0.3300 & 0.1384 & -0.0223 & -0.0047 \\ 1.0292 & -1.4083 & 1.9635 & 0.3305 \end{array}\right] \tag{5.14}$$

which does stabilize the system.

Figure 5.6 shows closed loop singular value plots and system step responses. As the singular value plot shows, the design meets bandwidth requirements. The system exibits approximately a first order response: hence little or no overshoot but only modest rise and settling time. Cross channel coupling, although not unacceptable, is nonetheless pronounced, and efforts to increase the bandwidth of $H_D(s)$ resulted in increased coupling effects.

5.4.2 H_2 controller

In contrast to Edmunds method, one formulates an H_2 SOF optimization problem as in §5.2 above and uses the coupled Ricatti/Lyapunov solving technique as described in §2.4 and the software as

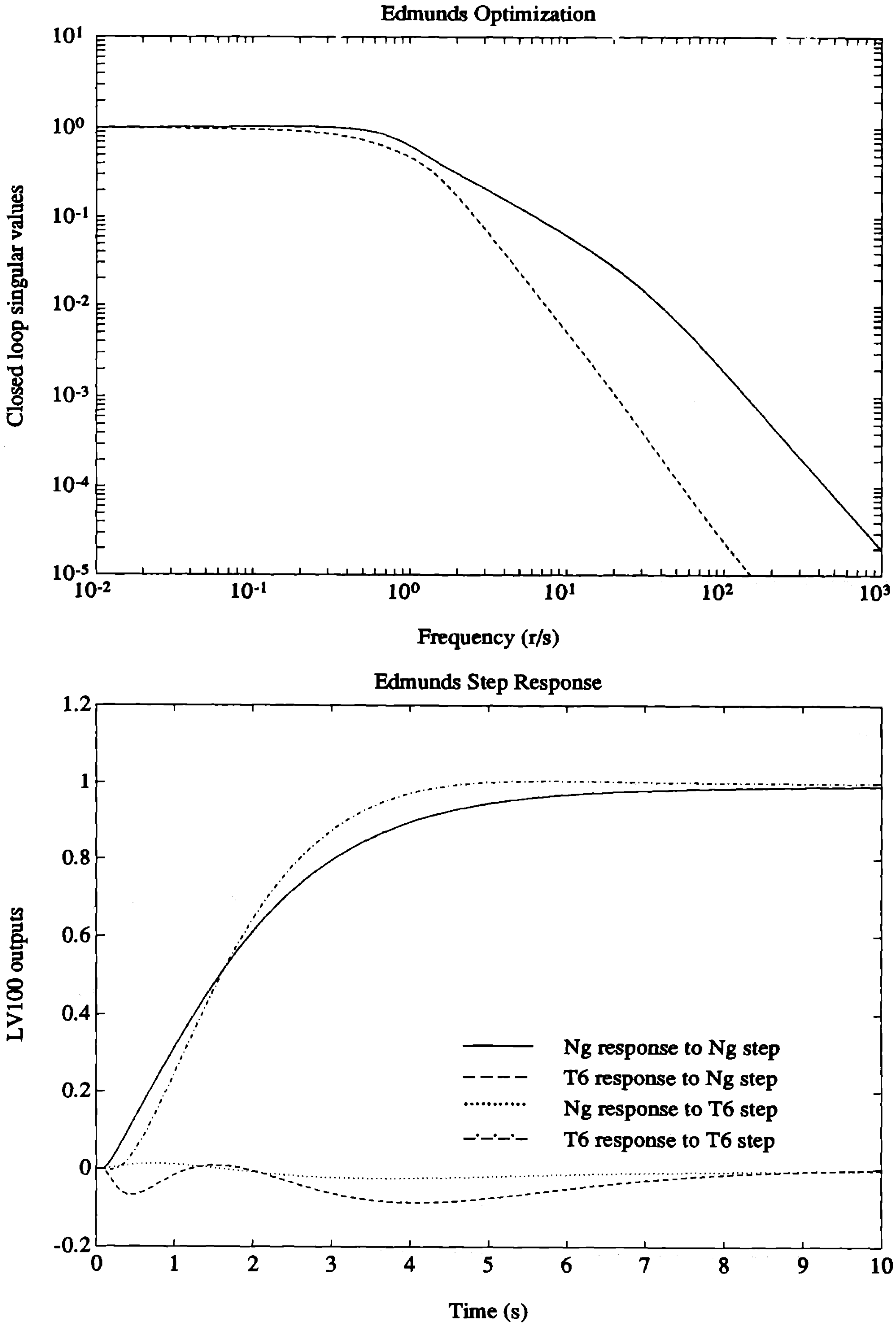


Figure 5.6: Edmunds controller.

described in §4. If the algorithm is initiated with an F_0 as in (4.3), it will terminate after 10 iterations with a cost gradient norm of $\| \partial J/\partial F \| = 3.041 \times 10^{-7}$, outputting

$$K_{H_2}(s) \longleftrightarrow \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1.1488 & 0.5058 & 0.0985 & 0.0302 \\ 3.2494 & -5.0084 & 7.4335 & -0.4575 \end{array}\right]. \tag{5.15}$$

which has a cost of $J = 859.9$.

Figure 5.7 shows closed loop singular value plots and system step responses. One notes that the design more than achieves its specified bandwidth with rise times on the order of one second and the second channel exibiting some overshoot. Cross channel coupling, while still present, has magnitude slightly less than that of the Edmunds' controller and settles out more quickly (3s versus 7s). The closed loop infinity norm of the system is 1.04. Although not shown, the H_2 design requires greater control action than the design produced by Edmund's method. Should the design be judged unfeasible due to actuator saturation, one might increase r and attempt another design.

5.4.3 H_2/H_∞ controller

Next one formulates an H_2/H_∞ optimization problem as in §5.3 above and uses a modification of the feedback descent technique discussed in §2.4 to solve the optimality equations (2.26)-(2.28). Initiating the system with

$$F_0 = \begin{bmatrix} -0.3927 & -0.4807 & -0.0336 & -0.0470 \\ -0.2243 & 0.3772 & -0.0267 & 0.0366 \end{bmatrix} \tag{5.16}$$

produces a closed loop H_∞ norm of 1.03. Constraining the optimal H_∞ norm to be less than 1.1 yields the controller

$$K_{H_2/H_\infty}(s) \longleftrightarrow \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1.6911 & 0.4965 & 0.1535 & 0.0272 \\ 4.0119 & -4.9771 & 11.7787 & -0.6102 \end{array}\right] \tag{5.17}$$

which has an H_∞ norm of 1.02. This solution is reached after 12 iterations. Although the cost gradient norm is $\| \partial J/\partial F \| = 10.99$ no further cost decreases beyond $J = 730.8$ can be found. Differences in the optimal gain and minimum cost as compared with the H_2 result are assumed to be caused by the inclusion of cross weighting terms (between system state and input) in the H_2/H_∞ formulation which are not included in the H_2 formulation (although they certainly may be).

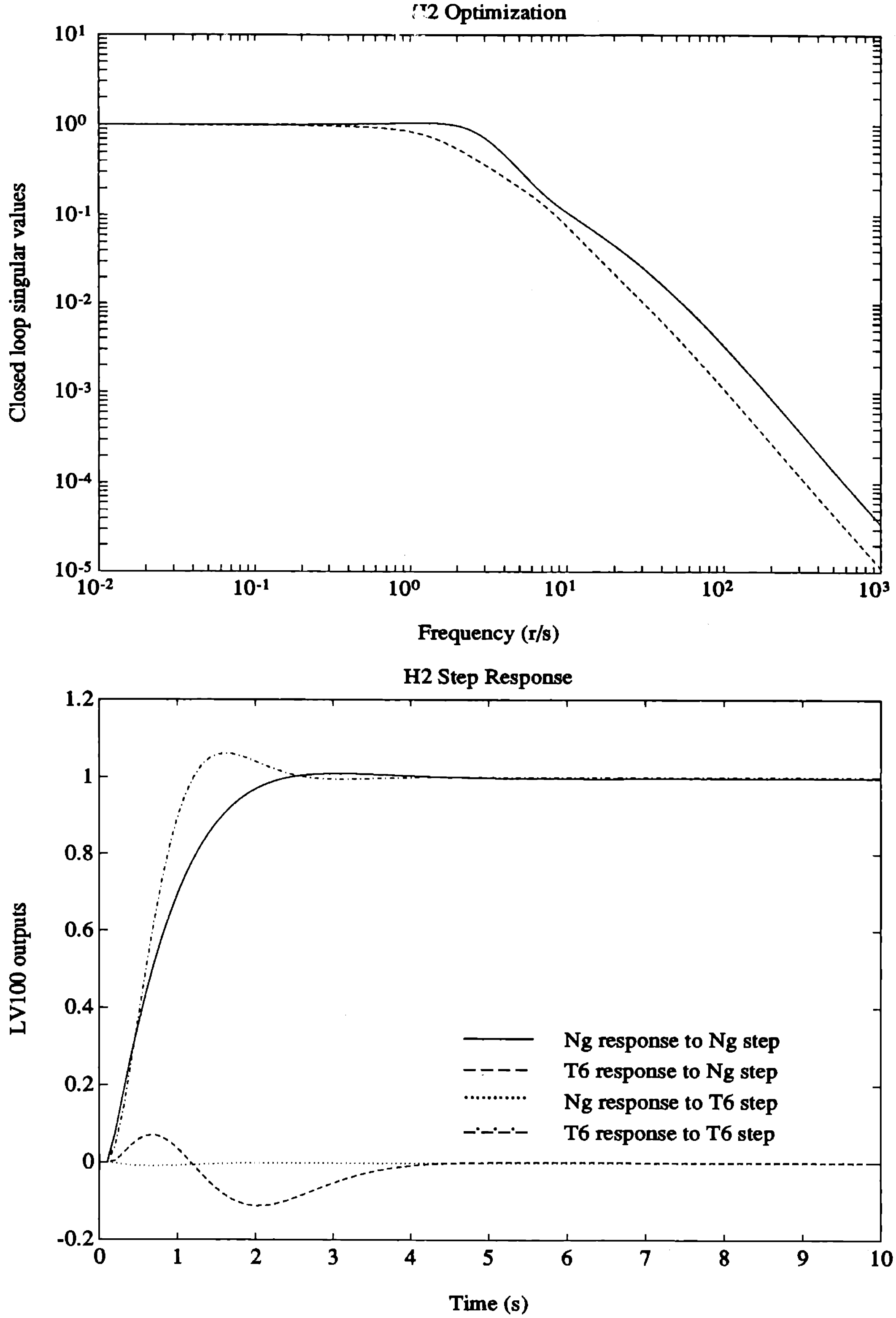


Figure 5.7: Optimal H_2 controller.

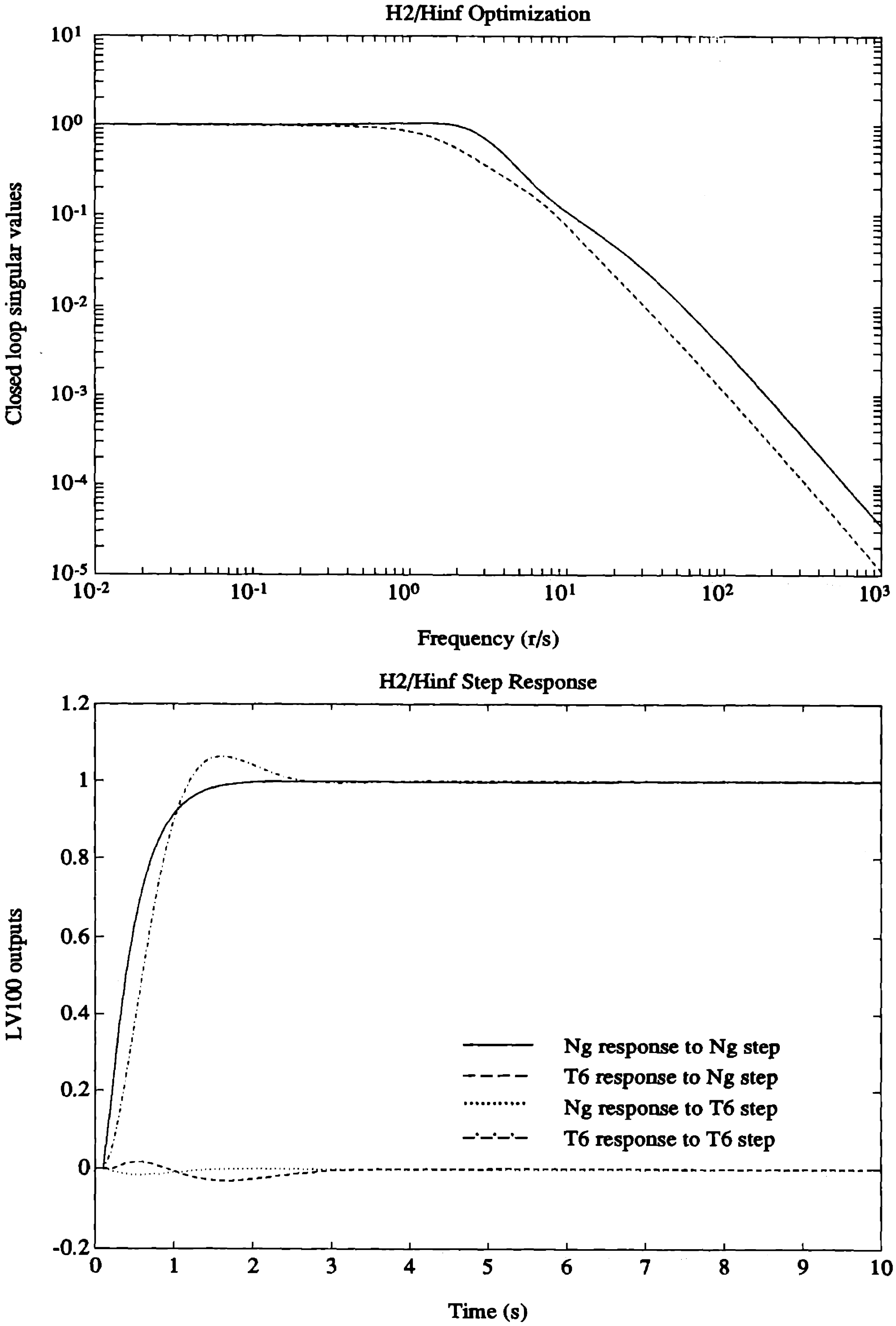


Figure 5.8: Optimal H_2/H_∞ controller.

Figure 5.8 shows closed loop singular value plots and system step responses. As in the H_2 formulation, the system exibits superb bandwidth and response. In addition, note that the cross channel coupling is further decreased; thus it appears as though cross weighting terms provide a distinct advantage in attenuating such disturbances. As may be expected from the higher gains of K_{H_2/H_∞}, this design requires greater control action than either the Edmund's or H_2 design.

5.4.4 Other observations

As discussed in §4, the FD algorithm is susceptible to local minima and sometimes exibits pathological behavior. Here we provide a few examples of how sensitive the algorithm can be to changes in the initializing gain or changes in weighting parameters q_c, r, and ρ. First consider verifying that the coupling between state and control weights is indeed the reason for the diference between the H_2 and H_2/H_∞ solutions. By initializing the H_2 (uncoupled) formulation with K_{H_2/H_∞}, one finds the cost to be $J = 942.6$ (re: with coupling, the cost is $J = 730.8$). After four iterations, the algorithm converges to K_{H_2} (within five significant digits — the differnce results in the cost gradient now being $\| \partial J/\partial F \| = 5.2 \times 10^{-8}$). Thus we conclude that the coupling is indeed the reason for the difference between the two solutions. Conversely, initializing the H_2/H_∞ (coupled) formulation with K_{H_2}, one finds the cost to be $J = 759.8$ (re: without coupling, the cost is $J = 859.8$). After four iterations, the algorithm terminates due to zero step size to a gain which is different than K_{H_2/H_∞} and which has a cost $J = 728.3$. However the solution's closed loop singular values and step and control responses match those of K_{H_2/H_∞} almost exactly.

A curious result is obtained when H_2 is initiated with F_0 from (5.16) and $r = 1$. After nine iterations, the algorithm terminates with a cost $J = 5329.1$ because the FD gradient is headed toward instability (and the DAM and minus FD directions cannot improve cost). The resulting closed loop step responses exibit rise times on the order of 10s and unacceptable cross coupling. Curiously, if one now sets $r = 0.2$, the algorithm will proceed as before for eight iterations, but will then recover finally converge after 16 iterations. The solution is not equal to K_{H_2} and has a cost $J = 1006.6$. Although the closed loop singluar values of this solution look like those exibited by K_{H_2}, the step response shows more cross coupling than before and less control action.

Such convergence results suggest that one should initialize the FD algorithm with a number of different gains before concluding that a given control configuration or weighting functions cannot achieve the desired system performance.

Invention is 1% in inspiration and 99% software debugging.

Chapter 6

Thesis Summary

This thesis considered both theoretical and practical issues associated with the entire cccld method proposed by BHN including (1) problem setup and formulation, (2) data reduction to SOF, (3) solution of SOF, and (4) practical applications. On the theoretical side, it showed how a large class of constrained configuration control problems can be reformulated as static output feedback (SOF) problems for which optimality conditions have been previously derived. On the practical side, this thesis contributed to the development of a software algorithm for iteratively solving SOF optimality equations. Also this thesis developed a weighting scheme to aid designers in using the method to synthesize controllers for command tracking applications, and verified the SOF control design procedure via application to a gas turbine engine.

A more detailed review of thesis results may be found in §6.1. Recommendations for future research may be found in §6.2.

6.1 Review of thesis results

This thesis contributed to the development of a unified software solution to a class of constrained configuration control law design (cccld) problems. The method assumes a fixed order but otherwise arbitrarily constrained controller, and poses the problem as an H_2 minimization problem (or H_2 minimization problem subject to a maximum H_∞ cost) in a standard output feedback configuration. It then transforms the constrained dynamic optimization problem to a possibly block decentralized static output feedback problem for which necessary conditions for solution are known. Solving the resulting equations however presently requires iterative numerical algorithms.

The main theoretical contribution of this thesis was the development of a technique for decomposing an arbitrarily constrained dynamic system into an LFT of a fixed dynamic system and a static parameter matrix. Thus a dynamic compensation problem may be recast as a static output feedback problem. The decomposition assumes that the elements of the state space matrices of the controller can be expressed as at most a sum of a fixed value and a tunable parameter. This yields an intermediate result where the parameter matrix may have some of its elements constrained to be zero. A multiplicative reduction of the intermediate parameter matrix and subsequent augmentation of the associated intermediate fixed dynamic system results in an LFT of the desired form. The dynamics portion of the LFT decomposition is characterized by

- dynamic order equal to that of the original controller, and
- no direct feedthrough in the "22" channel parameters matrix.

This thesis provided examples showing the easily automated technique to be applicable to decompositions which would be cumbersome to perform by hand with an ad hoc technique. It also showed how the technique can reproduce a previous result concerning SOF fixed order control formulations which use canonical form representations. A further extension of the decomposition determined how to minimize the occurances of repeated parameters in some problems.

The thesis reviewed an iterative algorithm for solving coupled optimality equations which arise in SOF problem formulations. The algorithm, the feedback descent method, uses Newton expansions, Kronecker products, and sparse matrix techniques to directly and efficiently compute a cost gradient direction for each iteration. Given the gradient direction, this thesis developed a modified binary line search for greater efficiency. Further enhancements aid in ensuring the convergence of the algorithm.

The thesis developed a frequency weighting technique for command tracking problems to accompany the SOF design procedure so that system performance objective may be specified in terms of desired loop shapes. The technique resembles the weighting characteristic of LQG and H_∞ problems. For non–strictly proper controllers, one must also introduce a low pass weight at the commanded inputs to satisfy direct feedthrough constraints imposed by the SOF formulation. The designer ends up adjusting weights to vary such system performance measures as rise times, overshoots, and bandwidth. Other weight adjustments were found which appear to allow trade–offs in cross channel coupling.

Finally, the thesis integrated the entire cccld into a unified design procedure, and showed the design procedure to be effective for synthesizing a PI controller for a jet engine.

6.2 Recommendations for future research

Recommendations for future research regarding cccld may be divided into those that are incremental improvements on the approach explored here and those that are different from the approach explored here. Incremental improvements include further

- development of necessary conditions for different system configurations and different cost criterion,

- development of necessary conditions for less restrictive assumptions, for example the requirement that $\alpha^2 R_{1\infty} = \beta^2 R_{2\infty}$ in the H_2/H_∞ formulation (2.23-2.24),
- comparison to other cccld methods, and
- enhancements to FD algorithm.

Different system configurations of interest include H_2/H_∞ (block) decentralized control problems or systems with repeated parameter problems.

As mentioned in the introduction, other methods of generating constrained configuration controllers exist. In this thesis, comparison was only made to the model matching method of Edmunds. Further comparison might be made to the H_∞–balanced truncation of Mustafa and Glover [45, 47] or the goal attainment methods of Grace [19]. Enhancements to the FD algorithm might include incorporating more general formulations: the (block) decentralized H_2 problem need only be coded [6] while a FD algorithm for reduced order H_∞ control needs to be developed from existing necessary conditions and then coded. Alternately, the software might be further streamlined to increase computational efficiency. Additionally, more applications experience might produce more insight into loop shaping and weighting techniques.

New directions for research include developing

- different solution algorithms for the SOF optimality equations, for example homotopy algorithms [39, 4, 3, 54, 56, 48],
- closed–form solutions for H_2 or H_2/H_∞ cccld problems,
- different cccld cost formulations or solution techniques, and
- symbolic generation/manipulation of necessary conditions.

Different cost formulations include for example ℓ^1 formulations [11] and goal attainment formulations [19]. As opposed to using Lagrangian techniques to derive SOF optimality conditions, different solution techniques might produce equations that are both necessary and sufficient.

Since discovery of closed–form solutions seem unlikely if not impossible, more fruitful results will more likely be obtained in the area of incremental improvements. Of course any results on the existence of stabilizing solutions to SOF problems would be a major contribution, but again such results do not seem eminent.

When all is said and done, your advisor says it's done.

Bibliography

[1] M. Athans and F. C. Schweppe, "Gradient matrices and matrix calculations," *M.I.T. Lincoln Laboratory, Technical Note 1965-53*, Lexington, MA, November, 1965.

[2] S. Basuthakur and C. H. Knapp, "Optimal Constant Controllers for Stochastic Linear Systems," *IEEE TAC*, AC-20, 1975, 664-6.

[3] Dennis S. Bernstein, Principal Investigator, *OPUS: Optimal Projection for Uncertain Systems, final report*, Harris Corporation, Melbourne, FL, October 1988.

[4] Dennis S. Bernstein and Wassim M. Haddad, "LQG control with an H_∞ performance bound: a Ricatti equation approach," *IEEE TAC*, AC-34, 1989, 293-305.

[5] D. S. Bernstein, W. M. Haddad, and C. N. Nett, "Minimal complexity control law synthesis, part 2: Problem solution via H_2/H_∞ optimal static output feedback," in *Proc. ACC*, June 1989, 2506-11.

[6] Jan W. Beseler, *An Implementation of Newton's Method for the Optimal Output and Decentralized Feedback Problems*, Master's Thesis, Rensselaer Polytechnic Institute, 1990.

[7] Jan Beseler, Joe H. Chow, and K. D. Minto, "A feedback descent method for solving constrained lqg control problems," in preparation.

[8] P. J. Blanvillain and T. L. Johnson, "Invariants of optimal minimal-order observer-based compensators," *IEEE TAC*, AC-23, 1978, 473-4.

[9] S. Boyd, V. Balakrishnan, and P. Kabamba, "On conputing the H_∞ norm of a transfer function," in *Proc. ACC*, Atlanta, GA, 1988, 396-7.

[10] John W. Brewer, "Kronecker products and matrix calculus in system theory," *IEEE CAS*, CAS-25, 1978, 772-781.

[11] Munther A. Dahleh, *Design of Multivariable Feedback Controllers: ℓ^1-Optimal Systems*, Ph.D. Thesis, Rice University, 1987.

[12] R. Lane Dailey, *Lecture Notes for the Workshop* ι~ *H_∞ and μ Methods for Robust Control,"* 1990 ACC, San Diego, CA, May 1990.

[13] John Doyle, *Lecture Notes for ONR/Honeywell Workshop on Advances in Multivariable Control*, Minneapolis, 1984.

[14] John C. Doyle, Keith Glover, Pramod P. Khargonekar, and Bruce A. Francis, "State-space solutions to standard H_2 and H_∞ control problems," *IEEE TAC*, AC-34, 1989, 831-47.

[15] J. M. Edmunds, "Control system design and analysis using closed-loop Nyquist and Bode arrays," *Int. J. Control*, Vol. 30, 1979, 773-802.

[16] Dale F. Enns, "Model reduction with balanced realizations: an error bound and a frequency weighted generalization," in *Proc. CDC*, December 1984, 127-30.

[17] Adly T. Fam and James S. Meditch, "A canonical parameter space for linear system design," *IEEE TAC*, AC-23, 1978, 454-8.

[18] Israel Gohberg and Sorin Rubinstein, "Cascade decompositions of rational matrix functions and their stability," *Int. J. Control*, Vol. 46, 1987, 603-29.

[19] A.C.W. Grace, *Computer-Aider Control System Design Using Optimization Methods*, Ph.D. thesis, University of Wales, Bangor, England, 1989.

[20] Andrew Grace, personal communication.

[21] Guoxiang Gu, "Stabilizability Conditions of Multivariable Uncertain Systems via Output Feedback Control," *IEEE TAC*, AC-35, 1989, 925-7.

[22] W. J. Hutcheson, "A simple derivation of the gradient conditions for optimal constant output feedback," *IEEE TAC*, AC-23, 1978, 937-8.

[23] David C. Hyland and Dennis S. Bernstein, "The optimal projection approach to model reduction and the relationship between the methods of Wilson and Moore," in *Proc. CDC*, Las Vegas, 1984, 120-126.

[24] David C. Hyland and Dennis S. Bernstein, "The optimal projection equations for fixed-order dynamic compensation," *IEEE TAC*, AC-29, 1984, 1034-7.

[25] Timothy L. Johnson and Michael Athans, "On the design of optimal constrained dynamic compensators for linear constant systems," *IEEE TAC*, AC-15, 1970, 658-660.

[26] Thomas Kailath, *Linear Systems*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1980

[27] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, March 1960, 35-45.

[28] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *Journal of Basic Engineering*, March 1961, 95-108.

[29] C. H. Knapp and S. Basuthakar, "On optimal output feedback," *IEEE TAC*, AC-17, 1972, 823-5.

[30] Friedrich S. Kramer and Anthony J. Calise, "Fixed-order dynamic compensation for multivariable linear systems," *J. Guidance*, Vol. 11, 1988, 80-5.

[31] William S. Levine, Timothy L. Johnson, and Michael Athans, "Optimal limited state variable feedback controllers for linear systems," *IEEE TAC*, AC-18, 1971, 785-793.

[32] W. S. Levine and M. Athans, "On the determination of the optimal constant output feedback gains for linear multivariable systems," *IEEE TAC*, AC-15, 1970, 44-48.

[33] D. G. Luenberger, "Observers for Multivariable Systems," *IEEE TAC*, AC-11, 1966, 190-7.

[34] David G. Luenberger, *Introduction to Dynamic Systems: Theory, Models, and Applications*, John Wiley & Sons, New York, 1979.

[35] J. M. Maciejowski, *Multivariable Feedback Design*, Addison-Wesley Publishing Company, Reading, MA, 1989.

[36] Pertti M. Mäkilä, "On the Anderson-Moore method for solving the optimal output feedback problem," *IEEE TAC*, AC-29, 1984, 834-6.

[37] Pertti M. Mäkilä and Hannu T. Toivonen, "Computational methods for parametric LQ problems — a survey," *IEEE TAC*, AC-32, 1987, 658-71.

[38] Jerry M. Mendel, "A concise derivation of optimal constant limited state feedback gains," *IEEE TAC*, AC-19, 1974, 447-8.

[39] Mathieu Mercadal, *H_2, Fixed Architecture, Control Design for Large Scale Systems*, Ph.D. thesis, M.I.T., June 1990.

[40] Mathieu Mercadal, personal communication.

[41] K. D. Minto, *Design of Reliable Control Systems: Theory and Computations*, Ph.D. thesis, University of Waterloo, 1985.

[42] K. D. Minto, J. H. Chow, and C. Chalek, *ISICLE© User's Guide: Version 1.0,* General Electric Company, 1989.

[43] C. Molar, J. Little, S. Bangert, and S. Kleinman, *PRO/PC-MATLAB User's Guide*, The Mathworks, Inc., Sherburn, MA, 1990.

[44] B. Morton and R. Mc Afoos. "A mu-test for real parameter variations," In *Proc. ACC*, June 1985.

[45] Denis Mustafa, "A class of systems for which balanced truncation is Hankel-norm optimal," submitted CDC, 1991.

[46] Denis Mustafa, "Combined H_∞/LQG control via the optimal projection equations: on minimizing the LQG cost bound," to appear *Int. J. Robust & Nonlinear Control*, 1991.

[47] Denis Mustafa and Keith Glover, "Controller reduction by H_∞–balanced truncation," to appear *IEEE TAC.*

[48] Denis Mustafa and Keith Glover, *Minimum Entropy H_∞ Control*, LIDS–R–1946, M.I.T., February, 1990.

[49] C. N. Nett, D. S. Bernstein, W. M. Haddad, "Minimal complexity control law synthesis, part 1: Problem formulation and reduction to optimal static output feedback," in *Proc. ACC*, June 1989, 2056-64.

[50] Katsuhiko Ogata, *Modern Control Engineering*, Prentice–Hall, Inc., Englewood Cliffs, NJ, 1970.

[51] Andrew Kelly Packard, *What's new with μ: Structured Uncertainty in Multvariable Control*, Ph.D. thesis, UC Berkley, 1988.

[52] A.F Potvin and K.D. Minto, "A General Framework for Constrained Configuration Control Law Design," to appear IFAC Symposium, Zurich, 1991.

[53] R. M. Redheffer, "On a certain linear fractional transformation," *J. Math. Physics*, Vol. 39, 1960, 269-286.

[54] Stephen L. Richter and Raymond A. DeCarlo, "Continuation methods: theory and applications," *IEEE TAC*, AC-28, 1983, 660-5.

[55] James K. Roberge, *Operational Amplifiers: Theory and Practice*, John Wiley & Sons, Inc., New York, 1975.

[56] J. P. Shellhorn "An homotopy method for Ricatti equations," in 4^{th} *IFAC/IFORS Symposium on Large Scale Systems: Theory and Application*, Zurich, 1986.

[57] Gilbert Strang, *Linear Algebra and its Applications*, Academic Press, New York, 1980.

[58] H. T. Toivonen and P. M. Mäkilä, "Newton's method for solving parametric linear quadratic control problems," *Int. J. Control*, Vol. 46, 1987, 897-911.

[59] M. Vidyasager, *Control System Synthesis: A Factorization Approach*, MIT Press, Cambridge, MA, 1985.

[60] Carl J. Wenk and Charles H. Knapp, "Parameter optimization in linear systems with arbitrary constrained controller structure," *IEEE TAC*, AC-25, 1980, 496-500.

[61] Jan C. Willems, "Least squares stationary optimal control and the algebraic Ricatti equation," *IEEE TAC*, AC-16, 1971, 621-34.

[62] G. Zames, "Feedback and optimal sensitivity: model reference transformations, multiplicative seminorms, and approximate inverses," *IEEE TAC*, AC-26, 1981, 301-320.