Computer-Aided Analysis and Design of Linear Control Systems
Computer-Aided Analysis and Design of Linear Control Systems

Mohammad Jamshidi
The University of New Mexico
Albuquerque, New Mexico

Mahmoud Tarokh
San Diego State University
San Diego, California

Bahram Shafai
Northeastern University
Boston, Massachusetts

Prentice Hall, Englewood Cliffs, New Jersey 07632
To

My sisters, Parvin, Mahin, Pooran, and Giti
and brothers, M. Hossein, M. Hassan, Ahmad,
Mahmoud, and Abolghasem

My wife Enyc

My wife Soheila, my daughter Ghazaleh,
and my son Shahin
CONTENTS

FOREWORD xiii
PREFACE xvii

1 INTRODUCTION TO LINEAR CONTROL SYSTEMS 1

1.1 Introduction 1
1.2 System Classifications and Control Problems 2
   1.2.1 System Classifications, 2
   1.2.2 Control Problems, 4
1.3 Feedback Control System Characteristics 7
   1.3.1 Response and Stability, 7
   1.3.2 Sensitivity, 9
   1.3.3 Disturbance Rejection, 9
1.4 Evolution of Modern Control Systems 11
1.5 Computer-Aided Control System Design 15
1.6 Scope of the Book 16

2 ANALYSIS OF LINEAR SYSTEMS: THE STATE-SPACE METHOD 18

2.1 Introduction 18
2.2 State Variables of a System 19
Contents

2.3 State Transformation 24
2.4 Transfer Function Matrix of a System 25
2.5 Standard Forms of State Equations 30
2.6 Solution of the State Equation 34

2.6.1 Response to Initial Conditions—Homogeneous Case, 34
2.6.2 Response to Input-Forced Case, 40
2.6.3 Complete Solution, 41
2.6.4 Discrete-Time Systems, 42

2.7 Discretization of a Continuous-Time System 47
2.8 Eigenvalues, Poles, and Zeros 49

Problems 52

3 SYSTEM PROPERTIES

3.1 Introduction 56
3.2 Linear Independence of Time Functions 57
3.3 Controllability 59
3.4 Observability 64
3.5 Discrete-Time Systems 69
3.6 Frequency-Domain Criteria 70
3.7 Decomposition of Systems 74
3.8 Minimal Realization 77
3.9 Stability of Linear Systems 80

3.9.1 Zero-Input Stability, 80
3.9.2 Zero-State Stability, 85
3.9.3 Total Stability, 88

3.10 Lyapunov Stability Analysis 88

Problems 92

4 STATE FEEDBACK AND OBSERVER DESIGN

4.1 Introduction 96
4.2 State Feedback Concept 97
4.3 Pole Placement 98

4.3.1 Coordinate Transformation, 101
4.3.2 Pole Placement for Single-Input Systems, 113
4.3.3 Pole Placement for Multi-Input Systems, 119
4.3.4 Pole Placement for High-Order Systems, 124
4.3.5 Technical Discussion and Computational Aspects of Pole Placement Problem, 137
5 OUTPUT FEEDBACK AND COMPENSATOR DESIGN

5.1 Introduction 174
5.2 Static Output Feedback 175
  5.2.1 A Direct Method for Pole Assignment by Output Feedback, 177
  5.2.2 An Iterative Method for Pole Assignment by Output Feedback, 182
5.3 Two and Three Terms Controllers 187
  5.3.1 Proportional Plus Derivative Output Feedback, 187
  5.3.2 Proportional Plus Integral Output Feedback, 197
  5.3.3 Proportional Plus Integral Plus Derivative Output Feedback, 207
5.4 Dynamic Compensator 208
  5.4.1 State-Space Design for Stabilization, Tracking, and Disturbance Rejection, 209
  5.4.2 Robust Control Design for Tracking and Disturbance Rejection, 217
  5.4.3 Dynamic Compensator and Static Output Feedback, 220

Problems 222

6 OPTIMAL CONTROL DESIGN

6.1 Introduction 230
6.2 Solution of the Optimal Control Problem 235
  6.2.1 The Hamilton-Jacobi Equation, 235
  6.2.2 The Minimum Principle, 241
  6.2.3 Numerical Solutions of Matrix Riccati Equations, 256
Contents

6.3 The Discrete-Time Maximum Principle 267
6.4 An Optimal Pole Placement Design 278
Problems 282

7 LARGE-SCALE SYSTEMS DESIGN 285

7.1 Introduction 285
7.2 Aggregated Models of Large-Scale Systems 286
   7.2.1 General Aggregation, 287
   7.2.2 Modal Aggregation, 296
   7.2.3 Balanced Aggregation, 306
   7.2.4 Near-Optimum Design via Aggregated Models, 313
7.3 Hierarchical Control 319
   7.3.1 Goal Coordination-Interaction Balance, 321
   7.3.2 Interaction Prediction, 324
7.4 Decentralized Control 332
   7.4.1 Stabilization and Pole Assignment by Decentralized
        Output Feedback, 333
   7.4.2 An Iterative Solution to Decentralized Output
        Control, 335
Problems 339

Appendix A REVIEW OF LINEAR ALGEBRA 342

A.1 Introduction 342
A.2 Linear Spaces 343
A.3 Linear Independence and Basis 344
A.4 Inner Product and Norms 344
A.5 Eigenvalues and Eigenvectors 346
A.6 Functions of a Matrix 348
A.7 Quadratic Forms 350
A.8 Determinant and Inverse Formulas 351
A.9 Singular Value Decomposition 354
Problems 355

Appendix B COMPUTER-AIDED CONTROL SYSTEMS DESIGN—
PACKAGES AND LANGUAGES 359

B.1 Introduction 360
B.2 Development of CAD Methods and Programs 365
B.2.1 CAD Methods, 365
B.2.2 CAD Programs, 367

B.3 MATLAB-Based CACSD Packages 371
B.3.1 CTRL-C, 372
B.3.2 MATRIXx, 380
B.3.3 CONTROL.lab, 403
B.3.4 PC-MATLAB, 409
B.3.5 MATLAB Toolboxes and Extensions, 415

B.4 Other Design Packages 418
B.4.1 FREDOM-TIMDOM-LSSPAK, 418
B.4.2 CADACS (KEDDC), 425
B.4.3 L-A-S, 434

B.5 CACSD Tools—A Brief Survey 450
B.6 Future of CACSD Programs 453

BIBLIOGRAPHY 454
INDEX 473
The mathematical theory of linear systems, such as linear differential, integral, and difference equations, comprises a rich and dynamic part of mathematics. A major portion of the related work on the fundamental structure and methodology of this field was developed in the nineteenth and twentieth centuries. Presently the research in linear mathematics is still growing at a rapid pace.

In most problems of technological and societal origin we are confronted with a study of a dynamical system, that is, a system such that its behavior evolves with time. From the early part of this century, scientists have attempted to develop analytic procedures for representing, approximating, and estimating the behavior of dynamical systems via linear mathematical models.

Since the nineteen twenties, a new breed of engineers and scientists has succeeded in developing a semi-formal liaison between the physical phenomena and the mathematical theory of linear systems. Their efforts have given birth to a number of engineering disciplines such as linear network, estimation, filtering, signal processing, feed-back and control theories. These new disciplines have exercised a major impact on the advancement of technology, and will continue to play a central role in the future.

Modern control theory embraces the study and the optimal design of systems of diversified origin. The broad spectrum of its applications includes various problems of manufacturing, maneuvering, air craft guidance, space vehicle tracking and robotics.

This book deals with the control theory of linear dynamical systems. The authors
present a comprehensive coverage of the analysis and design of control systems at a graduate level. There is an abundance of examples of applications, with an emphasis on computer-aided design (CAD).

The growth of mathematical research on linear systems opens new avenues for the study of physical and societal problems. The liaison between mathematics and technological applications requires continuous updating. In this respect, the authors should be commended for incorporating the new and very important use of CAD in their models.

The value of books of this caliber depends on the depth of the mathematical substance, the diversity of methods of application, and how the new tools of technology are implemented. To maintain an optimal balance between the advancement of abstract research and the progress of technology, a continuous re-evaluation of the field is indispensable. It is hoped that the authors of the present comprehensive work will continue with their up-to-date aims and efforts in the future editions.

The publication of this book coincides with a cultural event pertinent to the authors’ origin and backgrounds. The year 1990 is highlighted by United Nations Educational Scientific and Cultural Organizations (UNESCO) as the millennium of Shahnameh, the great masterpiece of Persian Epic Poetry.

Inspired by this event, this volume is dedicated to the brilliant contributions of early Persian scholars to the fundamentals of mathematics. Here, we recognize but a few of those mathematicians and philosophers who laid down foundations of algebra and trigonometry about a thousand years ago. It is to be noted that during this period most scientific treaties of Persian scholars were written in Arabic.

**KHOWARIZMI (Mohammed ibn Musa) died between 835–845.**
Author of the first known book bearing the name of Algebra. His mathematical and astronomical works were translated into Latin under the title Algoritmi de numero indorum. The commonly used word “algorithm” is derived from the name Al-Khowarizmi.

**RAZI (Mohammed ibn Zarariya, known in the West as Rhases, died about 932).**
Celebrated physician, discoverer of alcohol and author of books on geometry and astronomy.

**BEIRUNI (Abu Rihan, 973–1048, approximately).**
Mathematician-astronomer and historian. His work comprises major contributions to mathematics, astronomy and history. He also systematically used the Balance as an early computing tool for solving algebraic problems.

**IBN-SINA (known in the west as Avicenna, 980–1037 approximately).**
Famous physician, philosopher, who wrote extensively on many subjects including music, medicine, geometry and the theory of numbers. The Latin translation of
Avicenna's contribution to medicine, "Canon" was a major medical source book in Europe for several centuries.

The development of science has been undoubtedly enriched by the efforts of all nations at one time or other. Fortunately, this pattern of global scientific contributions prevails with growing intensity. In time, science will transcend all subjective boundaries.

Fazlollah Reza*
Ottawa, ONT.
Canada

* Fazlollah Reza, a prominent scientist, engineer, and Persian literary personality, is internationally recognized for his contributions to the theory of Electrical Circuits, Systems and Communications. He has lectured and conducted research during four decades at many major academic centers in the United States of America, in Europe and in Canada. He has numerous scientific publications including the first book on information theory, (published by McGraw Hill in 1961). He is currently a visiting Professor at Concordia University and an honorary Professor at McGill University in Montreal.
This text is intended to give an introduction to modern control systems with a special emphasis on design. The book has been written for students who have had one course in conventional feedback control systems with or without an introduction to the concepts of state space and state variables. The book has two primary goals. One is to give an introduction to the multivariable control systems design, and the other is the introduction of computer-aided control system design tools to students, engineers, and researchers alike. For the latter goal of the book, the reader needs only minimal knowledge of computers. A mere familiarity with the use of a personal computer or a terminal of a main frame is sufficient.

The text has, in part, evolved over a number of years of the authors' courses on computer-aided analysis and design of control systems. The text is intended to bridge a gap between a first course in classical/modern control and theoretically-oriented graduate courses such as optimal control, large-scale systems, and an advanced course in multivariable control. Both continuous and discrete-time formulations of linear systems are presented. Wherever appropriate, algorithms are given to follow the theoretical concepts and are followed by an analysis and a design example on the computer. Two classes of numerical examples prevail throughout the book. One class is intended to illustrate the theoretical concepts, while the other is presented to explain solutions within a computer-aided design (CAD) environment. The latter class is designated as CAD EXAMPLES. These two classes of problems are also so designated in the PROBLEMS section at the end of each chapter. More advanced topics are
designated in both the table of contents and within the text by an asterix (*). The reader for an introductory control systems design course may choose to skip these sections.

Only a few years ago, the tools for analysis and synthesis available to the control engineer were paper, pencil, slide rule, spirule, and analog computer. The tools and the methods were simple enough that an engineer could easily master them in a relatively short time. However, over the past thirty years, control theory has evolved to a state where the digital computer has now become a requirement for the control system engineer, and Computer-Aided Control System Design (CACSD) environment has emerged as an indispensable tool.

A good CACSD program or package draws on expertise from many disciplines including aspects of computer engineering, computer science, applied mathematics (e.g., numerical analysis and optimization), as well as control systems engineering and theory. The need for such a breadth of knowledge is partially responsible for the paucity of high quality CACSD software, and indeed, CACSD can be considered still in its infancy. There has been, however, a certain degree of maturity among a number of CACSD packages. This point along with a historical view on the subject with an overview of 22 such programs is presented in Appendix B.

Chapter 1 serves as an introduction to the text. It provides a brief introduction to the design of multivariable systems and CACSD as a new discipline. A concise account of the evolution of modern control systems is presented.

Chapters 2 and 3 present the analysis of linear control systems. In Chap. 2, the state space and transfer function descriptions of continuous and discrete-time systems are reviewed. The various solutions of the state equation are also given. The properties of linear systems are discussed in Chap. 3, where the three primary properties—controllability, observability, and stability—are reviewed. Both conventional and Lyapunov stability concepts are covered. Other topics such as minimal realization, frequency domain criterion for controllability, duality, and state transformations are also discussed.

Chapters 4 to 7 constitute the main theme of this text—Design. Chapter 4 is concerned with feedback and observer design. In this chapter, state feedback pole placement (eigenvalue assignment) of linear time-invariant systems is treated. Both single-input and multi-input systems using unity or full rank feedback matrices are treated. The other main themes of this chapter are full and reduced order observer design and feedback control using observers.

Chapter 5 is concerned with output feedback and compensator design. Among topics discussed here are static output feedback, two and three term (PD, PI, and PID) output feedback, and dynamic compensator design. Optimal control design is the subject of Chap. 6. The optimal control problem has been treated through both the sufficiency conditions (Hamilton-Jacobi equation) and necessary conditions (the Minimum Principle). Both continuous-time and discrete-time optimal control problems are considered. Ample discussions are made on the state regulator problem and solutions to the matrix Riccati equations.

In Chap. 7, large-scale systems theory is reviewed briefly. Aggregated models
of large-scale systems and near-optimum design based on aggregated models are discussed first. The celebrated control of large-scale systems via hierarchical and decentralized control constitute the next two themes of this chapter.

Appendix A provides a brief review of linear algebra. Topics discussed here are spaces, independence, basis, inner product, norms, eigenvalues, eigenvectors, functions of a square matrix, Schur transformation, singular-value decomposition, and quadratic forms. The final presentation of the book is devoted to computer-aided control systems design. Appendix B provides a detail and, as much as possible, an up to date overview of CACSD programs available today. Some 22 programs have been mentioned in this appendix. The programs have been divided into two main groups. The first group are those based on MATLAB (original matrix laboratory software developed by Cleve Moler) and the second are those which have been developed independent of MATLAB. In this appendix, four of the seven MATLAB-based programs are reviewed in some detail. These are: CTRL_C (c) Systems Control Technology, Inc., MATRIXx (c) Integrated Systems, Inc., CONTROL.lab (c) University of New Mexico's CAD Laboratory, and PC_MATLAB (c) the Mathworks, Inc.

The notable non-MATLAB software programs reviewed here are FREDOM—TIMDON—LSSPAK (c), developed at University of New Mexico and CADACS (KEDDC) at Bochum University, West Germany, and L-A-S developed by Professor S. Bingulac of Virginia Polytechnic and State University. There are many more proven CACSD programs available such as CC, developed by Peter Thompson, LUND, developed by Lund Institute of Technology, Sweden. However, unfortunately because of lack of space, we could not cover all noteworthy CACSD programs in this book. Appendix B finishes with a brief survey and a discussion on the future of CACSD programs.

As a guide to potential instructors of this textbook, we offer the following suggestions: For a design-dominated course on control systems, cover the following chapters and sections: Chapters 1, 2, 3, followed by Sec. 4.3, Chapter 5, Secs. 6.2, 7.3 to 7.4, and Appendix A. For a CAD-dominated course on control systems, follow the same chapters and sections, but with a great influence from Appendix B. On the other hand, for a first course on linear multivariable systems, the following order of chapters and sections is recommended: Chapters 1–4, followed by Secs. 5.2, 6.2, and Appendix A. For advanced research, the following list of sections is recommended: Sections 1.4, 3.8 to 3.11, 4.5, 5.2, 5.4, 6.3, 6.4, 7.1 to 7.4, and B.5.

We wish to acknowledge the many colleagues, graduate students, and staff members of the University of New Mexico, San Diego State University, University of California at San Diego, and Northeastern University who have encouraged us or contributed to the development of this book. In particular, we wish to express our appreciation to Dr. R. H. Seacat (former Chair), Dr. N. Ahmed (current Chair) of the Department of Electrical and Computer Engineering, and Dr. J. E. Thompson, Dean of Engineering at the University of New Mexico. We wish to express our appreciation to Dr. D. Short, Dean of College of Sciences and Dr. J. Elwin, Chair of Department of Mathematical Sciences at San Diego State University, Dr. J. Arnold,
Preface

University of California, San Diego, for constant support and encouragement. The authors acknowledge the support of Dr. J. G. Proakis, Chair of Electrical and Computer Engineering Department of Northeastern University. The authors would wish to thank Dr. H. Seraji of Jet Propulsion Laboratory, Professor M. Mansour of Swiss Federal Institute of Technology, and Professor R. L. Carroll of George Washington University for their encouragement. We appreciate with deep sincerity Professor F. Reza's decision to write a foreword for this book.

We are particularly indebted to many of our present and former graduate students, especially T. C. Yenn, R. Morel, G. L. Schotik, P. Cuneo, R. A. Carreras, and D. Moser of the University of New Mexico, D. Colannino, D. Cowley, and S. Beale of Northeastern University some of whom have worked very hard in the development and organization of software programs: CONTROL.lab, TIMDOM, and CAD__MCS (Computer-Aided Design of Multivariable Control Systems). We would like to thank Ms. Yvonne Armstead (George Washington University) for typing a major portion of the manuscript. We would also like to thank Ms. Janet Watts for editorial assistance to the first author. Our final tribute goes to our own families (Jila, Ava, and Nima Jamshidi), (Ency, Leila, and Ali Tarokh), and (Soheila, Ghazaleh, and Shahin Shafai) for their patience and moral support throughout this project.

PREFACE TO CAD__MCS SOFTWARE

We have incorporated a set of design and analysis modules for linear time-invariant systems into an IBM_PC® based software package called CAD__MCS. Although this software would not be able to implement every design technique described in the book, it can help solve a great many problems which would expand student's knowledge. User's guide for CAD__MCS is available. This program allows analysis, design, and simulation of linear systems. This software program is basically for educational use. We strongly suggest that the readers use some CACSD package while studying this book.

This program can run on the IBM PC and its compatible computers (PC, XT, or AT), with two floppy or hard disk drives. Some versions of it will also run on Apple MacIntosh computers as well. A plotting capability would allow the user to plot the results. However, a graphics card is necessary.

The program CAD__MCS is available from the first author on double-sided density 5½" disks. An order form for CAD__MCS, TIMDOM, LSSPAK, L-A-S, CADACS (KEDDC), as well as newly developed toolboxes for MATLAB on linear systems, large-scale systems, and robotics is included in the back of this book.

Mohammad Jamshidi, University of New Mexico, Albuquerque
Mahmoud Tarokh, San Diego State University, San Diego
Bahram Shafai, Northeastern University, Boston
Computer-Aided Analysis and Design of Linear Control Systems
1

Introduction to
Linear Control
Systems

1.1 INTRODUCTION

This chapter serves as a general introduction to the book. It is intended to provide a
framework within which the analysis and design of modern control systems can be
explained. As such, we may refer to concepts that will be defined and explained fully
in later chapters.

Control systems have played a vital role in the advancement of engineering and
science. The importance of control discipline lies in its application to a wide class
of systems ranging from space vehicle, missile guidance, aircraft, industrial processes,
biomedical devices, and social and economical systems to the present-day advanced
technological systems such as robotics and automated manufacturing. The basis for
the understanding of control systems is the foundation provided by linear system
theory which assumes a cause and effect relationship for the components of a system.

Section 1.2 begins with a qualitative description of the nature of systems and
their representations and is followed by a discussion on systems classifications and
control problems. In Sec. 1.3, the characteristics of feedback control systems in terms
of stability and response, sensitivity, tracking, and disturbance rejection is analyzed.
In Sec. 1.4, historical development of classical control and its evolution into modern
control is discussed. The emphasis is placed on the major development of modern
control theory and design.

In Sec. 1.5, the role of computers in the analysis and design of linear control
systems is discussed. We briefly explain how the use of computer-aided design (CAD)
packages can help in better understanding of a given design method. Finally, in Sec. 1.6 we give an outline of the main contents of the chapters that follow. The development of this chapter is influenced in part by Ogata (1970), Swisher (1976), Dorf (1986), Phillips and Harbor (1988), Kailath (1980), Patel and Munro (1982), Chen (1984), and Brogan (1985).

1.2 SYSTEM CLASSIFICATIONS AND CONTROL PROBLEMS

Webster's dictionary defines a system as "a regularly interacting or independent group of items forming a unified whole." From our point of view, a system is a collection of components that act together to perform a useful function and can be characterized by a finite number of attributes. Considering the example of a weight attached to a spring, then the system consists of two components and the attributes of this system are the mass of the weight and spring constant. System theory is the study of the interactions and behavior of the components mentioned above, when subjected to certain input excitations.

It is logical to begin the study of the systems from their models. System models can be developed by two distinct approaches: analytical modeling and experimental modeling. Analytical modeling consists of a systematic application of physical laws to system components in order to develop mathematical descriptions (model). Models are also derived from experimental modeling, known as system identification, whereby the recorded input-output data are used to estimate the unknown parameters of a preassumed model structure. Mathematical representation of a system could be input-output (external) description or input-state-output description known as state variable description. The latter provides a complete picture of the system structure representing how the internal variables, known as state variables, interact with one another, how the input variables affect the system states, and how the output variables are obtained from various combinations of the state variables and the inputs. Traditional differential (difference) equations representing the dynamical behavior of physical systems, inherently include both of these representations. Alternative forms for system representations are also possible, and will be discussed later in the text.

1.2.1 System Classifications

As a result of modeling, systems are classified according to the types of equations by which they are described. Systems can be classified in five levels as shown in Fig. 1.1 where at each level two major categories are distinguished. The dashed lines indicate the possibility of branching, similar to those shown on the same level.

Lumped-Parameter and Distributed-Parameter Systems. Lumped-parameter systems are those which can be described by ordinary differential (difference) equations. Such systems are also referred to as finite dimensional systems for reasons which will become clear later. In contrast, distributed parameter systems, referred
to as infinite dimensional systems, are those which require partial differential equations for their description. An electric circuit, in which the element sizes are negligible compared to the wavelength of the highest frequency of operation, is an example of a lumped-parameter system. A long transmission line is an example of a distributed-parameter system.

**Deterministic and Stochastic Systems.** In deterministic systems, all the parameters are known exactly. On the other hand, in stochastic systems some or all parameters are known only probabilistically.

**Continuous-Time and Discrete-Time Systems.** In continuous-time systems described by differential equations, the variables are defined for all values of time. More specifically, the independent variable which is time \( t \) can take any value from the finite interval \([t_1, t_2]\). In contrast, the variables in discrete-time systems are defined only at discrete instants of time. Discrete-time systems are described by difference equations.

**Linear and Nonlinear Systems.** A system satisfying the superposition principle is called linear. A linear system can be described by a linear differential or
difference equation. If one or more of the equations that describe the system are nonlinear, then the overall system is nonlinear. Alternatively, a system for which the superposition principle is not valid, is said to be a nonlinear system.

**Time-Invariant and Time-Varying Systems.** A system is called time-invariant or stationary if all its parameters are constant. Such systems can be described by constant coefficient differential or difference equations. If one or more of system parameters vary with time, the system is called time-varying or nonstationary. Such systems are described by differential or difference equations with time-varying coefficients.

Systems can also be categorized according to other classifications such as large-scale systems and small-scale systems. The large dimensionality of systems is an important issue which will be considered in this book. A definition of large-scale systems is given in Chap. 7.

### 1.2.2 Control Problems

The described system classifications allows us to define the associated control problems. A control system is a system capable of monitoring and regulating the operation of a process or a plant. A control system also is defined as an interconnection of components forming a system configuration that will produce a desired goal. The study of control systems is essentially a study of an important aspect of systems engineering and its applications. There are two major classes of control systems known as open-loop and closed-loop (feedback) control systems. Control systems in which the outputs of the plant or process (system under control) have no effect on the input quantities are called open-loop control systems. In contrast to an open-loop control system, a closed-loop control system utilizes a measure of the actual output and compares it with the desired output in order to produce the control signal. A closed-loop control system is generally more capable of coping with unexpected disturbances affecting the plant and uncertainties in the plant parameters than an open-loop control system. Figure 1.2 illustrates this important class of control systems. For a better understanding of this configuration, consider a home temperature control system. The desired goal is the setting of the thermostat (command input). The actual temperature is the output signal which is measured by the sensor. This measured value is compared with the command input and a difference (error) is obtained. If the error is not zero, it acts on the compensator (controller) and an actuating or control signal is generated. In a properly designed system, the magnitude of the error signal is reduced to zero and the plant output remains at its desired value. In this example, the comparator and the controller can be visualized as one block denoted by control law as shown in Fig. 1.2.

Many other examples of closed-loop control systems can be found in our everyday life and in science and technology. Process control systems, automatic control of aerospace vehicles, and feedback control of robot manipulators are some examples of extensive application of closed-loop control systems.
Since the classical text of Truxal (1955), many books have been published in the field of control systems showing the development of the control theory. We will attempt to give reference to many of them in this section and also in Sec. 1.4 in conjunction with our discussion on the evolution of the control field. With this introduction, it is now proper to state the main problems of feedback control systems. First, we start with the deterministic control problem and then proceed with the stochastic case. Finally, other control problems are addressed briefly.

Consider a system with a given mathematical representation, then the problem is to find a control strategy or law so that the feedback system is stable and meets some desired performance specifications. One of the important specifications is to require the output of the plant to track a reference signal. Because of physical limitations, it is not always possible to design a feedback system that meets this requirement at all times. The best we can achieve is usually to track the reference signal asymptotically. Next, we assume that a constant or slowly time-varying disturbance signal acts on the system. An additional requirement in this type of control problem is to reject the disturbance effect at the output. Thus, a typical deterministic control problem is to stabilize the system and to achieve asymptotic tracking and disturbance rejection (Kailath, 1980; and Chen, 1984). The problem of asymptotic tracking in control systems is traditionally referred to as the servomechanism problem, even though the system may not contain any servo or mechanical components. The problem of rejecting the effect of disturbance on the output, when the reference signal is constant, is called regulation.

Other design requirements may be stated in terms of relative stability, perfor-
mance measure or criterion, sensitivity, bandwidth, overshoot, response time, damping factor, interaction, integrity, robustness, and so on.

Let us briefly discuss some of these specifications in the framework of a control problem. If the specification is a performance criterion expressed in a suitable mathematical form, then we can formulate the problem of determining a control law which minimizes or maximizes the performance criterion. In this case, we have the optimal deterministic control problem (Athans and Falb, 1966; Kirk, 1970; Anderson and Moore, 1971; Kwakernaak and Sivan, 1972; Sage and White, 1977; and Lewis, 1986).

The robustness requirement in the design of control systems plays an important role. This characteristic is important because of the difficulty in finding an accurate linear model of a physical system and because of other uncertainty factors influencing the system. A controller design is said to be robust if the given system can be controlled by it in a desired manner, in spite of the allowable disturbances and changes in the system parameters. Thus, the concept of robustness of a control system has two essential ingredients: a system property which relates either to stability or performance, and a class of uncertainties against which the system property is robust such as neglected dynamics, parameter variations, and so on.

Stochastic control problem is similar to the deterministic control problem except that random signals are included. In other words, if in the control problem the disturbance and measurement error processes are modeled as random or stochastic phenomena, the adjective "stochastic" is used in the problem description. Furthermore, if a performance measure is introduced to evaluate the quality of the system’s behavior, and the control law is to be specified to minimize or maximize this measure, the problem is one of optimal stochastic control problem. (See the previous references on optimal deterministic control problem which for the most part include the stochastic case as well.)

In approaching the control problem, one is often motivated to separate it into two problems. First, that of estimating the system’s variable, and second, that of specifying a control algorithm which utilizes the estimates. This separation is intuitively appealing, since it is natural to determine the system states before a control input can be specified. Relevant references on estimation and control problems are Anderson and Moore (1979), Kwakernaak and Sivan (1972), O’Reilly (1983), Friedland (1986), Lewis (1986), Stengel (1987), and Grimble (1988).

Another important problem is system identification and adaptive control. System identification is concerned with determining the best estimate of unknown parameters of the system. The adaptive control problem is to determine a control law for a system with unknown parameters. Adaptive control systems have the ability of self-adjustment to cope with unpredictable changes in the system characteristics. Good sources of information on new results in system identification are the books by Ljung (1987), and Söderström and Stoica (1989). For a study of adaptive control and its application refer to Landau (1979), Narendra and Monopoli (1980), Chalam (1987), Astrom and Wittenmark (1988), Sastry and Bodson (1989), and Narendra and Anasawami (1989).

We conclude this section by pointing out that alternative versions of these control
problems exist as a consequence of system classifications of Fig. 1.1. For example, in dealing with discrete-time systems, we are concerned with the digital or computer control system problem (Cadzow and Martens, 1970; Kuo, 1980; Philips and Nagle, 1984; Ackermann, 1985; Ogata, 1988; Franklin et al., 1990). Similarly, we may have a nonlinear control system problem (Vidyasagar, 1978; Atherton, 1982; Leigh, 1983; Isidori, 1989), time-delay control problem (Malek-Zavarei and Jamshidi, 1987), distributed control system problem (Curtain and Pritchard, 1978; Schumacher, 1982; Curtain, 1982), and large scale control system problem (Siljak, 1978; Jamshidi, 1983).

1.3 FEEDBACK CONTROL SYSTEM CHARACTERISTICS

Feedback control systems have inherent desirable characteristics in terms of some but not necessarily all requirements imposed on them. Generally speaking, it is desirable that a control system respond in some controlled manner to applied inputs and initial conditions. This means that we require that a control system be stable with satisfactory transient response and steady-state accuracy. We would also like a control system to be insensitive to parameter changes of the system, and to reject unwanted disturbances. In this section, we analyze certain desired characteristics that a control system should have and explore the role of compensators which enhance the closed-loop system characteristics.

1.3.1 Response and Stability

Consider the single-input single-output system of Fig. 1.3(a) with the reference input \( R(s) \), disturbance input \( D(s) \), and the output \( C(s) \). The transfer function of the plant, the compensator, and the sensor are represented by \( G_p(s) \), \( G_c(s) \), and \( H(s) \), respectively. Note that for simplicity, we did not include the sensor noise in this configuration. By superposition, we can write the output in terms of \( R(s) \) and \( D(s) \) as

\[
C(s) = \frac{G_c(s) G_p(s)}{1 + G_c(s) G_p(s) H(s)} R(s) + \frac{1}{1 + G_c(s) G_p(s) H(s)} D(s) \tag{1.1}
\]

Let us start our analysis by assuming that \( D(s) = 0 \). Then the closed-loop transfer function is given by

\[
T(s) = \frac{C(s)}{R(s)} = \frac{G_c(s) G_p(s)}{1 + G_c(s) G_p(s) H(s)} \tag{1.2}
\]

It is well-known that the characteristic equation of the system can be expressed by \( 1 + G_c G_p H = 0 \). The roots of the characteristic equation are the closed-loop poles of the system and determine stability and the shape of the system's response. Although stability and response of linear system will be discussed in details in Chap. 3, our aim here is to provide a brief introduction to the subject. It is generally desirable
that a control system to have characteristics introduced in this section and controller is designed to improve such characteristics. A particular type of controller encountered in classical control design technique is the familiar proportional-integral-derivative (PID) controller. The design of such a controller to meet the desirable specifications is known for single-input single-output systems. For example, it is known that the steady-state accuracy is increased if poles at \( s = 0 \) are added to the open-loop transfer function or the open-loop gain is increased. This implies the use of a PI controller and design \( G_c(s) \), accordingly. However, in doing so, we may reduce stability margins or even cause instability. Recall that gain and phase margins are possible measures of relative stability. With respect to the Nyquist diagram, they are approximate indications of the closeness of open-loop frequency response to the point \(-1\). Thus, we can conclude that there is a trade-off between steady-state accuracy and relative stability requirements. To improve stability and/or transient response of the system, it also is recognized that a possible compensator \( G_c(s) \) is a PD controller. Since the time response can be related to the closed-loop frequency response, there is also a trade-off between the rise time and bandwidth. It is known that the product of rise time and bandwidth is approximately constant. Thus, for a fast response, the rise time should be decreased which forces the system bandwidth to be increased. However,
an increase in bandwidth will increase the system response to high-frequency noise sources which may be present in the system. In this case, the trade-off is between a fast-rise time and an acceptable noise response.

### 1.3.2 Sensitivity

The sensitivity of a control system to parameter variations of the system is a very important concept. A mathematical definition for sensitivity measure is given by

\[
S_a^T = \frac{\delta T(s)}{\partial a} \frac{a}{T(s)}
\]  

(1.3)

which represents the sensitivity of the transfer function \(T(s)\) with respect to the parameter \(a\). If we replace \(s\) in Eq. (1.3) with \(j\omega\), then we can treat the sensitivity for frequencies within the bandwidth of the system. A primary advantage of a closed-loop control system is its ability to reduce the system’s sensitivity. A closed-loop system senses the change in the output because of process variations and attempts to correct the output. To analyze this point mathematically, it is sufficient to consider the sensitivity of the closed-loop transfer function \(T(s)\) with respect to the plant transfer function \(G_p(s)\), which will vary because of the variation of some of its parameters. Thus, we have

\[
S_{G_p}^T = \frac{\delta T G_p}{\delta G_p} \frac{1}{T} = \frac{1}{1 + G_c G_p H}
\]  

(1.4)

It is apparent that the sensitivity of the system may be reduced by increasing the magnitude of the loop transfer function \(G_c G_p H\). Generally, one of the purposes of the compensator \(G_c(s)\) is to allow the loop gain to be increased. However, as noted before, increasing the open-loop gain can cause instability. Thus, once again, we are faced with design trade-offs. At this point, it is also interesting to derive the sensitivity of the closed-loop system transfer function to parameter changes of the sensor, that is,

\[
S_H^T = \frac{\delta T H}{\delta H} \frac{1}{T} = \frac{-G_c G_p H}{1 + G_c G_p H}
\]  

(1.5)

The minus sign indicates that an increase in \(H\) results in a decrease in \(T\). Thus, to decrease the system sensitivity with respect to the sensor, the loop gain must be decreased. However, this makes the system sensitive to variations in the plant. This reveals that the system cannot be insensitive to both the sensor and the plant parameter variations.

### 1.3.3 Disturbance Rejection

Now we investigate the problem of disturbance rejection by referring to the block diagram of Fig. 1.3(a) with the disturbance input \(D(s) \neq 0\). A method for reducing disturbance is to increase the loop gain via \(G_c(s)\). This causes the transfer function
from the disturbance input \( D(s) \) to the output in Eq. (1.1) to be approximately zero and the transfer function from the reference input \( R(s) \) to the output to be

\[
C(s) \equiv \frac{1}{H(s)} R(s)
\]  

(1.6)

Assuming perfect measurement or unity feedback, \( H(s) = 1 \), the output tracks the reference input very well over the frequency band for which the loop gain is large, and at the same time the disturbance is rejected as desired. An important conclusion can be made at this point if we let \( H(s) = 1 \) in Eq. (1.1); that is,

\[
C(s) = \frac{G_c(s) G_p(s)}{1 + G_c(s) G_p(s)} R(s) + \frac{1}{1 + G_c(s) G_p(s)} D(s)
\]

(1.7)

\[
= T(s) R(s) + S(s) D(s)
\]

where \( S(s) \) is the sensitivity function and \( T(s) \) is closed-loop transfer function in the absence of disturbance, which is also known as complementary sensitivity function.\(^1\)

One important design trade-off may be expressed by noting that the sensitivity and complementary sensitivity functions satisfy the identity

\[
S(j\omega) + T(j\omega) = 1
\]

(1.8)

This indicates that \( |S(j\omega)| \) and \( |T(j\omega)| \) cannot be both very small at the same frequency. Hence, at each frequency, there exists a trade-off between those feedback properties such as sensitivity reduction and disturbance response which are quantified by \( |S(j\omega)| \) and those properties such as sensor noise and robustness to high frequency uncertainty quantified by \( |T(j\omega)| \). Since \( S(j\omega) \) and \( T(j\omega) \) characterize important properties of a feedback system, the design specifications are usually expressed as frequency dependent bounds on the magnitudes of these functions. In practice, the levels of uncertainty and sensor noise become large at high frequencies, while sensitivity reduction and disturbance rejection are desired at low frequencies. Therefore, the trade-off imposed by Eq. (1.8) is generally performed by requiring the bound on \( S(j\omega) \) to be small at low frequencies, and the bound on \( T(j\omega) \) to be small at high frequencies, and neither bound to be excessively large at any frequency.

Before leaving this section, we discuss another method of disturbance rejection known as feedforward control which can be applied if the disturbance can be measured. Feedforward disturbance rejection is illustrated in Fig. 1.3(b). In this configuration the disturbance \( D(s) \) is measured and transmitted to the summing junction through the transfer function \( G_d(s) \), which is a compensator for the disturbance input. The output can then be expressed as

\[
C(s) = \frac{G_c(s) G_p(s)}{1 + G_c(s) G_p(s) H(s)} R(s) + \frac{1 - G_d(s) G_c(s) G_p(s)}{1 + G_c(s) G_p(s) H(s)} D(s)
\]

(1.9)

\(^1\) Note that if the sensor noise \( N(s) \) is included in the feedback path of Fig. 1.3(a), then its effect on the output can be expressed by the complementary sensitivity function as \( -T(s)N(s) \).
It is obvious that by choosing
\[ G_d(s) = \frac{1}{G_c(s) G_p(s)} \] (1.10)
the disturbance will be rejected completely. However, Eq. (1.10) cannot be satisfied exactly for all frequencies and hence good disturbance rejection will occur if Eq. (1.10) is satisfied approximately. The limitation of this type of disturbance rejection is also apparent from this equation.

In summary, design specifications are usually defined in terms of the step response requirements in the time domain such as rise time, settling time, percent overshoot, steady-state error, or in the frequency domain in terms of phase margin, gain margin, bandwidth, and so on. It is also possible to define them in a single functional form known as performance index. In any event, certain specifications may impose conflicting requirements on the system and all cannot be satisfied. In these cases, trade-offs in the design are necessary so that specified characteristics are satisfied at least to some extent.

### 1.4 Evolution of Modern Control Systems

In this section we give a brief historical review of the major developments in the area of modern control systems. We shall attempt to discuss the development of classical control and its evolution into modern control. The emphasis of our discussion is directed towards advances of modern control theory within which the framework of this book is based on.

The concept of feedback in control systems has a fascinating historical background. An early application of feedback control is the water-level float regulator which has been traced back to the period of 300 to 1 B.C. (Mayr, 1970; Fuller, 1976). However, the period preceding 1900 was characterized as the development of automatic control systems by intuitive inventions. The temperature regulator of Cornelis Drebbel, the pressure regulator of steam boilers by Denis Papin, and the water-level float regulator of Polzunov are among the first feedback systems invented in that period. However, it is generally agreed that the first automatic feedback controller used in an industrial process is the flyball governor developed by James Watt in 1769. Prior to World War II, control theory in the United States and Western Europe was greatly influenced by the development of electronic feedback amplifiers. The frequency domain approaches of Bode and Nyquist were used successfully in this respect. In Russia and Eastern Europe, applied mathematicians and engineers worked on the field of control theory by utilizing a time domain formulation using differential equations. The importance of automatic control was recognized during and after World War II when it became necessary to design military systems based on the feedback control methods. Frequency-domain techniques continued to dominate with the increased use of the Laplace transform and the complex frequency plane. Later, the development of the root-locus method by Evans provided a great step forward
and together with the Bode and Nyquist techniques established the core of what is known as classical control theory (Truxal, 1955; Evans, 1954). Design techniques based on the frequency response and the root-locus methods made it possible for control engineers to design feedback control systems that are stable and exhibit satisfactory performance requirements. However, classical control theory, which deals primarily with single-input single-output systems becomes, in general, inapplicable to multi-input multi-output systems.

In early 1960, the time domain representation of a system by means of state space description became a more advantageous approach to the analysis and design of control systems (Zadeh and Desoer, 1963). The state space description was by no means new at that time. Such a representation was used by Poincare (1892) in his work on celestial mechanics and by Lyapunov (1907, 1947) in his work on stability of motion. The study of multivariable systems by the state space approach is regarded as modern control theory. Today, there are several new developments in the area of multivariable control systems which show clearly the merging of the two approaches. Consequently, it appears that modern control engineering must consider both the time-domain and the frequency domain approaches in such a study. Although the aim of the present text is to concentrate on the state variable approach, the connection between the two will be studied at several points throughout this text.

The evolution of modern control theory was inspired by the need to solve important control problems such as maneuvering, guidance and tracking of aircraft, and space vehicles. It was further developed because of the need for more realistic models of the system, for optimal system design, and for new approaches to overcome the shortcomings of the classical methods. These considerations were accompanied by the continued developments of digital computers which facilitated more detailed and extensive studies of practical control problems. As a result of these requirements, accurate dynamic models of complex systems such as spacecrafts were obtained. Also, design specifications could be formulated in terms of minimizing or maximizing certain performance criteria. In other words, the problem was formulated as an optimal control problem. Early theory of optimal control, based on state space description, was developed by Pontryagin et al. (1963) in the USSR and by Bellman (1956, 1957) in the United States. The optimal control problem in its most general form is quite complex. Kalman (1960a, 1963b, 1964) gave the first comprehensive design procedure for linear optimal control problem with quadratic performance criterion. The significance of his work was that a feedback control law could be obtained directly from the problem specifications via the so called algebraic Riccati equation. Furthermore, it has been realized that such stabilizing controller possesses desirable robustness properties for the feedback system. Anderson and Moore (1971) established robustness measures for single-input optimal linear quadratic regulator (LQR) problems.

All systems are stochastic in nature to some extent. In the classical system theory, the work of Wiener (1949) provided a frequency domain design criteria for single-input single-output filters with optimal statistical characteristics. On the other hand, the time domain solution to optimal estimation problem for multi-input multi-output systems was developed by Kalman (1960c), and by Kalman and Bucy
(1961) based on state space. Kalman noticed that the optimum estimation problem was dual to the deterministic optimal control problem. His work known as Kalman filtering, has been traced back to the basic idea of least square filtering originated by Gauss.

Another major contribution of Kalman (1960a) was the introduction of structural properties of linear systems known as controllability and observability. Further investigations by Gilbert (1963) and Kalman (1962, 1963a, 1965) established the link between state space and transfer function representations through canonical structure theorem. This in turn related the concepts of controllability, observability, and minimality of the state space realizations. These results, in particular the structural properties of linear systems, laid the foundation of many new design techniques of modern control systems. For example, a major drawback of deterministic optimal control problem was the assumption of the availability of state variables for feedback implementation. In most practical systems, however, the state variables themselves are not available for feedback. This difficulty was overcome by Luenberger (1964, 1966, 1971) who introduced a device called an observer, which accomplished the estimation on the system state variables from the available inputs and outputs of the system. He showed that the condition under which the design could be performed was the observability of the system. Thus, the feedback design was carried out in two steps: first, the state feedback was designed assuming that all the states were available and then the feedback was implemented using an observer. A separation principle guaranteed the stability properties of the overall system. The linear quadratic regulator (LQR) problem was incorporated with the Luenberger observer in a deterministic environment where the available measurements were not badly contaminated with noise. On the other hand, the linear quadratic Gaussian (LQG) problem was used in conjunction with the Kalman-Bucy filter in situations where noisy measurements had to be considered.

The separation principle was so elegant and convenient that it was immediately adopted by Wonham (1967) for a more general class of problem. Wonham established the correspondence between controllability and pole (eigenvalue) assignment by state feedback. He showed that the poles of a linear time invariant system could be shifted to any desired locations by state feedback under the assumption of controllability of the system. The contribution of Wonham led to a variety of results on pole assignment by means of constant state and output feedback as well as observer-based controller and dynamic feedback compensator. Gopinath (1971), Brasch and Pearson (1970), Fallside and Seraji (1971, 1973), Fallside (1977, 1979) constitute early research in this area. At present, researchers consider the numerical aspects and improvements of the solution which will be discussed in Chaps. 4 and 5. It should be pointed out that in this and other areas, many researchers found it convenient to work in and use concepts from both the time domain and the frequency domain.

In the mid 1970s, new interest was directed towards frequency domain methods for multivariable systems. It seemed that in most industrial processes, application of the optimal control methods was not necessary. Instead, it was found that a simple controller stabilizing the process in some operating region to achieve certain transient specifications was sufficient for most applications. A number of investigators, es-
pecially in England, proposed frequency domain design methods for multivariable systems. One idea was to reduce the design of the multivariable control system to a number of single-loop designs via decoupling approach and to use the well-established classical control methods. The main disadvantages of such an approach were that a complicated cascade controller was required to achieve noninteraction and that exact decoupling was impossible in general. The latter shortcoming is due to the fact that the cascaded controller is essentially the inverse of the given system which is generally not physically realizable. In addition, when the system model is not known precisely, exact decoupling becomes impossible. Rosenbrock (1969) suggested the idea of reducing the interaction rather than eliminating it completely. He used the concept of diagonal dominance in his inverse Nyquist array method to reduce the interaction to an acceptable level so that single-loop techniques could be employed. With this success, other frequency domain methods were also developed such as characteristic-loci method of MacFarlane, sequential-return-difference method of Myne, and dyadic design of Owens (MacFarlane; 1979, 1980). It was inevitable that efforts would be made to generalize other results from classical control theory and design for multivariable systems. These investigations revealed that such generalizations were possible and the design methods based on them were developed (Seraji; 1979, 1980). The basis for such developments was the elegant algebraic theory of multivariable systems which was also established by Rosenbrock (1970, 1974). This theory was based on two new model representations called matrix fraction description and its generalized form referred to as polynomial matrix description. Rosenbrock defined elegantly, for the first time, the poles and the zeros of multivariable systems in this framework. Various other concepts and design problems were also introduced and solved. Most of these are discussed in Wolovich (1974), Kailath (1980), Patel and Munro (1982), Chen (1984), Sinha (1984), D’Azzo and Houpis (1988). Here, an important point of the design philosophy is the formulation of the problem in terms of the so called Diophantine equation, which can be translated into a set of linear algebraic equations in order to determine the compensator parameters. Wang and Davison (1973) showed that the majority of control problems such as exact model-matching, decoupling, constructing dynamic observers, and system inverses, could be formulated in a unified form called the minimal design problem. They also derived an elegant method for its solution. Forney (1975) reduced the problem to finding a minimal polynomial basis for a vector space over the field of rational functions. Soon several other researchers provided alternative solutions to this problem and in particular a numerically preferred algorithm was proposed by Kung, Kailath, and Morf (1977). Another algebraic theory applicable to a general class of linear multivariable system was developed by Pernebo (1981).

In parallel with the investigations in the frequency-domain approach, many significant contributions were made in the state-space area. As we pointed out previously, a fundamental property of LQR designs is that the resulting closed loop system is always stable with impressive robust properties against fairly large perturbations. Such robustness measures have been derived by Safonov and Athans (1977) and Safovov (1980) for multi-input systems. The problem of expressing the robustness
property of LQR designs quantitatively in terms of bounds on the perturbations in the system matrices was also resolved by Patel, Toda, and Sridhar (1977). The bounds were expressed in terms of the weighting matrices in the quadratic performance index, thereby enabling one to select appropriate weighting matrices to attain a robust LQR design. Doyle and Doyle and Stein (1978, 1979, 1981) showed that robustness properties of LQR is destroyed when an observer is included in the feedback design and consequently provided a recovery approach to preserve the robustness of observer-based controller design. The development in this direction led to a systematic design procedure known as linear quadratic Gaussian method with loop transfer recovery (LQG/LTR). A tutorial on this design technique is provided by Athans (1986). Several other results on the design of linear multivariable control systems are published in the special issue of IEEE Transaction on Automatic Control edited by Sain (1981).

A consequence of all these efforts was a growing recognition that frequency domain and state space methods enhance and complement each other.

Since 1976, a new theory based on stable factorization has attracted many researchers in the theory and practice of modern control systems. The beginnings of this theory can be attributed to the paper by Youla et al. (1976). It was observed that every rational function could be expressed as a ratio of stable rational functions, which resulted in a method for parametrizing all compensators that stabilize a given system. This result was greatly improved and simplified by many researchers and enabled to the resolution of several important control problems in the framework of the so called \( H_{\infty} \) control theory. For a unified treatment of this theory and recent development of robust control using frequency domain and state space techniques one may refer to Vidyasagar (1985), Francis (1987), Bhattacharyya (1987), Dorato (1987), Zafiriou and Morari (1989), and Dorato and Yedavalli (1990).

In this text, we will cover some of the available design methods. Our primary goal is to fill the gap between the classical methods and recent approaches to control theory using computer-aided analysis and design methods.

### 1.5 COMPUTER-AIDED CONTROL SYSTEM DESIGN

One of the aspects of modern control system that has received considerable attention is that of developing efficient and stable computational algorithms. This is important because the success of the elegant theories and design methods will depend on the efficiency and accuracy of implemented algorithms. Parallel with the advances in modern control, computer technology has made its own progress and has played a vital role in today’s society. Consequently, the field of control has been influenced by the revolution in computer technology. The first major impact of this technology is the availability of microcomputers in homes and in the offices. Now most control engineers have easy access to the powerful computer packages for system analysis and design. The second impact of computers is that the size and the class of problems which can be modeled, analyzed, and controlled are considerably larger than those previously treated. The third major impact which follows from the first one is the
common use of computers as an integral part of control systems. The wide range of computer usage is due to their low cost, small size, and reliability. The state variable method of modern control theory provides an ideal formulation for computer implementation and is responsible for much of the progress in this area.

One of the main goals of this book is to establish a relation between the two fields of modern control and computer software engineering. We believe that by using a computer aided design approach, all the principles and techniques of control can be demonstrated in fairly simple fashion. The software in creating and solving problems, is not limited to a particular one, rather a collection of powerful available packages has been used. Even if the reader does not have access to these packages, it is still worthwhile to study the numerical results that have been presented. In this way certain trade-offs, trends, and other interesting results will become apparent.

A major breakthrough in computer-aided control system design (CACSD) was the creation of a “matrix laboratory” for linear algebra by Moler (1980). This software called MATLAB, although not initially intended for control system design, was turned into a stepping stone for a great many powerful CACSD programs in a relatively short period of time. In parallel to this effort, several other CACSD programs have been developed for a wide range of problems and classes of systems. In Appendix B, as many as 22 such CACSD packages will be discussed in various degrees of detail.

1.6 SCOPE OF THE BOOK

The basic themes of this text are analysis and design of linear control systems in a computer-aided design framework. Before analyzing a dynamic systems, a mathematical model that describes the system completely must be determined. The mathematical representations used in this text are transfer function and state variable descriptions. In Chap. 2, we show how these descriptions can be set up and analyzed.

Usually, control engineers simulate the model on a computer to test the behavior of the system in response to various signals and disturbances. This is known as quantitative analysis and will be studied in Chap. 2 for both continuous as well as discrete-time systems. On the other hand, in the qualitative analysis, we are concerned with the general properties of the system, such as controllability, observability, and stability. These important concepts are treated in Chap. 3 using both the state space and transfer function formulations. This part of analysis is very important, because design techniques often depend on this study.

If the response of the system is not satisfactory, the system has to be improved (optimized) by adjusting certain parameters, otherwise, compensators have to be included to achieve the performance requirements. In Chap. 4, the problem of pole assignment by state feedback is discussed. Several algorithms are provided for the purpose of pole assignment. The construction of dynamic observers for estimating the inaccessible states of a system is also described. This enables stabilization by
feedback to be achieved even when the states are not available. The robustness of observer-based controller is also discussed in this framework.

Chapter 5 is devoted to the design of constant as well as dynamic output feedback. The design of proportional-integral-derivative controller from classical control is extended to multivariable systems. Both the state space and transfer function designs are explored with the aim of clarifying the basic goals of control system design, that is, stabilization, asymptotic tracking, and disturbance rejection.

In Chap. 6, we treat the problem of designing controller for multivariable systems, which are optimal in the sense that they minimize a quadratic cost function of the states and the control inputs. In doing so, we start with the general optimal control design setting and reduce the problem to the previously mentioned form, known as linear quadratic regulator problem. This design is one of the earliest and most powerful state-space methods available for the design of multivariable systems, as pointed out in Sec. 1.4. Among its attractive features is that it has also some desirable robustness properties.

Chapter 7 serves as an introduction to large-scale systems. Among topics discussed are model reduction, and hierarchical and decentralized control problems. Today, an integral part of modern control system deals with the analysis and design of large scale systems. Conventional techniques can not be directly applied to such systems because of high dimensionality and computational complexity. The purpose of this chapter is an attempt to introduce the fundamental and settled issues on the subject.

Finally, Appendix A reviews the most relevant topics on linear algebra and in Appendix B, a detailed description is given for four MATLAB-based CACSD programs (CTRL-C, MATRIX, CONTROL, and PC-MATLAB) and three non-MATLAB programs (KEDDC, TIMDOM, and L-A-S). Then, a brief survey of some 22 CACSD packages will follow.

A major thrust of this book is to explain how and why certain methods work and establish a bridge between computer-aided design (CAD) and modern control theory. Consequently, throughout the book numerical examples as well as CAD examples are included to support each topic and to familiarize the reader with the available CAD packages in the analysis and design of modern control systems.
2

Analysis of Linear Systems:
The State-Space Method

2.1 INTRODUCTION

The model of a physical system can be described in a variety of forms. A particularly useful description is the state-space representation where the dynamic behavior of the system is described by a set of first order differential or difference equations. The state equations of a given system are not unique and various transformations can be performed to obtain a set of state equations in a form suitable for the particular application.

We will be concerned mainly with the linear time invariant system models. Furthermore, for the cases where the input-output description is of primary interest, we can use the transfer-function description. There are simple methods of obtaining transfer-function description from the state-space representation, although generally, the converse is not true. For the analysis of a system, it is important to obtain the response of the system to excitations of interest. There are several convenient and computationally efficient methods of solving state equations of linear time invariant systems.

In this chapter, we will review state-space and transfer-function descriptions of continuous and discrete-time systems. We will also discuss methods of solving the state equations. A number of computer examples will be presented to illustrate the use of several available software packages.
2.2 STATE VARIABLES OF A SYSTEM

Time-domain analysis of systems uses the concept of the states of a system. To every system, we may attribute three sets of variables: that is, inputs, outputs, and states. The inputs or control variables \(u_1(t), u_2(t), \ldots, u_m(t)\) are stimuli to the system which are applied from external sources. The outputs \(y_1(t), y_2(t), \ldots, y_r(t)\) are the response of the system to inputs or initial conditions which can be measured physically. The states \(x_1(t), x_2(t), \ldots, x_n(t)\) are the set of variables such that the knowledge of these variables and the inputs are adequate to describe the system dynamics and to determine the outputs. More precisely, the state of a system at time \(t_0\) is defined as the minimum amount of information which together with the knowledge of the inputs for \(t \geq t_0\) is sufficient to uniquely determine the outputs of the system for \(t \geq t_0\). The state vector \(x(t)\) is an \(n \times 1\) vector whose elements are the states variable of the system, that is, \(x^T(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T\). Similarly, the input vector \(u(t)\) and the output vector \(y(t)\) are \(m \times 1\) and \(r \times 1\) vectors, and may be represented as \(u^T(t) = [u_1(t), u_2(t), \ldots, u_m(t)]^T\) and \(y^T(t) = [y_1(t), y_2(t), \ldots, y_r(t)]^T\), respectively.

In general, the dynamic equations of a lumped-parameter continuous system may be represented by

\[
\begin{align*}
\dot{x} &= f[x(t), u(t), t] \\
y &= g[x(t), u(t), t]
\end{align*}
\]  

(2.1a)  

(2.1b)

where \(f\) and \(g\) are nonlinear vector-valued functions. Equations (2.1a) and (2.1b) are the set of \(n\) first-order differential equations that describe the dynamic behavior of a nonlinear system. Equations (2.1a) and (2.1b) are called state and output equations, respectively. For linear systems, Eq. (2.1) (see Prob. 2.9) may be expressed as

\[
\begin{align*}
\dot{x}(t) &= A(t) x(t) + B(t) u(t) \\
y(t) &= C(t) x(t) + D(t) u(t)
\end{align*}
\]  

(2.2a)  

(2.2b)

where \(A(t), B(t), C(t),\) and \(D(t)\) are matrices of dimensions \(n \times n, n \times m, r \times n,\) and \(r \times m,\) respectively. The matrix \(A(t)\) is known as the system matrix. Equation (2.2a) is a set of \(n\) first-order linear differential equations and together with (2.2b) describe the behavior of an \(n\)-th order \(m\)-input \(r\)-output linear system. The block diagram representative of Eq. (2.2) is shown in Fig. 2.1.

When all the matrices in Eq. (2.2) are constant, the system becomes linear time invariant. In this case, we can write

\[
\begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t) + Du(t)
\end{align*}
\]  

(2.3a)  

(2.3b)
In many systems, there is no direct transmission from the input $u(t)$ to the output $y(t)$, in which case $D = 0$.

In linear time-invariant discrete-time systems, the states, inputs, and outputs are defined only at discrete-time intervals. For these systems, the dynamic behavior is described by the set of difference equations of the form

$$x(k + 1) = Ax(k) + Bu(k)$$

$$y(k) = Cx(k) + Du(k)$$

where $x(k)$, $u(k)$, and $y(k)$ are the values of the $n \times 1$ state vector, $m \times 1$ input vector, and $r \times 1$ output vector, respectively, at time $t = kT$ where $k$ is an integer and $T$ is a fixed (sampling) time interval. Note that Eq. (2.4) can also represent a continuous system whose states, inputs, and outputs have been sampled at time intervals $T$. In this case, the matrices $(A, B, C, D)$ of the discretized system Eq. (2.4) are different from those of the original continuous system Eqs. (2.3a) and (2.3b), (see Sec. 2.7).

**Example 2.1**

Consider the classical problem of an inverted pendulum mounted on a cart as shown in Fig. 2.2. A stiff rod of length $l$ connects the mass $m$ to the cart having a mass equal to $M$. The cart must be moved so that the mass $m$ is always in an upright position. Assuming that the angle $\theta$ is small, the equations of the motion are given as

$$M \ddot{x} + ml \ddot{\theta} - f(t) = 0$$  \hspace{1cm} (2.5a)

$$ml \ddot{x} + ml^2 \ddot{\theta} - mlg \theta = 0$$  \hspace{1cm} (2.5b)

where $f(t)$ is the applied force, $x(t)$ is the displacement and $g$ is the gravity constant. Obtain a state-variable representation for the system, assuming $M >> m$.

**Solution** The state variables of this mechanical system may be defined as

$$x_1 = x(t), \quad x_2 = \dot{x}(t), \quad x_3 = \theta(t), \quad x_4 = \dot{\theta}(t)$$  \hspace{1cm} (2.6)

Note that state variables for this system can be defined in many alternative ways, as will be discussed in Sec. 2.3. Substituting Eq. (2.6) into Eq. (2.5), we have

$$M \ddot{x}_2 + m l \ddot{x}_4 - f(t) = 0$$

$$l \ddot{x}_2 + g \ x_3 = 0$$
Solving for $\dot{x}_2$ and $\dot{x}_4$ and using the assumption $\frac{m}{M} << 1$, we obtain

$$\dot{x}_2 = \frac{-mg}{M} x_3 + \frac{1}{M} f(t)$$

(2.7)

$$\dot{x}_4 = \frac{g}{l} x_3 - \frac{1}{Ml} f(t)$$

Suppose that the angle $\theta(t)$ and the displacement $x(t)$ are measurable quantities and constitute the outputs. The state and output equations of the system are obtained from Eqs. (2.6) and (2.7) as

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{l} & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{M} \\ 0 \\ -\frac{1}{Ml} \end{pmatrix} f(t)$$

$$\begin{pmatrix} x \\ \theta \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

(2.8)

Equation (2.8) is in the form of Eq. (2.3) and describes the motion of the fourth order, single-input $u(t) = \frac{\Delta}{f(t)}$, two-output $y = \frac{\Delta}{x}$ inverted pendulum system. The matrices $(A, B, C, D)$ of the system are

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{l} & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1/M \\ 0 \\ -1/Ml \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix}, \quad D = 0$$
Example 2.2

Consider the two-link planar robot manipulator shown in Fig. 2.3. The dynamic equations of the motion of the robot joints can be represented by the nonlinear differential equation of the form

\[ M(\theta) \ddot{\theta} + N(\theta, \dot{\theta}) + G(\theta) = T \]  

(2.9)

where \( \theta(t) \), \( \dot{\theta}(t) \), and \( \ddot{\theta}(t) \) are the \( 2 \times 1 \) vectors of joint angles, velocities, and accelerations, respectively, and \( T \) is the \( 2 \times 1 \) vector of applied joint torques. The inertia matrix \( M(\theta) \) is given by

\[
M(\theta) = \begin{pmatrix}
    a_1 + a_2 \cos \theta_2 & a_3 + \frac{1}{2} a_2 \cos \theta_2 \\
    a_3 + \frac{1}{2} a_2 \cos \theta_2 & a_3
\end{pmatrix}
\]  

(2.10a)

where \( a_1, a_2, \) and \( a_3 \) are constants that depend on the lengths and masses of the robot links, and \( \theta_1 \) and \( \theta_2 \) are the joint angles. \( N(\theta, \dot{\theta}) \) is the \( 2 \times 1 \) vector of Coriolis and centrifugal torques and is given by

\[
N(\theta, \dot{\theta}) = \begin{pmatrix}
    -(a_2 \sin \theta_2)(\dot{\theta}_1 \dot{\theta}_2 + \frac{1}{2} \dot{\theta}_2^2) \\
    \frac{1}{2} (a_2 \sin \theta_2)(\dot{\theta}_1^2)
\end{pmatrix}
\]  

(2.10b)

Finally, \( G(\theta) \) is the gravity loading vector given by

\[
G(\theta) = \begin{pmatrix}
    a_4 \cos \theta_1 + a_5 \cos (\theta_1 + \theta_2) \\
    a_5 \cos (\theta_1 + \theta_2)
\end{pmatrix}
\]  

(2.10c)

where \( a_4 \) and \( a_5 \) are constant quantities. Assuming \( a_1 = 4, a_2 = 2, a_3 = 1, \) and \( a_5 = 25 \), obtain a linearized model of the robot for small motions around the steady-state configuration \( \theta_1 = 0, \theta_2 = 90^\circ \), that is, arm upright configuration (see Fig. 2.3).

Solution For small perturbation in joint angle vector \( \Delta \theta(t) = q(t) \), and joint torque vector \( \Delta T(t) = u(t) \), about the operating point \( P = (\theta_p, \dot{\theta}_p) \) where \( \theta_p = \begin{pmatrix} 0 \\ 90 \end{pmatrix}, \dot{\theta}_p = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \),
\[
\begin{pmatrix}
0 \\
0
\end{pmatrix}; \text{ Eq. (2.9) can be linearized. This is accomplished by expanding } M(\theta), N(\theta, \dot{\theta}), \text{ and } G(\theta) \text{ about the operating point using the Taylor series and ignoring second and higher order terms in } \theta \text{ and } \dot{\theta} \text{ to yield}
\]
\[A_2 \ddot{q}(t) + A_1 \dot{q}(t) + A_0 q(t) = u(t) \tag{2.11}\]
where the constant \(2 \times 2\) matrices \(A_0, A_1, \text{ and } A_2\) are defined by
\[
A_2 = (M)_p, \quad A_1 = \left( \frac{\partial N}{\partial \theta} \right)_p, \quad A_0 = \left( \frac{\partial (N + G)}{\partial \theta} \right)_p
\]
Using Eqs. (2.10a) to (2.10c), we have
\[
A_2 = \begin{pmatrix}
4 & 1 \\
1 & 1
\end{pmatrix},
A_1 = \begin{pmatrix}
-2a_2 \dot{\theta}_2 \sin \theta_2 & -a_2 (\dot{\theta}_1 + \dot{\theta}_2) \sin \theta_2 \\
a^2 \dot{\theta}_1 \sin \theta_2 & 0
\end{pmatrix}_p = 0
\]
and
\[
A_0 = \begin{pmatrix}
-a_4 \sin \theta_1 - a_5 \sin (\theta_1 + \theta_2) & -a_5 \sin (\theta_1 + \theta_2) - a_2 (\dot{\theta}_1 \dot{\theta}_2 + \frac{1}{2} \dot{\theta}_2) \cos \theta_2 \\
ation \sin (\theta_1 + \theta_2) & -a_5 \sin (\theta_1 + \theta_2) + \frac{1}{2} a_2 \dot{\theta}_1 \cos \theta_2
\end{pmatrix}_p
= \begin{pmatrix}
-25 & -25 \\
-25 & -25
\end{pmatrix}
\]
Defining \(x_1(t) = q_1(t), x_2(t) = q_2(t), x_3(t) = \dot{q}_1(t), x_4(t) = \dot{q}_2(t)\) and assuming the joint angles \(y_1(t) = q_1(t)\) and \(y_2(t) = q_2(t)\) are output quantities, we obtain from Eq. (2.11)
\[
\dot{x}(t) = \begin{pmatrix}
0 & I_2 & 0 & 0 \\
-A_2^{-1}A_0 & A_2^{-1}A_1
\end{pmatrix} x(t) + \begin{pmatrix}
0 \\
A_2^{-1}
\end{pmatrix} u(t) \tag{2.12}
\]
\[
y(t) = (I_2 \ 0) x(t)
\]
where \(x(t) = \begin{pmatrix}
x_1(t) \\
x_2(t) \\
x_3(t) \\
x_4(t)
\end{pmatrix}\) is the \(4 \times 1\) state vector, \(u(t) = \begin{pmatrix}
u_1(t) \\
u_2(t)
\end{pmatrix}\) is the \(2 \times 1\) input vector and \(y(t) = \begin{pmatrix}
y_1(t) \\
y_2(t)
\end{pmatrix}\) is the \(2 \times 1\) output vector.

Note that the linearized model Eq. (2.12) is valid only for small motions around the operating point \(P = (\theta_p, \dot{\theta}_p)\). When the operating point changes, new numerical values of the matrices \(A_0, A_1, \text{ and } A_2\) must be evaluated using Eqs. (2.10a) to (2.10c).
2.3 STATE TRANSFORMATION

The choice of state variables for a particular system is not unique. Given the system Eq. (2.3) with the input vector $u(t)$ and the output vector $y(t)$, a new state vector $z(t)$ may be chosen to describe the dynamic behavior of the system. The new equations of the system are

$$\dot{z}(t) = \hat{A} z(t) + \hat{B} u(t)$$
$$y(t) = \hat{C} z(t) + \hat{D} u(t)$$  \hspace{1cm} (2.13)

Clearly $z(t)$ and $x(t)$ must be related if Eqs. (2.3) and (2.13) are to describe the same system. The relationship

$$z(t) = P x(t)$$  \hspace{1cm} (2.14)

where $P^{-1}$ exists, is called a nonsingular linear transformation. Note that each new state variable is a linear combination of the old state variables $x_1, x_2, \ldots, x_n$. Alternatively, the state vectors $x$ and $z$ may be viewed as representing a point in two different coordinate systems $(x_1, x_2, \ldots, x_n)$ and $(z_1, z_2, \ldots, z_n)$, respectively. The relationship Eq. (2.14) transforms the $x$ coordinates into the $z$ coordinates by rotating the $x$ coordinates.

Substituting for $x(t) = P^{-1} z(t)$ into Eqs. (2.2a) and (2.2b), we obtain

$$\dot{z}(t) = P A P^{-1} z(t) + P B u(t)$$
$$y(t) = C P^{-1} z(t) + D u(t)$$  \hspace{1cm} (2.15)

Comparing Eq. (2.15) with Eq. (2.13), it is evident that the matrices of the new system $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ are related to the matrices of the original system $(A, B, C, D)$ by

$$\hat{A} = P A P^{-1}, \quad \hat{B} = P B, \quad \hat{C} = CP^{-1}, \quad \hat{D} = D$$  \hspace{1cm} (2.16)

In general, the transformation matrix can be time dependent in which case

$$z(t) = P(t) x(t)$$  \hspace{1cm} (2.17)

where it is assumed that $P(t)$ is nonsingular for all $t$ and continuously differentiable in $t$. Thus

$$\dot{z}(t) = \dot{P}(t) x(t) + P(t) \dot{x}(t)$$
$$= [\dot{P}(t) P^{-1}(t) + P(t) A P^{-1}(t)] z(t) + P(t) B u(t)$$

and

$$y(t) = C P^{-1}(t) z(t) + D u(t)$$

The matrices of the new system are now time dependent and are given by $\hat{A}(t) = P(t) P^{-1}(t) + P(t) A P^{-1}(t), \hat{B}(t) = P(t) B, \hat{C}(t) = C P^{-1}(t)$, and $\hat{D} = D$. 
CAD Example 2.1

In this example, PC-MATLAB\(^1\) will be used to demonstrate the state transformation problem of Sec. 2.3. The system used is the linearized model of the inverted pendulum of Ex. 2.1 with the new state vector \(z = (x, \dot{x}, \theta, \dot{\theta})^T\) and the following numerical values: \(m = 1\), \(g = 9.8\), \(M = 10\), and \(l = 1\).

\[
\begin{align*}
% & \text{state transformation using PC-MATLAB} \\
>> & a = [0 1 0 0 ; 0 0 -0.98 0 ; 0 0 0 1 ; 0 0 9.8 0]; \\
>> & b = [0 ; 0.1 ; 0 ; -0.1]; \quad c = [1 0 0 0 ; 0 0 1 0]; \\
>> & p = [0 0 1 0 ; 1 0 0 0 ; 0 0 0 1 ; 0 1 0 0]; \\
>> & ahat = p*a*inv(p), bhat = p*b,chat = c*inv(p)
\end{align*}
\]

\[
\begin{array}{ccc}
0 & 0 & 1.0000 \\
0 & 0 & 0 & 1.0000 \\
9.8000 & 0 & 0 & 0 \\
-0.9800 & 0 & 0 & 0 \\
\end{array}
\]

\[
\begin{array}{c}
0 \\
0 \\
-0.1000 \\
0.1000 \\
\end{array}
\]

\[
\begin{array}{c}
0 \\
1 \\
0 \\
0 \\
\end{array}
\]

2.4 TRANSFER FUNCTION MATRIX OF A SYSTEM

The state Eq. (2.3) is a time-domain representation of a system. In order to obtain further insight into the system behavior in terms of poles, zero, and frequency response, we find the transfer-function description of a system. The Laplace transform of Eq. (2.3) with zero initial condition is

\[
(sI - A) X(s) = B \ U(s) \tag{2.18a}
\]

\[
Y(s) = C \ X(s) + D \ U(s) \tag{2.18b}
\]

where \(I\) is the identity matrix, \(X(s) = L[x(t)], U(s) = L[u(t)], Y(s) = L[y(t)]\) and \(L\) denotes Laplace transform operator. Substituting for \(X(s)\) from Eq. (2.18a) into Eq. (2.18b), we obtain

\[
Y(s) = [C(sI - A)^{-1}B + D] \ U(s) = G(s) \ U(s) \tag{2.19}
\]

\(^1\) PC-MATLAB is a product of the Mathworks, Inc. See Appendix B for further information.
where \( G(s) = C(sI - A)^{-1}B + D \) is the \( r \times m \) transfer function matrix of the system relating the \( m \times 1 \) input vector \( U(s) \) to the \( r \times 1 \) output vector \( Y(s) \). For single-input single-output systems, \( G(s) \) is a scalar transfer function. For single-input multi-output (or multi-input single-output) systems, \( G(s) \) is a \( 1 \times r \) row (or a \( m \times 1 \) column) vector transfer function.

Since transfer function is an input-output description of the system, it remains unchanged under state transformation. To verify this statement, we obtain the transfer-function matrix of Eq. (2.13) as

\[
\hat{G}(s) = \hat{C}(sI - \hat{A})^{-1}\hat{B} + \hat{D} = C P^{-1}(sI - PAP^{-1})^{-1}PB + D = C(sI - A)^{-1}B + D = G(s)
\]

Thus, the transformed system \((\hat{A}, \hat{B}, \hat{C}, \hat{D})\) and the original system \((A, B, C, D)\) have the same transfer-function matrix.

The transfer-function matrix of the discrete-time system Eq. (2.4) is obtained by taking the \( z \) transform of Eq. (2.4) with zero initial conditions. This yields

\[
\begin{align*}
z X(z) &= A X(z) + B U(z) \\
Y(z) &= C X(z) + D U(z)
\end{align*}
\]

where \( X(z), U(z), \) and \( Y(z) \) are the \( z \) transforms of \( x(k), u(k), \) and \( y(k) \), respectively. Thus

\[
Y(z) = [C(zI - A)^{-1}B + D] U(z) = G(z) U(z)
\]

where \( G(z) = C(zI - A)^{-1}B + D \) is the \( r \times m \) transfer-function matrix of the discrete system.

**Example 2.3**

Find the transfer function of the inverted pendulum of Ex. 2.1.

**Solution** Here \( D = 0 \) and

\[
G(s) = C(sI - A)^{-1}B = \begin{pmatrix}
1 & 0 & 0 \\
0 & s & \frac{mg}{M} \\
0 & 0 & 1
\end{pmatrix}^{-1} \begin{pmatrix}
s & -1 & 0 \\
0 & \frac{mg}{M} & 0 \\
0 & 0 & -1
\end{pmatrix} \begin{pmatrix}
0 \\
1 \\
0 \\
\frac{M}{-1}
\end{pmatrix}
\]

\[
= \begin{pmatrix}
\frac{1}{M} s^2 + \frac{mg}{M^2 l} \\
\frac{s}{s^2 - \frac{g}{l}} \\
-\frac{1}{Ml} \\
\frac{s^2 - \frac{g}{l}}{s^2}
\end{pmatrix}
\]
Thus

\[
\begin{pmatrix}
\frac{1}{M} - s^2 + \frac{mg}{M^2 l} s^2 (s^2 - \frac{g}{l}) \\
- \frac{1}{M l} s^2 - \frac{g}{l}
\end{pmatrix}
\begin{pmatrix}
X(s) \\
\theta(s)
\end{pmatrix}
= F(s)
\]

**Leverrier-Faddeev Algorithm.** The procedure discussed for determining the transfer function matrix of a system is not convenient, specially for higher order systems, since it involves inversion of the polynomial matrix \((sI - A)\). The \(n \times n\) matrix \(\Phi(s) = (sI - A)^{-1}\) plays an important role in the analysis and design of linear time-variant systems, and is called the resolvent matrix. This matrix can be evaluated efficiently by Leverrier-Faddeev Algorithm. To use this algorithm, we write \(\Phi(s)\) as

\[
\Phi(s) = (sI - A)^{-1} = \frac{\text{adj} (sI - A)}{\text{det} (sI - A)} = \frac{Q(s)}{p(s)} \tag{2.20a}
\]

where \(Q(s) = \text{adj} (sI - A)\) is an \(n \times n\) polynomial matrix and can be expressed as

\[
Q(s) = Q_{n-1} s^{n-1} + Q_{n-2} s^{n-2} + \cdots + Q_1 s + Q_0 \tag{2.20b}
\]

where \(Q_1, \ldots, Q_n\) are constant \(n \times n\) matrices and \(p_1, p_2, \ldots, p_n\) are constant scalars coefficients of polynomial \(p(s)\). Note that the elements of adjoint matrix \(Q(s)\) are obtained by evaluating the determinants of \((n - 1) \times (n - 1)\) submatrices of \((sI - A)\), formed by deleting row-column pairs. Thus, the elements of \(Q(s)\) are polynomials in \(s\) of maximum degree \(n - 1\), as indicated in Eq. (2.20b). The polynomial \(p(s) = \text{det}(sI - A)\) is of fundamental importance in system analysis and design, and is known as the characteristic polynomial of \(A\). It can be written as

\[
p(s) = s^n + p_{n-1} s^{n-1} + \cdots + p_1 s + p_0
\]

where the roots of the characteristic polynomial \(\lambda_1, \lambda_2, \ldots, \lambda_n\) are called the eigenvalues of \(A\). (See Appendix A for a review of linear algebra.)

The following theorem due to Fadeev and Leverrier provides a procedure for the evaluation of \(Q(s), p(s)\), and \(\Phi(s)\).

**Theorem 2.1** The matrices \(Q_i\) and the scalars \(p_i, i = 0, 1, \ldots, n - 1\) of the resolvent matrix \(\Phi(s) = (sI - A)^{-1}\) can be determined as follows:

(a) \(Q_{n-1} = I\), (b) \(p_i = -\frac{1}{n - i} \text{tr} (AQ_i)\), (c) \(Q_{i-1} = AQ_i + p_i I\) \tag{2.21}
The term $\text{tr}(M)$ stands for the trace of the matrix $M$ (See Appendix A for a review of linear algebra). Furthermore, the matrices $A$ and $Q_i$ commute, that is, $AQ_i = Q_iA$, and

$$AQ_o + p_0 I = O \quad (2.22)$$

**Proof** Premultiply both sides of Eq. (2.20b) by $(sI - A)$ to yield

$$(sI - A)(Q_{n-1}s^{n-1} + Q_{n-2}s^{n-2} + \cdots + Q_1s + Q_0) = (s^n + p_{n-1}s^{n-1} + \cdots + p_1s + p_0)I$$

Equating the coefficients of like powers of $s$

$$Q_{n-1} = I$$

$$-AQ_i + Q_{i-1} = p_i I \quad i = 1, 2, \ldots, n - 1$$

$$AQ_0 = p_0 I$$

which are (a) and (c) in Eq. (2.21). On the other hand, if both sides of Eq. (2.20) are postmultiplied by $(sI - A)$ and the coefficients of like powers of $s$ are equated, we obtain

$$Q_{n-1} = I$$

$$-Q_iA + Q_{i-1} = p_i I \quad i = 1, 2, \ldots, n - 1$$

$$Q_0A = p_0 I$$

indicating that $AQ_i = Q_iA$ and hence the matrices $A$ and $Q_i$ commute. The proof of (b) in Eq. (2.21) is more involved and may be found in Chen (1984). Note that Eq. (2.22) is not needed and can be used only to check the calculations. The transfer function matrix $G(s)$ is now found by pre- and post-multiplying $Q_i$ by $C$ and $B$, respectively.

**CAD Example 2.2**

In this CAD example, TIMDOM/PC\(^1\) is used to calculate the transfer matrix of a 2-input, 2-output 4th order system in analytical form. The program utilizes a Fadeev-type algorithm to calculate the corresponding resolvent matrix $(sI - A)^{-1}$. Assuming that you have access to MS-DOS on an IBM PC, then type

\[D > \text{tranfn}\]

\[<<\text{TRANFNN}>>\] Calculates the Transfer Matrix, $G(s)$, for any Multi-input Multi-Output (MIMO) Time-Invariant (TIV) System of the form:

\[\frac{dx}{dt} = Ax + Bu \quad \text{where} \ A = nxn; \ B = nxm\]

\[y = Cx + Du \quad \text{where} \ C = nxn; \ D = nxm\]

\(^1\) TIMDOM/PC is a software program of CAD Laboratory Systems/Robotics, University of New Mexico. See Appendix B for further information.
Sec. 2.4 Transfer Function Matrix of a System

From the algorithm:
\[ G(s) = C(sI - A)^{-1}B + D \]

Number of States \( n = 4 \)
Number of Inputs \( m = 2 \)
Number of Outputs \( r = 2 \)

Matrix A
\[
\begin{array}{cccc}
0.000E + 00 & 0.100E + 01 & 0.000E + 00 & 0.000E + 00 \\
0.100E + 01 & 0.000E + 00 & 0.000E + 00 & 0.000E + 00 \\
0.000E + 00 & 0.000E + 00 & 0.000E + 00 & 0.100E + 01 \\
0.000E + 00 & 0.000E + 00 & -1.00E + 01 & 0.000E + 00 \\
\end{array}
\]

Matrix B
\[
\begin{array}{cc}
0.000E + 00 & 0.000E + 00 \\
0.100E + 01 & 0.000E + 00 \\
0.000E + 00 & 0.000E + 00 \\
0.000E + 00 & 0.100E + 01 \\
\end{array}
\]

Matrix C
\[
\begin{array}{cccc}
0.100E + 01 & 0.000E + 00 & 0.000E + 00 & 0.000E + 00 \\
0.000E + 00 & 0.000E + 00 & 0.100E + 01 & 0.000E + 00 \\
\end{array}
\]

Matrix D
\[
\begin{array}{cc}
0.000E + 00 & 0.000E + 00 \\
0.000E + 00 & 0.100E + 01 \\
\end{array}
\]

* TRANSFER MATRIX $G(s) = \hat{Q}(s)/p(s)$ where $\hat{Q}(s) = CQ(s)B$ and

\[
\hat{Q}(s) = D*\text{s}^4 + \text{SUM}(Q(i)*s^i)
\]

Matrix $\hat{Q}(3)$:
\[
\begin{array}{cc}
0.000E + 00 & 0.000E + 00 \\
0.000E + 00 & 0.000E + 00 \\
\end{array}
\]

Matrix $\hat{Q}(2)$:
\[
\begin{array}{cc}
0.100E + 01 & 0.000E + 00 \\
0.000E + 00 & 0.100E + 01 \\
\end{array}
\]
Matrix $\hat{Q}(1)$:

\[
\begin{array}{cc}
0.000E + 00 & 0.000E + 00 \\
0.000E + 00 & 0.000E + 00 \\
\end{array}
\]

Matrix $\hat{Q}(0)$:

\[
\begin{array}{cc}
1.000E + 00 & 0.000E + 00 \\
0.000E + 00 & -2.000E + 00 \\
\end{array}
\]

\[p(3) = 0\]
\[p(2) = 0\]
\[p(1) = 0\]
\[p(0) = -1\]

To recover the analytical answer:

\[
G(s) = \frac{\hat{Q}(s)}{p(s)} = \frac{Ds^4 + \sum_{i=0}^{3} \hat{Q}_i s^i}{s^4 + \sum_{i=0}^{3} p_i s^i}
\]

\[
= \frac{1}{s^4 - 1} \left[ s^2 + 1 \quad 0 \quad 0 \right] = \left[ \begin{array}{cc}
\frac{1}{s^2 - 1} & 0 \\
0 & \frac{s^2 + 2}{s^2 + 1} \\
\end{array} \right]
\]

### 2.5 STANDARD FORMS OF STATE EQUATIONS

The state equation of a system can be transformed into several standard forms, also known as canonical forms. These forms are generally applicable to single-input single-output systems. In this section, we consider four commonly used standard forms.

#### Controllable Canonical Form

The state equation

\[
\begin{align*}
\dot{x} &= \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 \\
-p_0 & -p_1 & -p_{n-2} & -p_{n-1}
\end{pmatrix} x + \begin{pmatrix}
0 \\
0 \\
\vdots \\
0 \\
1
\end{pmatrix} u(t) \\
y &= (\gamma_0 \gamma_1 \cdots \gamma_{n-1}) x
\end{align*}
\]  

(2.23)
is said to be in the controllable canonical form, where $\gamma_0, \ldots, \gamma_{n-1}$ are constant scalars. The reason for this terminology will become clear in Chap. 3. The scalar transfer-function of this form is

$$g(s) = \frac{Y(s)}{U(s)} = c(sI - A)^{-1}b$$

$$= (\gamma_0 \gamma_1 \cdots \gamma_{n-1}) \begin{pmatrix} s & -1 & 0 & 0 \\ 0 & s & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & s & -1 \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} p_0 & p_1 & \cdots & p_{n-2} & p_{n-1} & s + p_n \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}$$

It is easy to verify that $(sI - A)^{-1}b$ is equal to the cofactors of the last row of

$$\frac{1}{p(s)} (sI - A),$$

where

$$p(s) = \text{det} (sI - A) = s^n + p_{n-1}s^{n-1} + \cdots + p_1s + p_0$$

The cofactors of the last row of $\frac{1}{p(s)} (sI - A)$ are found to be

$$(sI - A)^{-1}b = \frac{1}{p(s)} \begin{pmatrix} 1 \\ s \\ s^2 \\ \vdots \\ s^{n-1} \end{pmatrix}$$

Hence

$$g(s) = c(sI - A)^{-1}b = \frac{\gamma_{n-1}s^{n-1} + \cdots + \gamma_1 s + \gamma_0}{s^n + p_{n-1}s^{n-1} + \cdots + p_1s + p_0}$$

Note that in the controllable canonical form, the coefficient of the characteristic polynomial appears as the elements of the last row of the system matrix $A$. A block diagram representation of the controllable canonical form is shown in Fig. 2.4. This block diagram is obtained from Eq. (2.23) by noting that

$$x_i = x_{i+1}, \ i = 1, 2, \ldots, n - 1; \quad x_n = -p_0 x_1 - \cdots - p_{n-1} x_n$$

**Observable Canonical Form**

The state equation

$$\begin{pmatrix} 0 & 0 & 0 & -p_0 \\ 1 & 0 & 0 & -p_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & -p_{n-1} \end{pmatrix} x + \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n-1} \end{pmatrix} u \quad (2.25)$$

$$y = (0 \ 0 \ \cdots \ 0 \ 1) x$$
is said to be in the observable canonical form, where $\beta_0, \ldots, \beta_{n-1}$ are constant scalars. Following a procedure similar to that of Controllable Canonical Form section, it can be shown that the transfer function of Eq. (2.25) is

$$g(s) = \frac{Y(s)}{U(s)} = C(sI - A)^{-1}B = \frac{\beta_{n-1}s^{n-1} + \cdots + \beta_1s + \beta_0}{s^n + p_{n-1}s^{n-1} + \cdots + p_1s + p_0}$$

(2.26)

The block diagram representation of Eq. (2.25) is shown in Fig. 2.5.

**Diagonal Form**

Consider the state equation

$$\dot{x} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 & 0 \\ 0 & \lambda_2 & \cdots & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \lambda_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & \lambda_n \end{pmatrix} x + \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n-2} \\ \beta_{n-1} \end{pmatrix} u$$

(2.27)

$$y = (\gamma_0 \gamma_1 \cdots \gamma_{n-1}) x$$

where $\lambda_i$ are distinct. Equation (2.27) is said to be in diagonal form since the system matrix $A$ is diagonal. The characteristic polynomial of this system is
Figure 2.5  Block diagram of an observable canonical form.

\[ p(s) = \det (sl - A) = (s - \lambda_1)(s - \lambda_2) \cdots (s - \lambda_n) \]

Thus the diagonal elements are the system eigenvalues. The transfer function of Eq. (2.27) is

\[ g(s) = C(sI - A)^{-1}B = \]

\[
\begin{pmatrix}
\gamma_0 & \gamma_1 & \cdots & \gamma_{n-1} \\
\frac{1}{s - \lambda_1} & 0 \\
& \frac{1}{s - \lambda_2} \\
& & \ddots \\
& & & \frac{1}{s - \lambda_n}
\end{pmatrix}
\begin{pmatrix}
\beta_0 \\
\beta_1 \\
\vdots \\
\beta_{n-1}
\end{pmatrix}
\]

or

\[ g(s) = \frac{\gamma_0 \beta_0}{s - \lambda_1} + \frac{\gamma_1 \beta_1}{s - \lambda_2} + \cdots + \frac{\gamma_{n-1} \beta_{n-1}}{s - \lambda_n} \]  

(2.28)

Note that in order to transform the state equation into a standard form, certain conditions must hold. For example, when the system eigenvalues are distinct, it is always possible to transform the system matrix into the diagonal form.
Hessenberg form. The transformation of system equations into controllable canonical form can be a numerically unstable procedure, particularly for a high order system. A special form of system equations, called Hessenberg form, can be obtained using numerically stable methods and is useful for many computer-aided design applications.

The Hessenberg form for single-input single-output systems is given by

\[
\dot{x} = \begin{bmatrix}
  x & x & x \\
  x & x & x \\
  0 & x & x \\
  0 & 0 & x \\
  \cdot & \cdot & \cdot \\
  \cdot & \cdot & \cdot \\
  0 & 0 & 0 \\
  0 & 0 & 0
\end{bmatrix} x + \begin{bmatrix}
  x \\
  0 \\
  0 \\
  x \\
  \cdot \\
  \cdot \\
  0 \\
  0
\end{bmatrix} u
\]

\[ y = c \cdot x \]

where the column \( a_i \), \( i = 1, 2, \ldots, n \) of the system matrix has the property that the first \( (i + 1) \) elements of \( a_i \) are nonzero while the remaining elements are zero. A system can always be brought into the Hessenberg form by a sequence of orthogonal transformations. This will be explained in Chap. 4 for multi-input multi-output systems where the application of Hessenberg form to pole assignment is discussed.

2.6 SOLUTION OF THE STATE EQUATION

For the analysis of a system, the response of the system to excitation of interest must be determined. That is, given any initial state \( x(t_0) \) and input \( u(t) \) for \( t \geq t_0 \), the state vector \( x(t) \) and the output vector \( y(t) \) must be determined for \( t \geq t_0 \). For linear systems, the response of \( x(t) \) to \( x(t_0) \) and \( u(t) \) can be found separately. Because of linearity, superposition can be applied to obtain the total response by adding the individual responses. The output vector \( y(t) \) is then obtained from \( x(t) \) and \( u(t) \) by simple matrix manipulations.

2.6.1 Response to Initial Conditions—Homogeneous Case

For the case where \( u(t) = 0 \), the state equation of a linear time-invariant system is

\[
\dot{x}(t) = A \cdot x(t), \quad x(t_0) = x_0
\]  

(2.29)

In order to solve Eq. (2.29), we define the transition matrix \( \phi(t, t_0) \) as an \( n \times n \) matrix that satisfies the following two conditions:
(a) \( \phi(t_0, t_0) = I \)

(b) \( \dot{\phi}(t, t_0) = A \phi(t, t_0) \)

for all \( t_0 \) and \( t \geq t_0 \)

From this definition, the solution of Eq. (2.29) can be written as

\[
x(t) = \phi(t, t_0) x(t_0)
\]

(2.30)

To verify that Eq. (2.30) is the solution of Eq. (2.29), we form

\[
x(t_0) = \phi(t_0, t_0) x(t_0) = x(t_0)
\]

\[
\dot{x}(t) = \dot{\phi}(t, t_0) x(t_0) = A \phi(t, t_0) x(t_0) = A x(t)
\]

The transition matrix \( \phi(t, t_0) \), transfers the initial state \( x(t_0) \) to the state \( x(t) \), as seen from Eq. (2.30).

The following properties are deduced from the definition of the transition matrix:

(a) \( \phi(t, t_0) = \phi(t, t_1) \phi(t_1, t_0) \) for all \( t_1 \geq t_0, \ t \geq t_1 \)

(b) \( \phi(t, t_0) = \phi^{-1}(t_0, t) \)

For the linear time-invariant system Eq. (2.29), the transition matrix can be expressed as

\[
\phi(t, t_0) = e^{A(t-t_0)}
\]

(2.31)

since Eq. (2.31) satisfies both conditions of the definition. Furthermore, for time-invariant systems, \( t_0 \) can be assumed to be zero, \( t_0 = 0 \). If \( t_0 \neq 0 \), we can replace \( t \) by \( (t - t_0) \) in the solution, as seen from Eq. (2.31). Thus, the main task in obtaining the solution to Eq. (2.29) is to calculate \( e^{At} \).

Several methods are available for the calculation of \( e^{At} \). We present two such methods, namely Laplace transform and Cayley-Hamilton methods.

Taking Laplace transform of Eq. (2.29) we obtain

\[
(sI - A) X(s) = x_0
\]

and

\[
X(s) = (sI - A)^{-1} x_0 = \Phi(s) x_0
\]

(2.32a)

where \( \Phi(s) \) is the resolvent matrix. Thus

\[
x(t) = L^{-1} (\Phi(s)) x_0
\]

(2.32b)

where \( L^{-1} (\Phi(s)) \) denotes the inverse Laplace transform of \( \Phi(s) \). The matrix \( \Phi(s) \) is calculated using the Fadeev-Leverrier algorithm of Sec. 2.4, the inverse Laplace transform of each element of \( \Phi(s) \) is then found.
In the second method, the Cayley-Hamilton theorem is used (see Appendix A). This theorem can be used to express a function \( f(A) \) of the \( n \times n \) matrix \( A \), in terms of powers 0 to \( n - 1 \) of \( A \), that is,

\[
f(A) = \sum_{i=0}^{n-1} \alpha_i A^i
\]  

(2.33)

Since we are interested in finding \( e^{At} \), we set \( f(A) = e^{At} \) and write Eq. (2.30) as

\[
e^{At} = \sum_{i=0}^{n-1} \alpha_i(t) A^i
\]  

(2.34)

where \( \alpha_i(t) \) are functions of time, to be determined. Furthermore, Cayley-Hamilton theorem states that Eqs. (2.33) and (2.34) also hold if \( A \) is replaced by an eigenvalue of \( A \), that is,

\[
f(\lambda_j) = e^{\lambda_j t} = \sum_{i=0}^{n-1} \alpha_i(t)\lambda_j^i \quad j = 1, 2, \ldots, n
\]  

(2.35)

Equation (2.35) is a set of \( n \) algebraic equations in the \( n \) unknowns \( \alpha_i(t) \). These equations can be solved if the eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_n \) are distinct. For the case where the eigenvalue \( \lambda_j \) has multiplicity \( m_j \), Eq. (2.35) and \( (m_j - 1) \) equations formed by taking derivatives with respect to \( \lambda_j \) are used to obtain \( m_j \) equations for the eigenvalue \( \lambda_j \). Thus, for a repeated eigenvalue at \( \lambda_j \) we have

\[
e^{\lambda_j t} = \sum_{i=0}^{n-1} \alpha_i(t) \lambda_j^i = \alpha_0(t) + \alpha_1(t) \lambda_j + \cdots + \alpha_{n-1}(t) \lambda_j^{n-1}
\]

\[
t e^{\lambda_j t} = \alpha_1(t) + \cdots + (n - 1) \alpha_{n-1}(t) \lambda_j^{n-2}
\]

\[ \vdots \]

\[ \vdots \]

(2.36)

In either case of distinct or repeated eigenvalues, we obtain a total of \( n \) equations in the \( n \) unknowns \( \alpha_i(t) \), \( i = 0, 2, \ldots, n - 1 \). These equations are solved to obtain \( \alpha_i(t) \) and \( e^{At} \).

**Example 2.4**

Determine \( x(t) \) for the system

\[
\dot{x} = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & -1 & 2 \end{pmatrix} x; \quad x_0 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}
\]

**Solution** The resolvent matrix is

\[
\Phi(s) = (sI - A)^{-1} = \frac{1}{(s - 2)^3} \begin{pmatrix} (s - 2)^2 + 1 & s - 2 & 1 \\ s - 2 & (s - 2)^2 & s - 2 \\ -1 & -s + 2 & (s - 2)^2 - 1 \end{pmatrix}
\]
Thus

\[
\Phi(t) = e^{A t} = \begin{pmatrix}
\left(1 + \frac{t^2}{2}\right) e^{2t} & t e^{2t} & \frac{t^2}{2} e^{2t} \\
t e^{2t} & e^{2t} & t e^{2t} \\
-\frac{t^2}{2} e^{2t} & -t e^{2t} & \left(1 - \frac{t^2}{2}\right) e^{2t}
\end{pmatrix}
\]  

(2.37)

and the response \(x(t)\) is

\[
x(t) = \phi(t) x_0 = \begin{pmatrix}
\left(1 - t + \frac{t^2}{2}\right) e^{2t} \\
(t - 1) e^{2t} \\
\left(-\frac{t^2}{2} + t\right) e^{2t}
\end{pmatrix}
\]  

(2.38)

Now this problem is solved by using the Cayley-Hamilton method. Since the system has an eigenvalue at \(\lambda = 2\) with multiplicity \(m = 3\), we have from Eq. (2.36)

\[
e^{x t} = \alpha_0(t) + \alpha_1(t) \lambda + \alpha_2(t) \lambda^2
\]

\[
\frac{d}{d\lambda} (e^{\lambda t}) = t e^{\lambda t} = \alpha_1(t) + 2\alpha_2(t) \lambda
\]

\[
\frac{d^2}{d\lambda^2} (e^{\lambda t}) = r^2 e^{\lambda t} = 2\alpha_2(t)
\]

This yields

\[
\alpha_2(t) = \frac{1}{2} r^2 e^{2t}, \quad \alpha_1(t) = (t - 2t^2) e^{2t}, \quad \alpha_0(t) = (1 - 2t + t^2) e^{2t}
\]

The transition matrix \(\phi(t)\) is obtained from Eq. (2.34) as

\[
\phi(t) = e^{A t} = \alpha_0(t) I + \alpha_1(t) A + \alpha_2(t) A^2
\]

Substituting for \(A, A^2, \alpha_0(t), \alpha_1(t), \) and \(\alpha_2(t)\), we obtain \(\phi(t)\) as in Eq. (2.37) and the response is given by Eq. (2.38).

**CAD Example 2.3**

In Example 2.4, the resolvent and transition matrices of a third-order system were obtained. In this CAD example, the same system is reconsidered and, with the help of TIMDOM/PC, the resolvent matrix is calculated analytically.

<<RESMAT>> FINDS AN ANALYTIC SOLUTION OF THE RESOLVENT MATRIX \([s I - A][t^(-1)\] )

\(= \hat{Q} (s) = \text{SUM(} \hat{Q}(i) \ast (s^i)\) AND P(s) WHERE :\)

\(\hat{Q}(s) = \text{SUM(} \hat{Q}(i) \ast (s^i)\ AND P(s) = s^N + \text{SUM(p(i) \ast (s^i))}\)

WHERE i = 1, \ldots, N-1 AND N IS THE ORDER OF MATRIX A,

\(Q(i)\) ARE N \(\times\) N CONSTANT COEFFICIENT MATRICES AND p(i) ARE CONSTANT SCALAR COEFFICIENT OF THE CHARACTERISTIC POLYNOMIAL
Matrix A

\[
\begin{bmatrix}
0.200E+01 & 0.100E+01 & 0.000E+00 \\
0.100E+01 & 0.200E+01 & 0.100E+01 \\
0.000E+00 & -0.100E+01 & 0.200E+01
\end{bmatrix}
\]

Matrix \( \hat{Q}(2) \):

\[
\begin{bmatrix}
1.00000 & 0.00000 & 0.00000 \\
0.00000 & 1.00000 & 0.00000 \\
0.00000 & 0.00000 & 1.00000
\end{bmatrix}
\]

Matrix \( \hat{Q}(1) \):

\[
\begin{bmatrix}
-4.00000 & 1.00000 & 0.00000 \\
1.00000 & -4.00000 & 1.00000 \\
0.00000 & -1.00000 & -4.00000
\end{bmatrix}
\]

Matrix \( \hat{Q}(0) \):

\[
\begin{bmatrix}
5.00000 & -2.00000 & 1.00000 \\
-2.00000 & 4.00000 & -2.00000 \\
-1.00000 & 2.00000 & 3.00000
\end{bmatrix}
\]

Coefficients of the Characteristic Polynomial \( p(s) \):

\[
p(3) = 1 \\
p(2) = -6 \\
p(1) = 12 \\
p(0) = -8
\]

Note that this answer is identical to that of Ex. 2.4, because

\[
\Phi(s) = \frac{\hat{Q}(s)}{p(s)} = \frac{\hat{Q}_2 s^2 + \hat{Q}_1 s + \hat{Q}_0}{s^3 + p_2 s^2 + p_1 s + p_0}
\]

\[
= \begin{bmatrix}
s^2 - 4s + 5 & s - 2 & 1 \\
s - 2 & s^2 - 4s + 4 & s - 2 \\
-1 & -s + 2 & s^2 - 4s + 3
\end{bmatrix} \times \frac{1}{s^3 - 6s^2 + 12s - 8}
\]

which checks with result of Example 2.4.

**CAD Example 2.4**

Consider the linear system,

\[
\dot{x} = \begin{bmatrix}
0 & 1 & 0 \\
-2 & -3 & 1 \\
0 & 0 & -3
\end{bmatrix} x, \ x_0 \text{ is given}
\]

It is desired to find its transition matrix \( \exp(At) \) using TIMDOM/PC. The actual computer output is given by simply typing: D> expata (See Appendix B for details).
<<EXPATA>> Determines an ANALYTIC EXPRESSION for the EXPONENTIAL of a REAL nxn SQUARE MATRIX A, i.e. Exp (At)

Matrix A

\[
\begin{array}{ccc}
0.000\text{E}+00 & 0.100\text{E}+01 & 0.000\text{E}+00 \\
-2.000\text{E}+01 & -3.000\text{E}+01 & 1.000\text{E}+01 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 3.000\text{E}+01 \\
\end{array}
\]

\[\text{Exp} \ (A\text{t}) = \text{SUM} \left\{ C(i,j) \ast \left( (t'[j] - 1) \ast \text{Exp}[\text{Eigenvalue}(i)\ast t] \right) \right\} \]

for \(i = 1\) to 0 { Number of Distinct Eigenvalues }

\[j = 1 \text{ TO Mi (i) } \{ \text{ Multiplicity of Eigenvalues(i) } \} \]

Eigenvalues (1) = -2
Mi (1) = 1

\[\text{MATRIX C(1,1):} \]

REAL PART:

\[
\begin{array}{ccc}
-1.000\text{E}+00 & -1.000\text{E}+00 & -1.000\text{E}+00 \\
2.000\text{E}+00 & 2.000\text{E}+00 & 2.000\text{E}+00 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
\end{array}
\]

IMAGINARY PART:

\[
\begin{array}{ccc}
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
\end{array}
\]

Eigenvalue (2) = -1
Mi (2) = 1

\[\text{MATRIX C(2,1):} \]

REAL PART:

\[
\begin{array}{ccc}
2.000\text{E}+00 & 1.000\text{E}+00 & 5.000\text{E}+01 \\
-2.000\text{E}+00 & -1.000\text{E}+00 & -5.000\text{E}+01 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
\end{array}
\]

IMAGINARY PART:

\[
\begin{array}{ccc}
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
0.000\text{E}+00 & 0.000\text{E}+00 & 0.000\text{E}+00 \\
\end{array}
\]
40  Chap. 2  Analysis of Linear Systems

Eigenvalues \(\lambda_1 = -3\)
\(\lambda_2 = 1\)

\[ C_{ij} \]

**REAL PART:**

\[
\begin{bmatrix}
0.000E+00 & 0.000E+00 & 5.000E-01 \\
0.000E+00 & 0.000E+00 & -1.500E+00 \\
0.000E+00 & 0.000E+00 & 1.000E+00
\end{bmatrix}
\]

**IMAGINARY PART:**

\[
\begin{bmatrix}
0.000E+00 & 0.000E+00 & 0.000E+00 \\
0.000E+00 & 0.000E+00 & 0.000E+00 \\
0.000E+00 & 0.000E+00 & 0.000E+00
\end{bmatrix}
\]

Note that the analytical answer to the problem is given by

\[ e^{At} = \sum_{i,j=1}^{3} C_{ij} t^{i-1} e^{\lambda_i t} \]

Hence,

\[
e^{At} = C_{11} e^{-2t} + C_{21} e^{-t} + C_{31} e^{-3t} \]

\[
= \begin{bmatrix}
2e^{-t} - e^{-2t} & e^{-t} - e^{-2t} & \frac{1}{2}e^{-2t} - e^{-3t} + \frac{1}{2}e^{-3t} \\
-2e^{-t} + 2e^{-2t} & -e^{-t} + 2e^{-2t} & -\frac{1}{2}e^{-2t} + 2e^{-3t} - \frac{3}{2}e^{-3t} \\
0 & 0 & e^{-3t}
\end{bmatrix}
\]

Note that at \(t = 0\), this matrix reduces to an identity matrix, as expected. The reader can verify, using the Cayley-Hamilton theorem, that this is in fact the correct answer.

### 2.6.2 Response To Input-Forced Case

Consider the system

\[ x(t) = A x(t) + B u(t), \quad x(t_0) = 0 \]  \hspace{1cm} (2.39)

Taking the Laplace transform of Eq. (2.39) with zero initial conditions, we have

\[ (sI - A) X(s) = B U(s) \]

or

\[
X(s) = \Phi(s) B U(s)
\]  \hspace{1cm} (2.40)
Since the product of two functions in the Laplace domain is equal to their convolution in the time domain, we have

$$x(t) = \int_0^t \phi(t - \tau) \, B \, u(\tau) \, d\tau$$

(2.41)

where \( \tau \) is a dummy variable. Equation (2.40) or (2.41) can be used to find \( x(t) \).

### 2.6.3 Complete Solution

Consider the system

$$\dot{x}(t) = A \, x(t) + B \, u(t), \quad x(t_0) = x_0$$

(2.42)

Due to linearity, the total response \( x(t) \) to the initial condition \( x_0 \) and the input \( u(t) \) is obtained from Eqs. (2.32a) and (2.40) as

$$X(s) = \Phi(s) \, x_0 + \Phi(s) \, B \, U(s)$$

(2.43)

in the Laplace domain. In the time domain Eq. (2.43) becomes

$$x(t) = \phi(t) \, x_0 + \int_0^t \phi(t - \tau) \, B \, u(\tau) \, d\tau$$

where \( \phi(t) = e^{A t} \). The output response is

$$Y(s) = C \, \Phi(s) \, x_0 + (C \, \Phi(s) \, B + D) \, U(s)$$

(2.44)

and

$$y(t) = C \, \phi(t) \, x_0 + C \int_0^t \phi(t - \tau) \, B \, u(\tau) \, d\tau + D \, u(t)$$

(2.45)

Equation (2.45) constitutes the total output response due to the initial condition and the input.

### Example 2.5

Consider the single-input two-output system

$$\dot{x}(t) = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & -1 & 2 \end{pmatrix} \, x(t) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \, u(t)$$

(2.46)

$$y = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} \, x(t)$$

Obtain the output response to the initial condition \( x_0 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \) and the unit step input applied at \( t = 0 \).
Solution: The system matrix $A$ in (2.46) is the same as that of Example 2.4. Substituting Eq. (2.37) in Eq. (2.45), we obtain

$$y(t) = \left(1 - \frac{t}{t}\right)e^{2t} + \int_0^t \left(\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & (t - \tau) & e^{2(t - \tau)} \end{array}\right) d\tau$$

or

$$y(t) = \left(1 - \frac{t}{t}\right)e^{2t} + \left(\begin{array}{c} t^2 - t + \frac{1}{2} \\ -t^2 + 3t + \frac{1}{2} \end{array}\right)e^{2t} - \frac{1}{8}$$

$$= \left(\begin{array}{c} -\frac{1}{8} + \frac{1}{4}t^2 + \frac{3}{4}t + \frac{1}{8}e^{2t} \\ -\frac{1}{8} + \left(-t^2 - \frac{1}{4}t + \frac{1}{8}\right)e^{2t} \end{array}\right), \quad t \geq 0$$

CAD Example 2.5

In this example, PC-MATLAB will be used to simulate a third order system with two inputs and one output. A linear multivariable system can be simulated using any one of the CAD packages overviewed in Appendix B. The PC-MATLAB set of commands are as follows:

```matlab
>> a = [0 1 1; -2 -2 1; 0 0 -2]; b = [1 1; 0 1; 1 0];
>> c = [1 -1 2]; d = [0 0];
>> time = [0:1:5];
>> % step responses of system for both inputs
>> yc1 = step(a,b,c,d,1,time);
>> yc2 = step(a,b,c,d,2,time);
>> plot(time, yc1, time, yc2)
>> title ('Step Responses - Continuous - Time System'), xlabel ('Time'), ylabel ('y1 & y2')
```

Figure 2.6 shows the output step response $y$, for $u_1 = 1$, $u_2 = 0$, and $u_1 = 0$, $u_2 = 1$ for this system.

2.6.4 Discrete-Time Systems

In this section, we obtain the solution of the state equation for the discrete-time system Eq. (2.4).

First, we consider the homogeneous system
Step Responses – Continuous – Time System

Figure 2.6 Output step response for CAD Example 2.5.

\[ x(k + 1) = A \ x(k), \quad x(k_0) = x_0 \]  \hspace{1cm} (2.47)

where \( A \) is a constant \( n \times n \) matrix. The transition matrix of the discrete system Eq. (2.47) is defined as the \( n \times n \) matrix \( \phi(k) \) satisfying the following two conditions:

(a) \( \phi(k + 1) = A \ \phi(k), \)  \hspace{1cm} (b) \( \text{det} \ \phi(k) \neq 0 \)

for all \( k \geq 0 \). It is seen that for constant \( A \), the transition matrix is \( \phi(k) = A^k \), since it satisfies these conditions. Methods for evaluating \( A^k \) are analogous to those used for calculating \( e^{At} \), that is, \( z \)-transform and Cayley-Hamilton methods.

The \( z \)-transform of Eq. (2.47) is

\[ z X(z) - z x_0 = A X(z) \]

where \( X(z) \) is the \( z \)-transform of \( x(k) \). Thus

\[ X(z) = (zI - A)^{-1} z x_0 = \Phi(z) x_0 \]

and

\[ x(k) = Z^{-1}(\phi(z)) x_0 \] \hspace{1cm} (2.48)
where $Z^{-1}$ is the inverse $z$-transform. Alternatively, the Cayley-Hamilton theorem can be used to determine transition matrix $A^k$ and state vector $x(k)$. Using this theorem, we can write

$$A^k = \sum_{i=0}^{n-1} \alpha_i(k) A^i \quad (2.49a)$$

where $\alpha_i(k)$ are functions of $k$ and can be obtained from

$$\lambda_j^k = \sum_{i=0}^{n-1} \alpha_i(k) \lambda_j^i \quad j = 1, 2, \ldots, n \quad (2.49b)$$

and $\lambda_j$ are the eigenvalues of $A$. The procedure for determining the coefficients $\alpha_i(k)$ for distinct or repeated poles is similar to that described in Sec. 2.6.1.

The solution to the state equation

$$x(k + 1) = A \ x(k) + B \ u(k) \quad x(k_0) = x_0 \quad (2.50)$$

is found by successive substitutions. From Eq. (2.50), we have

$$x(1) = A \ x(0) + B \ u(0)$$
$$x(2) = A \ x(1) + B \ u(1) = A^2 x(0) + A B \ u(0) + B \ u(1)$$

Continuing this procedure, we obtain the general expression

$$x(k) = A^k x_0 + \sum_{j=0}^{k-1} A^{k-j} B \ u(j) \quad k \geq 0 \quad (2.51)$$

$$= \phi(k) x_0 + \sum_{j=1}^{k-1} \phi(k-j) B \ u(j)$$

The total output response due to initial condition $x_0$ and the input $u(k)$, $k \geq 0$ is

$$y(k) = C \ \phi(k) x_0 + C \sum_{j=0}^{k-1} \phi(k-j) B \ u(j) + D \ u(k) \quad k \geq 0 \quad (2.52)$$

The following example illustrates the procedure.

**Example 2.6**

Consider the system

$$\begin{pmatrix} x_1(k + 1) \\ x_2(k + 1) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 2 & 8 \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1(k) \\ u_2(k) \end{pmatrix}$$

Find $x(k)$ assuming $\begin{pmatrix} x_1(0) \\ x_2(0) \end{pmatrix} = \begin{pmatrix} -1 \\ 3 \end{pmatrix}$. The input $u_1(k)$ is obtained by sampling the unit
ramp function at \( t = 0, 1, \ldots, k \) seconds intervals and the input \( u_2(k) \) is obtained by sampling \( e^{-t} \) at the same time intervals.

**Solution** The transition matrix is

\[
\phi(k) = A^k = \alpha_0 I + \alpha_1 A = \begin{pmatrix}
\alpha_0 + \frac{1}{2} \alpha_1 & \frac{1}{8} \alpha_1 \\
\frac{1}{8} \alpha_1 & \alpha_0 + \frac{1}{2} \alpha_1
\end{pmatrix}
\]

The eigenvalues of \( A \) are \( \lambda_1 = \frac{3}{8} \) and \( \lambda_2 = \frac{5}{8} \). Using Eq. (2.49b), we have

\[
\begin{align*}
\left( \frac{3}{8} \right)^k &= \alpha_0 + \left( \frac{3}{8} \right) \alpha_1 \\
\left( \frac{5}{8} \right)^k &= \alpha_0 + \left( \frac{5}{8} \right) \alpha_1
\end{align*}
\]

This gives \( \alpha_0 = \frac{5}{8} \left( \frac{3}{8} \right)^k - \frac{3}{2} \left( \frac{5}{8} \right)^k \) and \( \alpha_1 = 4 \left( \frac{5}{8} \right)^k - \left( \frac{3}{8} \right)^k \). Hence

\[
\phi(k) = \begin{pmatrix}
\frac{1}{2} \left[ \left( \frac{5}{8} \right)^k + \left( \frac{3}{8} \right)^k \right] & \frac{1}{2} \left[ \left( \frac{5}{8} \right)^k - \left( \frac{3}{8} \right)^k \right] \\
\frac{1}{2} \left[ \left( \frac{5}{8} \right)^k - \left( \frac{3}{8} \right)^k \right] & \frac{1}{2} \left[ \left( \frac{5}{8} \right)^k + \left( \frac{3}{8} \right)^k \right]
\end{pmatrix}
\]

The total state response is obtained from Eq. (2.51) as

\[
x(k) = \begin{pmatrix}
\left( \frac{5}{8} \right)^k - 2 \left( \frac{3}{8} \right)^k \\
\left( \frac{5}{8} \right)^k + 2 \left( \frac{3}{8} \right)^k
\end{pmatrix}
\]

\[
+ \sum_{n=0}^{k-1} \begin{pmatrix}
\frac{1}{2} \left[ \left( \frac{5}{8} \right)^{k-n} + \left( \frac{3}{8} \right)^{k-n} \right] & \frac{1}{2} \left[ \left( \frac{5}{8} \right)^{k-n} - \left( \frac{3}{8} \right)^{k-n} \right] \\
\frac{1}{2} \left[ \left( \frac{5}{8} \right)^{k-n} - \left( \frac{3}{8} \right)^{k-n} \right] & \frac{1}{2} \left[ \left( \frac{5}{8} \right)^{k-n} + \left( \frac{3}{8} \right)^{k-n} \right]
\end{pmatrix}
\]

\[
\left( e^{-j} \right)
\]

**CAD Example 2.6**

Using TIMDOM/PC let's simulate the discrete-time system

\[
x(k + 1) = \begin{bmatrix}
0 & 1 & 1 \\
-0.5 & -0.5 & 1 \\
0 & 0 & -0.5
\end{bmatrix} x(k) + \begin{bmatrix}
0.5 & 0.5 \\
0 & 0.05 \\
0.5 & 0
\end{bmatrix} u(k)
\]

\[
y(k) = [1 \ 0 \ 0] x(k) + [0 \ 0] u(k)
\]
for a combination of an impulse at \( u_1 \) and step at \( u_2 \). The actual solution of this problem using command "COMSDS" of TIMDOM/PC is shown.

\[
<<\text{COMSDS}>> \quad \text{Finds a COMPLETE Solution of a linear Discrete-time System and presents a graphical output for the system ORDER of the system } n = 3 \text{ No. of system INPUTS } m = 2 \text{ No. of system OUTPUTS } r = 1 \text{ Number of Steps Kf = 20}
\]

Matrix A

\[
\begin{bmatrix}
0.000E+00 & 0.100E+00 & 0.100E+01 \\
-0.500E+00 & -0.500E+00 & 0.100E+01 \\
0.000E+00 & 0.000E+00 & -0.500E+00 \\
\end{bmatrix}
\]

Matrix B

\[
\begin{bmatrix}
0.500E+00 & 0.500E+00 \\
0.000E+00 & 0.500E-01 \\
0.500E-01 & 0.000E+00 \\
\end{bmatrix}
\]

Matrix C

\[
\begin{bmatrix}
0.100E+01 & 0.000E+00 & 0.000E+00 \\
\end{bmatrix}
\]

Matrix D

\[
\begin{bmatrix}
0.000E+00 & 0.000E+00 \\
\end{bmatrix}
\]

Initial STATES:

1 \( 0.000E+00 \)

2 \( 0.000E+00 \)

3 \( 0.000E+00 \)

**** STATE VARIABLES ****

At \( k = 0 \times (k) \) Is: \( 0.000E+00 \quad 0.000E+00 \quad 0.000E+00 \)

At \( k = 1 \times (k) \) Is: \( 0.100E+01 \quad 0.500E-01 \quad 0.500E-01 \)

At \( k = 2 \times (k) \) Is: \( 0.600E+00 \quad -0.475E+00 \quad 0.250E-01 \)

\[
\cdots
\]

At \( k = 19 \times (k) \) Is: \( 0.415E+00 \quad -0.117E+00 \quad 0.333E-01 \)

At \( k = 20 \times (k) \) Is: \( 0.416E+00 \quad -0.116E+00 \quad 0.333E-01 \)

The states and output responses for this system are shown in Fig. 2.7
2.7 DISCRETIZATION OF A CONTINUOUS-TIME SYSTEM

Consider the linear time-invariant system

\[
\dot{x}(t) = A \ x(t) + B \ u(t) \\
y(t) = C \ x(t) + D \ u(t)
\] (2.53)
The response of the system is given by
\[ x(t) = e^{At} x_0 + \int_0^t e^{A(t-\tau)} B u(\tau) \, d\tau \quad (2.54) \]
as discussed in Sec. 2.6.3. In sampled-data systems or in systems in which a digital controller is used, the input \( u \) changes only at discrete instants of time. A piecewise constant function is then generated by a hold circuit such that
\[ u(t) = u(k) \quad kT \leq t \leq (k + 1)T \quad k = 0, 1, 2, \ldots \]
where \( T \) is the sampling period and \( u(k) \) is the value of \( u(t) \) at \( t = kT \). If the behavior at the sampling instants \( 0, T, 2T, \ldots \) is of interest, a discrete-time representation of Eq. (2.53) can be obtained. From Eq. (2.54), we have
\[
x(k + 1) = e^{A(k+1)T} x_0 + \int_{kT}^{(k+1)T} e^{A(kT+T-\tau)} B u(\tau) d\tau
\]
\[ = e^{AT} \left( e^{Akt} x_0 + \int_{kT}^{kT} e^{A(kT-\tau)} Bu(\tau) d\tau \right) + \int_{kT}^{(k+1)T} e^{A(kT+T-\tau)} Bu(\tau) d\tau
\]
where \( x(k + 1) \) denotes the value of \( x(t) \) at \( t = (k + 1)T \). The input \( u(\tau) \) is constant in the interval \( kT \) to \( (k + 1)T \), and this expression becomes
\[ x(k + 1) = e^{AT} x(k) + \left( \int_0^T e^{A\alpha} \, d\alpha \right) B u(k) \]
where \( \alpha = (k + 1)T - \tau \). Thus, if only the response at the sampling instant is of interest, the continuous Eq. (2.53) can be replaced by the following discrete-time equation
\[ x(k + 1) = A_d x(k) + B_d u(k) \]
\[ y(k) = C x(k) + D u(k) \quad (2.55) \]
where
\[ A_d = e^{AT} \text{ and } B_d = \left( \int_0^T e^{A\alpha} \, d\alpha \right) B \]

**Example 2.7**

Given the system
\[
\begin{pmatrix}
\dot{x}_1(t) \\
\dot{x}_2(t)
\end{pmatrix} =
\begin{pmatrix}
0 & 1 \\
0 & -2
\end{pmatrix}
\begin{pmatrix}
x_1(t) \\
x_2(t)
\end{pmatrix} +
\begin{pmatrix}
0 \\
1
\end{pmatrix} u(t)
\]

obtain an equivalent discrete-time representation of the system.
Solution  The transition matrix is obtained using the Laplace-transform method as
\[ \phi(t) = e^{At} = \begin{pmatrix} 1 & \frac{1}{2}(1 - e^{-2t}) \\ 0 & e^{-2t} \end{pmatrix} \]

Thus the matrices of the discrete-time system are
\[ A_d = e^{At} = \begin{pmatrix} 1 & \frac{1}{2}(1 - e^{-2T}) \\ 0 & e^{-2T} \end{pmatrix} \]

and
\[ B_d = \left( \int_0^T e^{At} \, dt \right) B = \int_0^T \begin{pmatrix} \frac{1}{2}(1 - e^{-2\alpha}) \\ e^{-2\alpha} \end{pmatrix} d\alpha = \begin{pmatrix} \frac{1}{2} \left( T + \frac{e^{-2T} - 1}{2} \right) \\ \frac{1}{2} \left( 1 - e^{-2T} \right) \end{pmatrix} \]

Thus
\[ \begin{pmatrix} x_1(k+1) \\ x_2(k+1) \end{pmatrix} = \begin{pmatrix} 1 & \frac{1}{2}(1 - e^{-2T}) \\ 0 & e^{-2T} \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} + \frac{1}{2} \begin{pmatrix} T + \frac{e^{-2T} - 1}{2} \\ 1 - e^{-2T} \end{pmatrix} u(k) \]

2.8 EIGENVALUES, POLES, AND ZEROS

Consider the \( n \)-th order \( m \)-input \( r \)-output system \((A, B, C, D)\). The eigenvalues of the system matrix \( A \) are defined as the roots of the characteristic polynomial
\[ p(\lambda) = \det(\lambda I - A) = (\lambda - \lambda_1)(\lambda - \lambda_2) \cdots (\lambda - \lambda_n) \quad (2.56) \]
as explained in Sec. 2.4. The eigenvalues \( \lambda_i \), \( i = 1, 2, \ldots, n \) can be real or complex, distinct or repeated. The eigenvalues specify the "modes" of the system since they describe the natural or unforced \((u = 0)\) behavior of the system. To see this, assume, for simplicity, that \( \lambda_i \) is distinct and consider the unforced system
\[ \dot{x}(t) = Ax(t), \quad x(t_0) = x_0 \]

Taking the Laplace transform and expanding in partial fraction, we can write
\[ X(s) = (sI - A)^{-1} x_0 = \frac{\text{adj}(sI - A)}{\det(sI - A)} x_0 = \sum_{i=1}^{n} \frac{R_i}{s - \lambda_i} x_0 \quad (2.57) \]

where \( R_i \) are constant \( n \times n \) residue matrices. The inverse Laplace transform of (2.57) is
\[ x(t) = \sum_{i=1}^{n} \gamma_i e^{\lambda_i t} \]  

(2.58)

where \( \gamma_i = R_i x_0 \) are \( n \times 1 \) vectors. In general, the response contains \( n \) modes \( e^{\lambda_i t}, i = 1, 2, \ldots, n \). However, the initial condition \( x_0 \) can be chosen such that only a particular mode, say \( e^{\lambda_i t} \), is excited in which case

\[ x(t) = \gamma_i e^{\lambda_i t} \]

Thus, modes of the system are associated with the eigenvalue of \( A \).

In forming the transfer-function matrix

\[ G(s) = C(sI - A)^{-1}B + D = \frac{C \text{ adj}(sI - A)B + D \det(sI - A)}{\det(sI - A)} \]

there may be common factors between \( \det(sI - A) \) and all the elements of the numerator polynomial matrix \( C \text{ adj}(sI - A)B + D \det(sI - A) \). After cancelling common factors, we can write

\[ G(s) = C(sI - A)^{-1}B + D = \frac{W(s)}{\Delta(s)} \]

The roots of

\[ \Delta(s) = (s - s_1)(s - s_2) \cdots (s - s_n) \]

are called the poles of the system \((A, B, C, D)\). Note that \( n \leq n \) and that poles are a subset of eigenvalues.

A zero (also referred to as transmission zero) of the system Eq. (2.3) with the transfer-function matrix \( G(s) = C(sI - A)^{-1}B + D \) is defined as the value of \( s \) such that if the input \( u(t) = e^{s} u_0 \) is applied to the system, the output is identically equal to zero, that is, \( y(t) = 0 \), where \( u_0 \) is a constant \( m \times 1 \) vector. On writing Eq. (2.3) as

\[ (sI - A)X(s) - B U(s) = 0 \]
\[ C \ X(s) + D \ U(s) = Y(s) \]

and setting \( Y(s) = 0 \), we obtain

\[ \begin{pmatrix} sI - A & -B \\ C & D \end{pmatrix} \begin{pmatrix} X(s) \\ U(s) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \]

(2.59)

Thus the zeros of the system Eq. (2.3) are the values of \( s \) that satisfy (2.59). For Eq. (2.59) to hold, the \((n + r) \times (n + m)\) matrix \( \begin{pmatrix} sI - A & -B \\ C & D \end{pmatrix} \) must be rank deficient, that is,
\[
\text{rank} \begin{pmatrix} sI - A & -B \\ C & D \end{pmatrix} < \min \{(n + r), (n + m)\} \tag{2.60}
\]

For systems with equal number of inputs and outputs, that is, \( m = r \), the matrix in Eq. (2.60) is square and the condition of inequality (2.60) reduces to

\[
\psi(s) = \det \begin{pmatrix} sI - A & -B \\ C & D \end{pmatrix} = 0 \tag{2.61}
\]

where \( \psi(s) \) is a polynomial in \( s \) of maximum degree \( n \) and is called the zero polynomial of the system. The roots of \( \psi(s) = 0 \) are the zeros (transmission zeros) of the system. Using Schur's identity for determinants, Eq. (2.61) is expressed as

\[
\psi(s) = \det(sI - A) \det [C(sI - A)^{-1}B + D] \tag{2.62}
\]

\[\triangleq p(s) \det G(s)\]

Note, once again, that Eq. (2.62) is valid only for square systems, that is, systems with equal number of inputs and outputs.

**Example 2.8**

Determine poles and zeros of the system

\[
\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} u
\]

\[
\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} u
\]

**Solution**

The characteristic polynomial is

\[p(\lambda) = |\lambda I - A| = \lambda^4 - 1 = (\lambda + 1)(\lambda - 1)(\lambda + j)(\lambda - j)\]

and thus the system eigenvalues are at \(-1, +1, -j, +j\). The transfer-function matrix of the system is

\[G(s) = C(sI - A)^{-1}B + D = \begin{pmatrix} \frac{s^2 + 1}{s^2 - 1}(s^2 + 1) \\ \frac{0}{s^2 + 2}(s^2 - 1) \end{pmatrix}\]

Since there are no common cancellations, the system poles are the same as the eigenvalues of \( A \). The zero polynomial is obtained from Eq. (2.62) as

\[\psi(s) = (s^4 - 1) \left[ \frac{s^2 + 2}{(s^2 - 1)(s^2 + 1)} \right] = s^2 + 2\]

Thus, the system has two transmission zeros at \( \pm \sqrt{2} \). See also CAD Example 2.2. Transmission zeros can be easily obtained within most CACSD environments.
PROBLEMS

2.1 Find a state-space representation for the system described by
\[ \begin{align*}
\dot{z}_1 + 2(z_1 + z_2) &= u_1 \\
\dot{z}_2 - 2\dot{z}_2 + 3z_2 &= u_2
\end{align*} \]

2.2 Find a state-space representation for the system shown in Fig. P2.2. Assume \( \theta \) is small.

2.3 Consider a separately excited d.c. motor driving an inertia load \( J \) with viscous function \( b \) at an angular velocity \( \omega \), as shown in Fig. P2.3. The torque developed on the motor shaft is \( T(t) = k_1 i_d(t) \) where \( k_1 \) is a constant. Assume that \( e = k_2 \omega(t) \), where \( k_2 \) is a constant, and load torque \( T(t) = J \frac{d\omega}{dt} + b\omega \) where \( J \) and \( b \) are inertia and friction, respectively. Write the state and output equations of the system. The input is the applied voltage \( e_d(t) \) and the output is the shaft velocity \( \omega(t) \).

2.4 Use the program PC-MATLAB or TIMDOM/PC to obtain the controllable and observable forms for the systems described by
a. \( \ddot{\theta} - \theta - 2\dot{\theta} + 3\theta = \ddot{u} + 2\dot{u} - 3u \)
b. \( \theta(k + 4) + \theta(k + 3) + \theta(k + 1) - 3\theta(k) = 2u(k + 3) - 4u(k + 1) \)

2.5 Consider the single-input single-output system
\[ \begin{align*}
\dot{x} &= A x + b u \\
y &= c x
\end{align*} \]
where \( b \) and \( c \) are \( n \times 1 \) and \( 1 \times n \) vectors, respectively. Let
\[ q_n = b \]
\[ q_{n-1} = A q_n + p_{n-1} q_n = Ab + p_{n-1} b \]
\[ \vdots \]
\[ q_1 = A q_2 + p_1 q_n = A^{n-1} b + p_{n-1} A^{n-2} b + \cdots + p_1 b \]
and form the \( n \times n \) matrix \( Q = (q_1 \ q_2 \ \cdots \ q_n)^\Delta \ P^{-1} \), where \( P^{-1} \) exists. Show that
the state transformation $z = Px$ transforms the system into controllable canonical form given by Eq. (2.23).

2.6 Consider the system $(A, B, C)$ and let the eigenvector $\xi_i$ associated with the distinct eigenvalue $\lambda_i$, $i = 1, 2, \ldots, n$ of $A$ be defined as an $n \times 1$ vector satisfying

$$\lambda_i \, \xi_i = A \xi_i \quad i = 1, 2, \ldots, n$$

Show that the transformation $P = (\xi_1, \xi_2, \ldots, \xi_n)$ transforms the system into the diagonal form.

2.7 Find the transfer function matrices of the following systems by Leverrier-Fadeev method:

a. $\dot{x}(t) = \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 2 \\ 1 & 1 & 0 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} u(t)$

$y(t) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u(t)$

b. $x(k + 1) = \begin{pmatrix} 3 & 1 & 2 \\ 1 & -1 & 0 \\ 0 & 1 & 1 \end{pmatrix} x(k) + \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} u(k)$

$y(k) = (1 \ 1 \ 0) x(k)$

2.8 Determine the transmission zeros of the system

$$\dot{x}(t) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -2 & -1 & -3 & -4 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 0 \\ 0 \\ \beta \end{pmatrix} u(t)$$

$y(t) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} x(t)$

where $\beta$ is a constant. The system has a transmission zero coinciding with a system pole for what value(s) of $\beta$?
A function \( f[x(t), u(t)] \) is said to be a linear function of \( x(t) \) and \( u(t) \) if and only if
\[
\alpha_1 f[x_1(t), u_1(t)] + \alpha_2 f[x_2(t), u_2(t)] = f[\alpha_1 x_1(t) + \alpha_2 x_2(t), \alpha_1 u_1(t) + \alpha_2 u_2(t)]
\]
for all real numbers \( \alpha_1, \alpha_2 \), and all \( x_1(t), x_2(t), u_1(t), \) and \( u_2(t) \). Show that \( f[x(t), u(t)] \) is a linear function of \( x(t) \) and \( u(t) \) if and only if it can be expressed as
\[
f[x(t), u(t)] = A(t) x(t) + B(t) u(t)
\]
for some \( A(t) \) and \( B(t) \).

Find the solution of the system in Prob. 2.7a with the initial condition \( x(0) = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \) and
\[
u(t) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}
\]
for \( t \geq 0 \).

Let
\[
A = \begin{pmatrix} 1 & 0 & -1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}
\]

Find \( e^{At} \) using the Laplace-transform method.

Find \( y(k) \) for the system of Prob. 2.7b assuming \( x(0) = 0 \) and \( u(k) \) is a unit step function sampled at 0, 1, 2, \ldots intervals.

Verify that \( z(t) = e^{At} K e^{Bu} \) is the solution of
\[
z(t) = A z(t) + z(t) B, \quad z(0) = K
\]
where \( A, B, \) and \( K \) are constant \( n \times n \) matrices.

Write a computer program to obtain \( \phi(k) = A^k \) using the Cayley-Hamilton method. Assume that \( A \) is a \( 5 \times 5 \) matrix with distinct eigenvalues.

Write a computer program to determine the characteristic polynomial \( p(s) = \det(sI - A) \) of a given \( 5 \times 5 \) matrix.

Write a computer program to find the resolvent matrix \( \Phi(s) \) of a \( 5 \times 5 \) matrix.

Use a CAD package to obtain the time response of the system
\[
\dot{x} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ -6 & -11 & -6 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u
\]
where \( x_0 = 0 \) and \( u \) is a unit step input.

Write the wave equation
\[
\frac{\partial^2 \theta}{\partial t^2} = \sum_{i=1}^{3} \frac{\partial^2 \theta}{\partial x_i^2} \quad \theta = \theta(x_1, x_2, x_3, t)
\]
in the state-space form.
2.19 Consider the single-input single-output system \((A, b, c)\) and let \(G(s) = c(sI - A)^{-1}b\), 
\(p(s) = \det(sI - A)\) and \(W(s) = c \text{adj}(sI - A)b\) where no common factors exist between 
\(p(s)\) and \(W(s)\). Show that there exists an initial state \(x_0\) such that with the input \(u(t) = e^{\alpha t}, t > 0\) response is \(y(t) = G(\alpha) e^{\alpha t}\), where \(\alpha\) is not an eigenvalue of \(A\).

2.20 Use TIMDOM/PC to solve Prob. 2.11.

2.21 Use PC-MATLAB or your favorite CAD package to verify the results of Prob. 2.8.
3

System Properties

3.1 INTRODUCTION

In the analysis and design of control systems, certain structural properties of the system must be investigated to determine the degree and effectiveness of control that can be exerted on the system. Important structural properties of a system are controllability, observability, and stability.

Controllability deals with the following question: Given a system in its initial state \( x(t_0) \), can an input \( u(t) \) be found to transfer the initial state to a desired state \( x(t) \) in finite time? If such an input can be found, we say that the system is controllable. This implies that we can influence the dynamic behavior of a system by the input, only if it is controllable.

Observability addresses the following question: Is it possible to identify the initial state \( x(t_0) \) of a system by observing the output \( y(t) \) for a finite time? Thus, if a system is observable, we can determine its state by measuring the output.

Stability is concerned with the following fundamental question: Suppose a system that is initially at some equilibrium state is perturbed. Will the trajectory of the perturbed system eventually return to the equilibrium state or will it move further away from the equilibrium state? In the latter case, we say that the system is unstable. Stability is of practical importance, since external disturbances such as noise, drift, and so on, are always present in a real system.

In a linear-time invariant system, it is easy to determine controllability, ob-
servability, and stability directly from the matrices \((A, B, C, D)\) of the state-space representation.

### 3.2 Linear Independence of Time Functions

In order to obtain conditions for controllability and observability in terms of the system matrices, it is necessary to discuss the concept of linear independence of time functions.

Consider the set of \(1 \times m\) vector functions \(f_i(t), i = 1, 2, \ldots, n\) where the \(m\) elements of \(f_i(t)\) are functions of time. The set of functions \(f_i(t), i = 1, 2, \ldots, n\) are said to be linearly independent on the time interval \([t_0, t_1]\) if

\[
\sum_{i=1}^{n} \beta_i f_i(t) = 0 \quad (3.1)
\]

for \(t\) on \([t_0, t_1]\) implies \(\beta_i = 0, i = 1, 2, \ldots, n\); otherwise the functions are said to be linearly dependent on \([t_0, t_1]\). It is important to specify the time interval, since a set of functions can be linearly independent on an interval but linearly dependent on another interval. Equation (3.1) can be written as

\[
\beta F(t) = 0
\]

where \(\beta = (\beta_1 \beta_2 \cdots \beta_n)\) is a \(1 \times n\) vector and \(F(t) = \begin{pmatrix} f_1(t) \\ f_2(t) \\ \vdots \\ f_n(t) \end{pmatrix}\) is an \(n \times m\) matrix. The following theorem provides a test for determining linear independence of a set of vector functions.

**Theorem 3.1** The \(1 \times m\) vector functions \(f_i(t), i = 1, 2, \ldots, n\) are linearly independent on the interval \([t_0, t_1]\) if, and only if, the \(n \times m\) matrix

\[
G(t_0, t_1) = \int_{t_0}^{t_1} F(t) F^T(t) dt \quad (3.2)
\]

is nonsingular.

**Proof.** The proof is by contradiction. Suppose \(f_i(t)\) are linearly independent on \([t_0, t_1]\) but \(G(t_0, t_1)\) is singular. The singularity of \(G(t_0, t_1)\) implies that there exists a constant nonzero \(1 \times n\) vector \(\beta\) such that \(\beta G(t_0, t_1) = 0\) and also \(\beta G(t_0, t_1) \beta^T = 0\). Thus

\[
\beta G(t_0, t_1) \beta^T = \int_{t_0}^{t_1} [\beta F(t)] [\beta F(t)]^T dt = 0 \quad (3.3)
\]
Since \([\beta F(t)][\beta F(t)]^T\) is a nonnegative scalar function for all \(t\) on \([t_0, t_1]\), Eq. (3.3) implies that \(\beta F(t) = 0\) for \(t\) on \([t_0, t_1]\). This contradicts the assumption of linear independence of \(f_i(t)\). Hence, if \(f_i(t), i = 1, 2, \ldots, n\) are linearly independent, we must have \(\det G(t_0, t_1) \neq 0\). This proves the necessity part. In order to prove the sufficiency, suppose that \(\det G(t_0, t_1) = 0\) but \(f_i(t)\) are linearly dependent on \([t_0, t_1]\). The latter assumption implies \(\beta F(t) = 0\) for a nonzero constant vector \(\beta\) and \(t\) on \([t_0, t_1]\). Furthermore

\[
\beta G(t_0, t_1) = \int_{t_0}^{t_1} \beta F(t)F^T(t) = 0
\]

which contradicts the assumption of nonsingularity of \(G(t_0, t_1)\). Thus, if \(G(t_0, t_1)\) is nonsingular, then \(f_i(t), i = 1, 2, \ldots, n\) are linearly independent on \([t_0, t_1]\).

**Example 3.1**

Consider the functions \(f_1(t) = (1 \quad t)\) and \(f_2(t) = (t^2 \quad 1)\). Determine the linear independence of \(f_1(t)\) and \(f_2(t)\) on \([1, 2]\).

**Solution**

We have \(F(t) = \begin{pmatrix} 1 & t \\ t^2 & 1 \end{pmatrix}\) and

\[
G(1, 2) = \int_1^2 \begin{pmatrix} 1 \\ t \\ t^2 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & t \\ t & 1 \\ t^2 & 1 \\ 1 & 1 \end{pmatrix} dt
\]

\[
= \begin{pmatrix} \frac{1}{3} t^3 + t \\ \frac{1}{3} t^3 + \frac{1}{2} t^2 \\ \frac{1}{3} t^2 + \frac{1}{2} t^2 \\ \frac{1}{5} t^5 + t \end{pmatrix}
\]

\[
= \begin{pmatrix} 10/3 & 23/6 \\ 23/6 & 36/5 \end{pmatrix}
\]

Since \(G(1, 2) = \frac{335}{36} \neq 0\), \(f_1(t)\) and \(f_2(t)\) are linearly independent on \([1, 2]\).

For time functions that have continuous derivatives up to order \((n - 1)\), the following lemma provides a simple sufficient condition for linear independence.

**Lemma 3.1.** Suppose that the derivatives of \(f_i(t), i = 1, 2, \ldots, n\) up to order \((n - 1)\) exist and are continuous on \([t_0, t_1]\). Then the functions \(f_i(t)\) are linearly independent on \([t_0, t_1]\) if there exists some time \(t_a\) in \([t_0, t_1]\) such that

\[
\text{rank } \left[ F(t_a) \ F^{(1)}(t_a) \ldots \ F^{(n-1)}(t_a) \right] = n
\]

(3.4)

where

\[
F^{(j)}(t_a) = \left. \frac{d^j F(t)}{dt^j} \right|_{t=t_a}, \quad j = 1, 2, \ldots, n - 1
\]

**Proof**

The proof is by contradiction. Assume that Eq. (3.4) holds, but \(f_i(t)\) are linearly dependent on \([t_0, t_1]\). The latter implies that \(\beta F(t) = 0\) for some nonzero
1 \times n vector \beta and for all t on \([t_0, t_1]\) and that \(\beta F^j(t) = 0; j = 1, 2, \ldots, n - 1\) for all t on \([t_0, t_1]\). Thus, we have
\[
\beta [F(t_a) F^{(1)}(t_a) \cdots F^{(n-1)}(t_a)]
\] (3.5)
for \(t_a\) on \([t_0, t_1]\), and thus the n rows of the \(n \times nm\) matrix in Eq. (3.5) are linearly dependent which contradicts the assumption that Eq. (3.4) holds. We conclude that if condition Eq. (3.4) is satisfied, the functions \(f_i(t), i = 1, 2, \ldots, n\) must be linearly independent.

Note that Theorem 3.1 gives a necessary and sufficient condition for linear independence of time functions, whereas Lemma 3.1 provides only a sufficient condition. However, the lemma is easier to apply, as demonstrated by the following example.

Example 3.2

Determine the linear independence of functions given in Ex. 3.1 using Lemma 3.1.

Solution We have \(F^{(1)}(t) = \begin{pmatrix} 0 & 1 \\ 2t & 0 \end{pmatrix}\). Choose \(t_a = 1.5\) which is on \([1, 2]\), then
\[
\text{rank} \begin{pmatrix} F(1.5) & F^{(1)}(1.5) \end{pmatrix} = \text{rank} \begin{pmatrix} 1 & 1.5 & 0 & 1 \\ 2.25 & 1 & 3 & 0 \end{pmatrix} = 2
\]

Thus \(f_1(t)\) and \(f_2(t)\) are linearly independent on \([1, 2]\).

3.3 CONTROLLABILITY

Consider the linear time-varying system
\[
\dot{x}(t) = A(t)x(t) + B(t)u(t)
\] (3.6)
where \(x(t)\), and \(u(t)\) are \(n \times 1\) and \(m \times 1\) vectors of state and input, respectively. The state vector \(x(t_0)\) of the system Eq. (3.6) is said to be controllable at \(t_0\) if some input \(u(t)\) exists on \([t_0, t_1]\) which transfers \(x(t_0)\) to any desired state \(x(t_1)\) at time \(t_1\), otherwise the state \(x(t_0)\) is said to be uncontrollable at \(t_0\).

Note that there is no constraint on the magnitude of \(u(t)\) in this definition of controllability. Also note that if \(x(t_0)\) is controllable at \(t_0\), it can be transferred to any \(x(t_1)\), if and only if, it can be transferred to the origin of state space at time \(t_1\), that is, \(x(t_1) = 0\) (Reader can show this as an exercise). We can therefore assume, with no loss of generality, that the desired state is the origin.

Consider now the following theorem, which provides a condition for controllability of Eq. (3.6) in terms of the transition matrix \(\phi(t_0, t)\) and the input matrix \(B(t)\).

Theorem 3.2. The linear time varying system Eq. (3.6) is controllable at time \(t_0\), if and only if, for a \(t_1 > t_0\) the rows of \(F(t) = \phi(t_0, t)B(t)\) are linearly independent on \([t_0, t_1]\), where \(\phi(t, t_0)\) is the system transition matrix.
Proof. The solution of Eq. (3.6) at $t = t_1$ is
\begin{equation}
\begin{aligned}
x(t_1) &= \phi(t_1, t_0)x(t_0) + \int_{t_0}^{t_1} \phi(t_1, \tau)B(\tau)u(\tau)d\tau \\
&= \phi(t_1, t_0)x(t_0) + \phi(t_1, t_0)\int_{t_0}^{t_1} \phi(t_0, \tau)B(\tau)u(\tau)d\tau
\end{aligned}
\end{equation}
(3.7)

Now if the rows of $F(t) = \phi(t_0, t)B(t)$ are linearly independent on $[t_0, t_1]$, we have from Theorem 3.1 that the constant $n \times n$ matrix
\begin{equation}
G(t_0, t_1) = \int_{t_0}^{t_1} \phi(t_0, \tau)B(\tau)B^T(\tau)\phi^T(t_0, \tau)d\tau
\end{equation}
(3.8)
is nonsingular. We now show that the following input $u(t)$ transfers $x(t_0)$ to the origin at time $t_1$, that is, $x(t_1) = 0$
\begin{equation}
u(t) = -B^T(t)\phi^T(t_0, t)G^{-1}(t_0, t_1)x(t_0)
\end{equation}
(3.9)

Substituting Eq. (3.9) into Eq. (3.7), we have
\begin{equation}
x(t_1) = \phi(t_1, t_0)x(t_0) - \phi(t_1, t_0)\int_{t_0}^{t_1} \phi(t_0, \tau)B(\tau)B^T(\tau)\phi^T(t_0, \tau)d\tau G^{-1}(t_0, t_1)x(t_0)
\end{equation}
\begin{equation}
= \phi(t_1, t_0)(x(t_0) - G(t_0, t_1)G^{-1}(t_0, t_1)x(t_0))
\end{equation}
\begin{equation}
= 0
\end{equation}

Thus, if the rows of $\phi(t_0, t)B(t)$ are linearly independent on $[t_0, t_1]$, Eq. (3.8) is nonsingular and Eq. (3.9) transfers $x(t_0)$ to the origin at $t = t_1$, which proves the sufficiency. The proof of necessity is by contradiction. Suppose that Eq. (3.6) is controllable at $t_0$ but the rows of $F(t) = \phi(t_0, t)B(t)$ are linearly dependent on $[t_0, t_1]$. Then a constant nonzero $\beta$ can be found such that $\beta \phi(t_0, t)B(t) = 0$ for $t$ on $[t_0, t_1]$. Since for controllability any initial state $x(t_0)$ must be transferable to $x(t_1) = 0$, we choose $x(t_0) = \beta^T$. Equation (3.7) becomes
\begin{equation}
0 = \phi(t_1, t_0)\beta^T + \phi(t_1, t_0)\int_{t_0}^{t_1} \phi(t_0, \tau)B(\tau)u(\tau)d\tau
\end{equation}
or
\begin{equation}
0 = \beta^T + \int_{t_0}^{t_1} \phi(t_0, \tau)B(\tau)u(\tau)d\tau
\end{equation}
(3.10)
since $\phi(t_1, t_0)$ is nonsingular. Premultiplying Eq. (3.10) by $\beta$, we obtain
\begin{equation}
0 = \beta\beta^T + \int_{t_0}^{t_1} \beta \phi(t_0, \tau)B(\tau)u(\tau)d\tau
\end{equation}
(3.11)
Due to the assumption of linear dependence of the rows of $\phi(t_0, \tau)B(\tau)$, we have $\beta\phi(t_0, \tau)B(\tau) = 0$, and Eq. (3.11) reduces to $\beta\beta^T = 0$ or $\beta = 0$ which is a contradiction.
In the case of linear time-invariant systems, \( F(t) = \phi(t_0, t)B(t) = e^{A(t_0-t)B} \) and due to the time-invariance, controllability need not be specified at time \( t_0 \). Thus, if a linear time-invariant system is controllable at some \( t_0 \), it is controllable at any time. In this case, we set \( t_0 = 0 \) for convenience and consider \( F(t) = e^{-At}B \). Furthermore, for linear time-invariant systems, the condition for controllability can be stated directly in terms of the matrices \( A \) and \( B \) as follows:

**Lemma 3.2.** The linear time-invariant system \( \dot{x}(t) = Ax(t) + Bu(t) \) is controllable, if and only if the \( n \times nm \) controllability matrix

\[
R_c = (B \ AB \cdots A^{n-1}B)
\]  

has rank \( n \).

**Proof.** According to Theorem 3.2, \((A, B)\) is controllable, if and only if the rows of \( F(t) = e^{At}B \) are linearly independent. Using Lemma 3.1 with \( t_a = 0 \), we have

\[
\text{rank}(e^{-At}B - e^{-At}AB \cdots (-1)^{n-1}e^{-At}A^{n-1}B \cdots)|_{t_a=0} = n
\]

or

\[
\text{rank } (B - AB \cdots (-1)^{n-1}A^{n-1}B \cdots) = n
\]

Note that by Cayley-Hamilton theorem, \( A^n, A^{n+1}, \ldots \), are linear combinations of \( I, A, \ldots, A^{n-1} \) and that the negative signs in Eq. (3.13) have no effect on the rank. Thus Eq. (3.13) reduces to

\[
\text{rank } (B \ AB \cdots A^{n-1}B) = n.
\]  

(3.14)

It can be shown (see Prob. 3.12) that in determining the rank condition Eq. (3.14), it is sufficient to consider powers of \( A \) up to \( n - m \), where \( m \) is the number of independent inputs, that is, \( m = \text{rank } B \). Thus we have

\[
\text{rank } R_c = \text{rank } (B \ AB \cdots A^{n-m}B)
\]  

(3.15)

**Example 3.3**

Determine the controllability of the system

\[
\dot{x}(t) = \frac{1}{12} \begin{pmatrix} 5 & 1 \\ 1 & 5 \end{pmatrix} x(t) + \begin{pmatrix} e^{\nu_2} \\ e^{\nu_2} \end{pmatrix} u(t)
\]

**Solution** Since \( B(t) \) is time-varying, we apply Theorem 3.2, that is, for controllability, the matrix \( G(t_0, t_1) = \int_{t_0}^{t_1} \phi(t_0, \tau)B(\tau)B^T(\tau) \phi^T(t_0, \tau)d\tau \) must be nonsingular. Furthermore, since matrix \( A \) is constant, we can evaluate \( \phi(t_0, t) \) using one of the methods described in Section 2.6. This gives

\[
\phi(0, t) = \frac{1}{2} \begin{pmatrix} e^{-\nu_2} + e^{-\nu_3} & e^{-\nu_2} - e^{-\nu_3} \\ e^{-\nu_2} - e^{-\nu_3} & e^{-\nu_2} + e^{-\nu_3} \end{pmatrix}
\]
Now
\[ G(t_0, t_1) = \phi(t_0, t_1) \int_{t_0}^{t_1} \phi^{-1}(\tau, 0) B(\tau) B^T(\tau) B^T(\tau) \left[ \phi^{-1}(\tau, 0) \right]^T \, d\tau \phi^T(t_0, 0) \]
and
\[ \phi^{-1}(\tau, 0) B(\tau) = \phi(-\tau, 0) B(\tau) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \]
hence
\[ \det G(t_0, t_1) = \det \phi(t_0, 0) \det \begin{pmatrix} t_1 - t_0 & t_1 - t_0 \\ t_1 - t_0 & t_1 - t_0 \end{pmatrix} \det \phi^T(t_1, 0) = 0 \]
for all \( t_0 \) and \( t_1 \). The system is, therefore, not controllable. Alternatively, using Lemma 3.1, we have
\[ F(t) = \phi(t_0, t) B(t) = \begin{pmatrix} e^{t - \frac{t_0}{2}} \\ e^{t - \frac{t_0}{2}} \end{pmatrix}, \quad \frac{dF(t)}{dt} = \begin{pmatrix} e^{t - \frac{t_0}{2}} \\ e^{t - \frac{t_0}{2}} \end{pmatrix} \]
and rank \( \begin{pmatrix} F(t) \quad \frac{dF(t)}{dt} \end{pmatrix} \) = 1 < 2 for all \( t \) and \( t_0 \), indicating that the system is not controllable.

**Example 3.4**
Consider the linear time-invariant system,
\[ \dot{x}(t) = \text{diag}(-1, -2, -3, -4) x(t) + \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} u(t) \]
Determine controllability of the system.

**Solution** For this system, we use the controllability criterion Eq. (3.15), that is,
\[ \text{rank } (B, AB, \ldots, A^{n-m}B) = \text{rank } (B, AB) \]
\[ \begin{pmatrix} 1 & 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -2 & 0 \\ 0 & 1 & 0 & 0 & -3 & 0 \\ 0 & 0 & 1 & 0 & 0 & -4 \end{pmatrix} = 4. \]
Thus, the system is controllable.

**CAD Example 3.1**
An approximate linear model of the dynamics of an aircraft is
\[ \begin{pmatrix} \dot{\rho}(t) \\ \dot{\gamma}(t) \\ \dot{\beta}(t) \\ \dot{\gamma}(t) \end{pmatrix} = \begin{pmatrix} -10 & 0 & -10 & 0 \\ 0 & -0.7 & 9 & 0 \\ 0 & -1 & -0.7 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \rho(t) \\ \gamma(t) \\ \beta(t) \\ \gamma(t) \end{pmatrix} + \begin{pmatrix} 20 & 2.8 \\ 0 & -3.13 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \delta_a(t) \\ \delta_c(t) \end{pmatrix} \]
where \( r(t) \) and \( y(t) \) are incremental roll and yaw rates, \( \beta(t) \) and \( \gamma(t) \) are incremental side slip and roll angles; \( \delta_a(t) \) and \( \delta_r(t) \) are incremental changes in the aileron and rudder angles, respectively.

1. Is the aircraft controllable?
2. Is it controllable using only \( \delta_a(t) \) as the control input?
3. Is it controllable through \( \delta_r(t) \) only?

A computer solution using TIMDOM/PC follows.

1.

<<CONOBS>> DETERMINES WHETHER A MULTIVARIABLE LINEAR TIME-ININVARIANT CONTINUOUS-TIME SYSTEM \( dx/dt = Ax + Bu \), \( y = Cx + Du \) OR DISCRETE-TIME SYSTEM \( x(k+1) = Ax(k) + Bu(k) \), \( y(k) = Cx(k) + Du(k) \) IS COMPLETELY CONTROLLABLE OR COMPLETELY OBSERVABLE.

Matrix dimensions are: \( A:nxn \), \( B:nxm \), \( C:rxn \), \( D:rxm \)
Matrix \( D \) does not influence system controllability or observability

MATRIX DIMENSIONS: \( n = 4 \), \( m = 2 \), \( r = 2 \)

Matrix A

\[
\begin{bmatrix}
-0.100 \cdot 10^2 & 0.000 \cdot 10^0 & -0.100 \cdot 10^2 & 0.000 \cdot 10^0 \\
0.000 \cdot 10^0 & -0.700 \cdot 10^0 & 0.900 \cdot 10^1 & 0.000 \cdot 10^0 \\
0.000 \cdot 10^0 & -0.100 \cdot 10^0 & -0.700 \cdot 10^0 & 0.000 \cdot 10^0 \\
0.100 \cdot 10^1 & 0.000 \cdot 10^0 & 0.000 \cdot 10^0 & 0.000 \cdot 10^0 \\
\end{bmatrix}
\]

Matrix B

\[
\begin{bmatrix}
0.200 \cdot 10^1 & 0.280 \cdot 10^1 \\
0.000 \cdot 10^0 & -0.313 \cdot 10^0 \\
0.000 \cdot 10^0 & 0.000 \cdot 10^0 \\
0.000 \cdot 10^0 & 0.000 \cdot 10^0 \\
\end{bmatrix}
\]

CONTROLLABILITY Matrix Qc =

\[
\begin{bmatrix}
2.0 \cdot 10^0 & 2.8 \cdot 10^0 & -2.0 \cdot 10^2 & -2.0 \cdot 10^2 & 2.8 \cdot 10^0 & 2.4 \cdot 10^2 & 2.4 \cdot 10^2 & -2.0 \cdot 10^0 & -2.4 \cdot 10^3 & -2.4 \cdot 10^3 \\
0.0 \cdot 10^1 & -3.1 \cdot 10^1 & 0.0 \cdot 10^2 & 2.1 \cdot 10^2 & 0.0 \cdot 10^2 & 2.6 \cdot 10^2 & 0.0 \cdot 10^2 & 5.8 \cdot 10^1 \\
0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & -2.4 \cdot 10^1 & -2.4 \cdot 10^1 & -2.4 \cdot 10^1 & -2.4 \cdot 10^1 \\
0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & 0.0 \cdot 10^0 & -2.4 \cdot 10^1 & -2.4 \cdot 10^1 & -2.4 \cdot 10^1 & -2.4 \cdot 10^1 \\
\end{bmatrix}
\]

THE SYSTEM IS COMPLETELY CONTROLLABLE
The Controllability Matrix has Rank 4 = n
2.

**CONTROLLABILITY Matrix** $Q_c =$

\[
\begin{bmatrix}
2.000E+01 & -2.000E+02 & 2.000E+03 & -2.000E+04 \\
0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
0.000E+00 & 2.000E+01 & -2.000E+02 & 2.000E+03
\end{bmatrix}
\]

**THE SYSTEM IS NOT COMPLETELY CONTROLLABLE**
The Controllability Matrix has Rank $2 < n$

3.

**CONTROLLABILITY Matrix** $Q_c =$

\[
\begin{bmatrix}
2.800E+00 & -2.800E+01 & 2.487E+02 & -2.443E+03 \\
-3.130E+00 & 2.191E+00 & 2.664E+01 & -5.808E+01 \\
0.000E+00 & 3.130E+00 & -4.382E+00 & -2.357E+01 \\
0.000E+00 & 2.800E+00 & -2.800E+01 & 2.487E+02
\end{bmatrix}
\]

**THE SYSTEM IS COMPLETELY CONTROLLABLE**
The Controllability Matrix has Rank $4 = n$

### 3.4 OBSERVABILITY

Consider the linear time-varying system

\[
\dot{x}(t) = A(t)x(t) + B(t) u(t)
\]

\[
y(t) = C(t) x(t) + D(t) u(t)
\]

where $x(t)$, $u(t)$, and $y(t)$ are the $n \times 1$, $m \times 1$, and $r \times 1$ vectors of state, input and output, respectively. The system Eq. (3.16) is said to be observable at time $t_0$, if every state vector $x(t_0)$ can be determined from the knowledge of the input vector $u(t)$ and the output vector $y(t)$ over the finite time interval $[t_0, t_1]$, otherwise the system is said to be unobservable.

Note that it is assumed that the matrices $A$, $B$, $C$, and $D$ are known. The following theorem provides conditions for observability of the system Eq. (3.16).

**Theorem 3.3** The system Eq. (3.16) is observable, if and only if the columns of the matrix $C(t) \phi(t, t_0)$ are linearly independent on $[t_0, t_1]$ or equivalently, if and only if the matrix
\[ M(t_0, t_1) = \int_{t_0}^{t_1} \phi^T(t, t_0) C^T(t) C(t) \phi(t, t_0) dt \]  

(3.17)

is nonsingular

**Proof.** The proof of necessity is by contradiction. The response \( y(t) \) is

\[ y(t) = C(t) \phi(t, t_0) x(t_0) + C(t) \int_{t_0}^{t} \phi(t, \tau) B(\tau) u(\tau) d\tau + D(t) u(t) \]  

(3.18)

Hence,

\[ C(t) \phi(t, t_0) x(t_0) = y(t) - C(t) \int_{t_0}^{t} \phi(t, \tau) B(\tau) u(\tau) d\tau - D(t) u(t) \]

\[ \Delta \]  

(3.19)

\[ \Delta = \hat{y}(t) \]

where \( \hat{y}(t) \) is a known vector in view of the fact that \( y(t), u(t), \) and \( A, B, C, D \) are known quantities. We now form

\[ \int_{t_0}^{t_1} \hat{y}^T(t) \hat{y}(t) dt = \int_{t_0}^{t_1} x^T(t_0) \phi(t, t_0) C^T(t) C(t) \phi(t, t_0) x(t_0) dt \]

Suppose now that the system is observable but \( M(t_0, t_1) \) is singular. This implies that a state \( x(t_0) \neq 0 \) exists such that \( M(t_0, t_1) x(t_0) = 0 \), implying that \( \hat{y}(t) = 0 \) for all \( t \) on \([t_0, t_1] \). Such an \( x(t_0) \) cannot be determined from the observation of \( \hat{y}(t) = 0 \). This contradicts the assumption of observability. Thus, if the system is observable, \( M(t_0, t_1) \) is nonsingular.

In order to prove the sufficiency, we multiply both sides of Eq. (3.19) by \( \phi^T(t, t_0) C^T(t) \), integrate from \( t_0 \) to \( t_1 \), and use Eq. (3.17) to obtain

\[ M(t_0, t_1) x(t_0) = \int_{t_0}^{t_1} \phi^T(t, t_0) C^T(t) \hat{y}(t) dt \]

If \( M(t_0, t_1) \) is nonsingular, we can find the initial state as

\[ x(t_0) = M^{-1}(t_0, t_1) \int_{t_0}^{t_1} \phi^T(t, t_0) C^T(t) \hat{y}(t) dt \]

which proves the sufficiency.

The condition for the observability of linear time-invariant systems can be expressed directly in terms of the matrices \( A \) and \( C \). This is done by applying Lemma 3.1 to the function \( F(t) = C(t) \phi(t, t_0) \) and using Cayley-Hamilton theorem. The procedure is similar to that discussed for controllability. The following result is readily obtained:
Lemma 3.3. The linear time-invariant system

\[\begin{align*}
\dot{x}(t) &= A x(t) + B u(t) \\
y(t) &= C x(t) + D u(t)
\end{align*}\]  \hspace{1cm} (3.20)

is observable, if and only if the \( nr \times n \) observability matrix

\[R_0 = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix}\]  \hspace{1cm} (3.21)

has rank \( n \).

As with the controllability matrix, it can be shown that

\[\text{rank } R_0 = \text{rank } \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-r} \end{pmatrix}\]  \hspace{1cm} (3.22)

where \( r = \text{rank } C \).

Example 3.5

Consider the system in the observable canonical form

\[\begin{align*}
\dot{x}(t) &= \begin{pmatrix} 0 & 0 & -p_0 \\
1 & 0 & -p_1 \\
0 & 1 & -p_2 \end{pmatrix} x(t) \\
y(t) &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} x(t)
\end{align*}\]

Determine the observability of the system.

Solution The observability matrix is

\[R_0 = \begin{pmatrix} C \\ CA \\ CA^2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\
0 & 1 & -p_2 \\
1 & -p_2 & -p_1 + p_2^2 \end{pmatrix}
\]

\[\det R_0 = -1 \neq 0\] for all \( p_0, p_1, \) and \( p_2 \) and thus the system is observable. Note that the general canonical form Eq. (2.25) can be shown to be observable. This explains the reason for using the terminology "canonical observable form."

CAD Example 3.2

Consider the linearized model of an aircraft given in CAD Ex. 3.1. In which of the following cases is the system observable?

1. Roll rate \( r(t) \) is the only measurable quantity.
2. Yaw rate \( y(t) \) and sideship angle \( \beta(t) \) are measurable quantities.
3. Roll angle \( \gamma(t) \) is the only measurable quantity.

A TIMDOM solution of this problem is shown below:
1. 

**<<CONOBSS>>** DETERMINES WHETHER A MULTIVARIABLE LINEAR TIME-ININVARIANT CONTINUOUS-TIME SYSTEM \( \frac{dx}{dt} = Ax + Bu \), \( y = Cx + Du \) OR DISCRETE-TIME SYSTEM \( x(k+1) = Ax(k) + Bu(k) \), \( y(k) = Cx(k) + Du(k) \) IS COMPLETELY CONTROLLABLE OR COMPLETELY OBSERVABLE.

Matrix dimensions are: \( A: nxn \), \( B: nxm \), \( C: rxn \), \( D: rxm \)
Matrix D does not influence system controllability or observability

**MATRIX DIMENSIONS:** \( n = 4 \), \( m = 1 \), \( r = 1 \)

**Matrix A**

\[
\begin{bmatrix}
-1.00E+02 & 0.000E+00 & -1.00E+02 & 0.000E+00 \\
0.000E+00 & -7.00E+00 & 0.900E+01 & 0.000E+00 \\
0.000E+00 & -1.00E+00 & -7.00E+00 & 0.000E+00 \\
0.100E+01 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
\end{bmatrix}
\]

**Matrix B**

\[
\begin{bmatrix}
0.200E+02 & 0.280E+01 \\
0.000E+00 & -0.313E+01 \\
0.000E+00 & 0.000E+00 \\
0.000E+00 & 0.000E+00 \\
\end{bmatrix}
\]

**Matrix C**

\[
\begin{bmatrix}
0.100E+01 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
\end{bmatrix}
\]

**OBSERVABILITY Matrix Qo =**

\[
\begin{bmatrix}
1.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
-1.000E+01 & 0.000E+00 & -1.000E+01 & 0.000E+00 \\
1.000E+02 & 1.000E+01 & 1.070E+02 & 0.000E+00 \\
-1.000E+03 & -1.140E+02 & -9.849E+02 & 0.000E+00 \\
\end{bmatrix}
\]

THE SYSTEM IS NOT COMPLETELY OBSERVABLE
The Observability Matrix has Rank 3 < n

2. 

**OBSERVABILITY Matrix Qo =**

\[
\begin{bmatrix}
0.000E+00 & 1.000E+00 & 0.000E+00 & 0.000E+00 \\
0.000E+00 & 0.000E+00 & 1.000E+00 & 0.000E+00 \\
0.000E+00 & -7.000E-01 & 9.000E+00 & 0.000E+00 \\
0.000E+00 & -1.000E+00 & -7.000E-01 & 0.000E+00 \\
0.000E+00 & -8.510E+00 & -1.260E+01 & 0.000E+00 \\
0.000E+00 & 1.400E+00 & -8.510E+00 & 0.000E+00 \\
0.000E+00 & 1.856E+01 & -6.777E+01 & 0.000E+00 \\
0.000E+00 & 7.530E+00 & 1.856E+01 & 0.000E+00 \\
\end{bmatrix}
\]
THE SYSTEM IS NOT COMPLETELY OBSERVABLE
The Observability Matrix has Rank $2 < n$

3.

**OBSERVABILITY Matrix $Q_0$**

\[
\begin{array}{cccc}
0.000E+00 & 0.000E+00 & 0.000E+00 & 1.000E+00 \\
1.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
-1.000E+01 & 0.000E+00 & -1.000E+01 & 0.000E+00 \\
1.000E+02 & 1.000E+01 & 1.000E+01 & 0.000E+00 \\
\end{array}
\]

THE SYSTEM IS COMPLETELY OBSERVABLE

The Observability Matrix has Rank $4 = n$

We have seen that there is a similarity between the criteria for controllability and observability. In particular, controllability criterion requires determining the linear independence of the rows of $\phi(t_0, t)B(t)$ while for observability the columns of $C(t)\phi(t, t_0)$ must be linearly independent. This similarity was first noted by Kalman who introduced the concept of duality and stated the following theorem:

**Theorem 3.4.** The system $\dot{x}(t) = A(t)x(t) + B(t)u(t)$, $y(t) = C(t)x(t) + D(t)u(t)$ is controllable (observable) at time $t_0$, if and only if the dual system $\dot{z}(t) = -A^T(t)z(t) + C^T(t)v(t)$, $w(t) = B^T(t)z(t) + D^T(t)v(t)$ is observable (controllable) at time $t_0$.

**Proof.** According to Theorem 3.2 the system $[A(t), B(t), C(t), D(t)]$ is controllable, if and only if the rows of $\phi_x(t_0, t)B(t)$ are linearly independent, on $[t_0, t_1]$, where $\phi_x(t, t_0)$ is the transition matrix of the system. On the other hand, according to Theorem 3.3 the dual system $[-A^T(t), C^T(t), B^T(t), D^T(t)]$ is observable, if and only if the columns of $B^T(t)\phi_z(t, t_0)$ are linearly independent on $[t_0, t_1]$, where $\phi_z(t, t_0)$ is the transition matrix of the dual system. Now columns of $B^T(t)\phi_z(t, t_0)$ are the rows of $\phi^T_z(t, t_0)B(t)$, and it is easy to show that $\phi^T_z(t, t_0) = \phi_x(t_0, t)$. Thus, the system $[A(t), B(t), C(t), D(t)]$ is controllable, if and only if the dual system $[-A^T(t), C^T(t), B^T(t), D^T(t)]$ is observable. It can similarly be shown that observability of a system requires controllability of its dual system.

Consider now the effect of nonsingular transformation $z = Px$ on the controllability and observability of the linear time-invariant system $(A, B, C, D)$ described by Eq. (3.20). The new system is $(\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$, where $\tilde{A} = PAP^{-1}$, $\tilde{B} = PB$, $\tilde{C} = CP^{-1}$, and $\tilde{D} = D$.

The controllability matrix of the new system is

\[
\tilde{K}_c = (\tilde{B} \tilde{A} \tilde{B} \cdots \tilde{A}^{n-1}\tilde{B}) = P(B AB \cdots A^{n-1}B) = P R_c
\]
Since $P$ is nonsingular, rank $\hat{R}_c = \text{rank } R$. We conclude that controllability of linear time-invariant systems is invariant under nonsingular transformation. Similarly the observability matrix of the new system is

\[
\hat{R}_0 = \begin{pmatrix}
\hat{C} \\
\hat{CA} \\
\vdots \\
\hat{CA}^{n-1}
\end{pmatrix} = \begin{pmatrix}
C \\
CA \\
\vdots \\
CA^{n-1}
\end{pmatrix} P^{-1} = R_0 P^{-1}
\]

Hence rank $\hat{R}_0 = \text{rank } R_0$ and observability is also invariant under nonsingular transformation.

3.5 DISCRETE-TIME SYSTEMS

Consider the linear time-varying discrete system

\[
x(k + 1) = A(k) x(k) + B(k) u(k)
\]

\[
y(k) = C(k) x(k) + D(k) u(k)
\]

with the initial state $x(k_0)$. The definitions of controllability and observability of discrete-time systems are similar to those of continuous-time systems, except that the time $t$ is replaced by $k$ and the time interval $[t_0, t_1]$ is replaced by $[k_0, k_1]$. As a result, the criteria for controllability and observability of discrete-time systems are similar to those of continuous-time cases. The following theorem can be stated for discrete-time systems. The proof is similar to those of Theorem 3.2 and 3.3.

Theorem 3.5. Consider the discrete-time system Eq. (3.23):

1. The system is controllable at $k_0$ if and only if the rows of $\Phi(k_0, k)B(k)$ are linearly independent on $[k_0, k_1]$ or equivalently, if and only if the matrix

\[
G(k_0, k_1) = \sum_{j=k_0}^{k_1} \Phi(k_0, j)B(j)B^T(j) \Phi^T(k_0, j)
\]

is nonsingular

2. The system is observable at $k_0$, if and only if the columns of $C(k)\Phi(k, k_0)$ are linearly independent on $[k_0, k_1]$ or equivalently, if and only if

\[
M(k_0, k_1) = \sum_{j=k_0}^{k_1} \Phi(j, k_0)C(j)C(j)^T \Phi(j, k_0)
\]

is nonsingular.

Controllability and observability of the linear time-invariant discrete-time system
\[ x(k + 1) = A \ x(k) + B \ u(k) \]
\[ y(k) = C \ x(k) + Du(k) \]

can be determined by applying the criteria of Lemmas 3.2 and 3.3, that is, by checking rank conditions Eqs. (3.15) and (3.21), respectively.

### 3.6 FREQUENCY-DOMAIN CRITERIA

In previous sections, we established criteria for controllability and observability in terms of time-domain description of the system. In this section, we discuss the effect of controllability and observability on the transfer function of the system.

Consider the system \((A, B, C, D)\) and for convenience assume that the eigenvalues \(\lambda_1, \lambda_2, \ldots, \lambda_n\) of \(A\) are distinct. Since controllability and observability are invariant under nonsingular transformation, with no loss of generality, we can assume that \(A\) is in the diagonal form \(A = \text{diag} (\lambda_1, \lambda_2, \ldots, \lambda_n)\). Furthermore, \(D\) has no effect on controllability and observability and can be set to zero. We can, therefore, write the state equation of the system as

\[
\begin{align*}
\dot{x}(t) &= \text{diag} (\lambda_1, \lambda_2, \ldots, \lambda_n) \ x(t) + Bu(t) \\
y(t) &= C \ x(t)
\end{align*}
\tag{3.26}
\]

Let \(B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}\) and \(C = (c_1, c_2, \ldots, c_n)\) where \(b_i\) and \(c_i\) are the \(i\)th row and the \(i\)th column of \(B\) and \(C\), respectively. Equation (3.26) is now

\[
\begin{align*}
\dot{x}_1(t) &= \lambda_1 \ x_1(t) + b_1 \ u(t) \\
\dot{x}_2(t) &= \lambda_2 \ x_2(t) + b_2 \ u(t) \\
&\quad \vdots \\
\dot{x}_n(t) &= \lambda_n \ x_n(t) + b_n \ u(t) \\
y(t) &= (c_1, c_2, \ldots, c_n) \ x(t)
\end{align*}
\tag{3.27}
\]

Equation (3.27) consists of \(n\) first order state equations where the states \(x_1(t), x_2(t), \ldots, x_n(t)\) are isolated from one another. Consequently, if \(b_i\) is zero, the corresponding state \(x_i(t)\) becomes unaffected by the input \(u(t)\) and is therefore uncontrollable. Similarly, if \(c_i\) is zero, \(x_i(t)\) does not appear in any of the outputs and \(x_i(t)\) becomes unobservable. We conclude that the system Eq. (3.27) is controllable (observable), if and only if \(b_i \neq 0, c_i \neq 0\), \(i = 1, 2, \ldots, n\). Note that this conclusion applies only when \(A\) is in the diagonal form.

The transfer-function matrix of Eq. (3.26) is
\[ G(s) = C(sI - A)^{-1}B \]
\[ = (c_1, c_2, \ldots, c_n)\text{diag}\left(\frac{1}{s - \lambda_1}, \ldots, \frac{1}{s - \lambda_n}\right) \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{3.28} \]
\[ = \sum_{i=1}^{n} \frac{c_i b_i}{(s - \lambda_i)} \]

Suppose that a row of \( B \) say \( b_j \) (or a column of \( C \) say \( c_j \)) is zero, implying that the corresponding mode \( 1/(s - \lambda_i) \), that is, \( e^{\lambda_i t} \) is uncontrollable (unobservable).

The transfer-function matrix Eq. (3.28) is now
\[ G(s) = \sum_{i=1}^{n} \frac{c_i b_i}{(s - \lambda_i)} \tag{3.29} \]
that is, in forming \( G(s) \), the pole at \( s = \lambda_j \) is cancelled by a zero at \( s = \lambda_j \) in all the elements of \( G(s) \). We can, therefore, state the following result.

**Lemma 3.4.** Consider the system \( (A, B, C) \), where the eigenvalues of \( A \) are distinct. The system is controllable and observable if, and only if, there is no common pole-zero cancellations in all elements of the transfer-function matrix \( G(s) = C(sI - A)^{-1}B \).

Note that for checking controllability and observability separately, we form \( G_c(s) = (sI - A)^{-1}B \) (or \( G_o(s) = C(sI - A)^{-1} \)) and determine pole-zero cancellations in all elements of \( G_c(s) \) or \( G_o(s) \).

Frequency-domain criteria for controllability and observability of systems with repeated eigenvalues is more involved and requires determining pole-zero cancellations in the system and all its square subsystems. This will not be pursued here.

**Example 3.6**

Consider the system
\[
\begin{pmatrix} 3 & -0.5 & -0.5 \\ -2 & 2.5 & -0.5 \\ 6 & -2.5 & 0.5 \end{pmatrix} x + \begin{pmatrix} -1 & -1 \\ -2 & -4 \\ 0 & 2 \end{pmatrix} u
\]
\[ y = \begin{pmatrix} -9 & 3.5 & 1.5 \\ 5 & -2 & -1 \end{pmatrix} x \]

Determine if the system is controllable.

**Solution** The characteristic polynomial is \( p(s) = (s - 1)(s - 2)(s - 3) \) and thus the eigenvalues are distinct. The transfer-function matrix is
\[ G(s) = C(sI - A)^{-1}B = \frac{2(s - 1)(s - 2) - 2(s - 1)(s - 2)}{(s - 1)(s - 2)(s - 3)} \]
Since all the elements of $G(s)$ have pole-zero cancellations at $s = 1$ and $s = 2$, the system is uncontrollable or unobservable or both. Furthermore,

$$G_c(s) = (sI - A)^{-1}B = \frac{1}{(s - 1)(s - 2)(s - 3)}\begin{pmatrix}0 & 0 \\ (s - 1)(s - 3) & (s - 1)(s - 2) \\ (s - 1)(s - 2) & (s - 1)(s - 2)\end{pmatrix}$$

Hence the mode at $s = 1$ is uncontrollable since there is pole-zero cancellations at $s = 1$ in all elements of $G_c(s)$. It is also concluded that the mode at $s = 2$ is unobservable, which can be verified by forming $G_o(s) = C(sI - A)^{-1}$.

Now applying the linear transformation $z = Px = Mx$, where $M$ is the modal matrix (see Appendix A), to the system, we obtain

$$\dot{z} = \begin{pmatrix}1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3\end{pmatrix}z + \begin{pmatrix}0 & 0 \\ 1 & 1 \\ 1 & -1\end{pmatrix}u$$

$$y = \begin{pmatrix}1 & 0 & 2 \\ -1 & 0 & -1\end{pmatrix}z$$

This shows that the state $z_1$ is uncontrollable and the state $z_2$ is unobservable which confirms the results obtained by the frequency-domain criterion.

**CAD Example 3.3**

Determine the controllability and observability of the following system by determining pole-zero cancellations in the transfer-function matrix.

$$\dot{x} = \begin{pmatrix}1 & 2 & -0.1 & -0.5 \\ 3 & -1.5 & 1 & 0.6 \\ 0 & 1.2 & 2 & -0.8 \\ 12 & -1.6 & 2.2 & 1.8\end{pmatrix}x + \begin{pmatrix}1 & -2 \\ 0 & -1 \\ 0 & 1 \\ 2 & 1.5\end{pmatrix}u$$

$$y = \begin{pmatrix}1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1\end{pmatrix}x$$

Below is a MATRIXx solution of this problem:

\[\Diamond A = \begin{bmatrix}1 & 2 & -1 & -.5 \\ 3 & -1.5 & 1 & .6 \\ 0 & 1.2 & 2 & -.8 \\ 12 & -1.6 & 2.2 & 1.8\end{bmatrix};\]
\[\Diamond B = \begin{bmatrix}1 & -2;0 & -1;0 \\ 1;2 & 1.5;\end{bmatrix};C = \begin{bmatrix}1 & 0 & 0 & 0;0 & 0 & 0 & 1\end{bmatrix};D = 0*ONES(2,2);\]
\[\Diamond S = [A, B; C, D];\]
\[\Diamond [NUM, DEN] = TFORM(S, 4)\]

\[
\text{DEN} = 
\begin{bmatrix}1.0000 & -3.3000 & 1.7200 & 8.3960 & -6.8560\end{bmatrix}
\]

\[
\text{NUM} = 
\begin{bmatrix}\end{bmatrix}
\]
Sec. 3.6 Frequency-Domain Criteria

COLUMNS 1 THRU 8

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>1.0000</td>
<td>-2.0000</td>
<td>-3.3000</td>
<td>1.7500</td>
<td>2.4800</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>2.0000</td>
<td>1.5000</td>
<td>9.0000</td>
<td>-22.4500</td>
<td>-30.2000</td>
</tr>
</tbody>
</table>

COLUMNS 9 THRU 10

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.7120</td>
<td>-23.1340</td>
</tr>
<tr>
<td>0.2400</td>
<td>153.6600</td>
</tr>
</tbody>
</table>

\# // check poles and zeros
\# ROOTS(DEN)

\[ \text{ANS} = \]

\[ 1.9664 + 1.2640i \]
\[ 1.9665 - 1.2640i \]
\[ -1.4807 \]
\[ 0.8478 \]

\# ROOTS(0 1 -3.3 2.48 1.712)

\[ \text{ANS} = \]

\[ 1.8612 + 0.7671i \]
\[ 1.8612 - 0.7671i \]
\[ -0.4225 \]

\# ROOTS(0 -2 1.75 11.305 -23.134)

\[ \text{ANS} = \]

\[ -2.7358 \]
\[ 1.8054 + 0.9832i \]
\[ 1.8054 - 0.9832i \]

\# ROOTS(0 2 9 -30.2 0.24)

\[ \text{ANS} = \]

\[ -6.7422 \]
\[ 2.2343 \]
\[ 0.0080 \]

\# ROOTS(0 1.5 -22.45 -26.09 153.66)
\[ \text{ANS} = \begin{pmatrix} 15.6596 \\ 2.2345 \\ -2.9275 \end{pmatrix} \]

\[ \triangleright \quad \text{// No pole-zero cancellation. The system is, therefore, both controllable and observable.}\]

### 3.7 Decomposition of Systems

In this section, it will be shown that an uncontrollable time-invariant system $S$ can be decomposed into two systems $S_1$ and $S_2$ where $S_1$ is controllable and $S_2$ is uncontrollable. Furthermore, we will show that the dimension of the controllable subsystem $S_1$ is equal to the rank of the controllability matrix $R_c$. Similar results will be stated for unobservable systems.

**Lemma 3.5.** Consider the $n$-dimensional, uncontrollable time-invariant system $(A, B, C, D)$ where rank $R_c = \nu < n$. The system can be transformed into the following form by a nonsingular transformation $z = Px$;

\[
\begin{pmatrix}
\dot{z}_1 \\
\dot{z}_2
\end{pmatrix} = \begin{pmatrix}
\hat{A}_{11} & \hat{A}_{12} \\
0 & \hat{A}_{22}
\end{pmatrix}
\begin{pmatrix}
z_1 \\
z_2
\end{pmatrix} + \begin{pmatrix}
\hat{B}_1 \\
0
\end{pmatrix} u
\]

\[
y = (\hat{C}_1 \quad \hat{C}_2)
\begin{pmatrix}
z_1 \\
z_2
\end{pmatrix} + Du \tag{3.30}
\]

**Proof.** The transformation matrix is

\[ P^{-1} R = (r_1, r_2, \ldots, r_\nu, \ldots, r_n) \]

where $r_1$, $r_2$, $\ldots$, $r_\nu$ are $\nu$ linearly independent columns of $R_c$ and the last $(n - \nu)$ columns are chosen to make $R$ nonsingular. Now the proof can be established by noting that the $i$th column of the transformed system matrix $\hat{A}$ is the representation of $A r_i$ with respect to $r_1$, $r_2$, $\ldots$, $r_n$. (see Chen, 1984 for a detailed proof).

The counterpart of Lemma 3.5 is stated, without proof, for unobservable systems as follows.

**Lemma 3.6.** Consider the $n$-dimensional unobservable system $(A, B, C, D)$ where rank $R_0 = \mu < n$. The system can be transformed into the following form by a nonsingular transformation $z = Px$

\[
\begin{pmatrix}
\dot{z}_1 \\
\dot{z}_2
\end{pmatrix} = \begin{pmatrix}
\hat{A}_{11} & 0 \\
\hat{A}_{21} & \hat{A}_{22}
\end{pmatrix}
\begin{pmatrix}
z_1 \\
z_2
\end{pmatrix} + \begin{pmatrix}
\hat{B}_1 \\
\hat{B}_2
\end{pmatrix} u \tag{3.31}
\]

\[
y = (\hat{C}_1 \quad 0)
\begin{pmatrix}
z_1 \\
z_2
\end{pmatrix} + Du
\]
where the $\mu$-dimensional subsystem $(\hat{A}_{11}, \hat{B}, \hat{C}_1, D)$ is observable. The transformation matrix is

$$P = \begin{pmatrix} r_1 \\ \vdots \\ r_\mu \\ \vdots \\ r_n \end{pmatrix}$$

(3.32)

where $r_1, \ldots, r_\mu$ are $\mu$ linearly independent rows of the observability matrix $R_0$, and the remaining $(n - \mu)$ rows of $P$ are chosen to make $P$ nonsingular.

**Example 3.7**

Separate the observable subsystem given the unobservable system

$$\dot{x} = \begin{pmatrix} 5 & -2 & -4 \\ 2 & 1 & -2 \\ 1 & -1 & 0 \end{pmatrix} x + \begin{pmatrix} 1 \\ 3 \\ -1 \end{pmatrix} u$$

$$y = \begin{pmatrix} 3 & -1 & -2 \\ -1 & 0 & 0 \end{pmatrix} x$$

Is the observable subsystem controllable?

**Solution** The observability matrix $R_o = \begin{pmatrix} C \\ CA \\ CA^2 \end{pmatrix}$ has

$$\begin{pmatrix} 3 & -1 & -2 \\ -1 & 0 & 0 \\ 11 & -5 & -10 \\ -5 & 2 & 4 \\ 35 & -17 & -34 \\ -17 & 8 & 16 \end{pmatrix}$$

rank 2 and hence the system is not unobservable. The transformation matrix is

$$P = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} 3 & -1 & -2 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where $r_3$ is chosen to make $P$ nonsingular. The new system is

$$\dot{z} = \begin{pmatrix} 5 & 4 & 0 \\ -2 & -1 & 0 \\ 1 & 2 & 2 \end{pmatrix} z + \begin{pmatrix} 2 \\ -2 \\ -1 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} x$$

which is in the form of Eq. (3.31). The observable subsystem is
\[
\begin{align*}
\dot{z}_1 &= \begin{pmatrix} 5 & 4 \\ -2 & -1 \end{pmatrix} z + \begin{pmatrix} 2 & -2 \\ -1 & 1 \end{pmatrix} u \\
y &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} x
\end{align*}
\]

The controllability matrix of this subsystem is

\[
R_c = \begin{pmatrix} 2 & -2 & 6 & -6 \\ -1 & 1 & -3 & 3 \end{pmatrix}
\]

Since rank \( R_c = 1 < 2 \), the observable subsystem is not controllable.

It is to be noted from this example that the decomposition of Lemma 3.5 or Lemma 3.6 does not guarantee that the acquired subsystem is both controllable and observable. A general result concerning decomposition into a system which contains a controllable and observable subsystem is obtained by combining these two lemmas. This result, due to Kalman, is called canonical decomposition theorem and may be stated as follows.

**Theorem 3.6.** Consider the linear time-invariant system \((A, B, C, D)\). There exists a nonsingular transformation \( z = Px \) such that

\[
\begin{pmatrix}
\dot{z}_1 \\
\dot{z}_2 \\
\dot{z}_3 \\
\dot{z}_4
\end{pmatrix} =
\begin{pmatrix}
\hat{A}_{11} & 0 & \hat{A}_{13} & 0 \\
\hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} & \hat{A}_{24} \\
0 & 0 & \hat{A}_{33} & 0 \\
0 & 0 & \hat{A}_{43} & \hat{A}_{44}
\end{pmatrix}
\begin{pmatrix}
z_1 \\
z_2 \\
z_3 \\
z_4
\end{pmatrix} +
\begin{pmatrix}
\hat{B}_1 \\
\hat{B}_2 \\
z_3 \\
z_4
\end{pmatrix} u
\]

\[(3.33)\]

\[
y = (\hat{C}_1 & 0 & \hat{C}_3 & 0)
\begin{pmatrix}
z_1 \\
z_2 \\
z_3 \\
z_4
\end{pmatrix} + D u
\]

where the subsystem

1. \((\hat{A}_{11}, \hat{B}_1, \hat{C}_1)\) is both controllable and observable
2. \((\hat{A}_{22}, \hat{B}_2, 0)\) is controllable but unobservable
3. \((\hat{A}_{33}, 0, \hat{C}_3)\) is uncontrollable but observable
4. \((\hat{A}_{44}, 0, 0)\) is neither controllable nor observable

A block diagram of system Eq. (3.33) showing the interconnection of these four subsystems is given in Fig. 3.1.
3.8 MINIMAL REALIZATION

Given the state-space representation \((A, B, C, D)\) of a system, its transfer-function matrix is uniquely determined as \(G(s) = C(sI - A)^{-1}B + D\). However, the converse is not true, that is, given the transfer function matrix \(G(s)\) of a system, there exists many state-space representations of different orders (dimensions) that realize the same \(G(s)\). In this section, we discuss the connection between the order (dimension) of the state-space realization and the controllability-observability of the system.

A realization \((A, B, C, D)\) of dimension \(n\) of the transfer function matrix \(G(s)\) is said to be minimal or irreducible if no realization of order less than \(n\) exists for \(G(s)\). Note that the minimal realization is not itself unique in the sense that there exists many realizations \((A_1, B_1, C_1, D_1), (A_2, B_2, C_2, D_2), \ldots\) such that \(\dim(A_1) = \dim(A_2) = \ldots\) which realize the same \(G(s)\).

Consider a general \(n\)-dimensional system \((A, B, C, D)\) which according to Theorem 3.6 can be decomposed into a controllable and observable subsystem and three uncontrollable or unobservable subsystems. The transfer function matrix of the system is

\[
G(s) = C(sI - A)^{-1}B + D = \hat{C}(sI - \hat{A})^{-1}\hat{B} + \hat{D}
\]  

(3.34)

where \(\hat{A}, \hat{B}, \hat{C},\) and \(\hat{D}\) are given in Eq. (3.33). On substituting these matrices from Eq. (3.33) into Eq. (3.34) and simplifying we obtain (see Prob. 3.17)

\[
G(s) = \hat{C}_1(sI - \hat{A}_{11})^{-1}\hat{B}_1 + D
\]  

(3.35)
Equation (3.35) implies that the transfer-function matrix contains only the controllable and the observable part of a system and that the matrices of uncontrollable or unobservable parts do not have any effect on the transfer-function matrix. The following theorem shows that the subsystem \((\hat{A}_{11}, \hat{B}_1, \hat{C}_1, D)\) is in fact the minimal realization of \(G(s)\).

**Theorem 3.7.** A realization \((A, B, C, D)\) of the transfer-function matrix \(G(s)\) is minimal, if and only if it is both controllable and observable.

**Proof.** The proof of the necessity part is by contradiction. Suppose that the \(n\)-dimensional system \((A, B, C, D)\) is a minimal realization of \(G(s)\) but is not controllable or observable. Then applying Theorem 3.6, we can find a realization \((A_1, B_1, C_1, D)\) which is both controllable and observable, and has dimension \(n_1 < n\). This implies that \((A, B, C, D)\) is not a minimal realization and contradicts the assumption. To prove the sufficiency, we must show that no system of dimension \(v < n\) exists whose transfer-function matrix is \(G(s)\). Let us assume that such a system exists and denote it by \((\hat{A}, \hat{B}, \hat{C}, D)\), then we have

\[
\hat{C}(sI - \hat{A})^{-1}\hat{B} + D = C(sI - A)^{-1}B + D \quad \text{for all } s
\]

or

\[
\hat{C} e^{\hat{A}t}\hat{B} = C e^{At}B \quad \text{for all } t
\]

which implies

\[
\hat{C} \hat{A}^i\hat{B} = CA^iB \quad i = 0, 1, 2, \ldots \tag{3.36}
\]

Consider the product of the controllability and observability matrices of the system \((A, B, C, D)\),

\[
\begin{pmatrix}
C \\
\vdots \\
CA^{n-1}
\end{pmatrix}
(B \cdots A^{n-1}B) =
\begin{pmatrix}
CB & \cdots & CA^{n-1}B \\
\vdots & \ddots & \vdots \\
CA^{n-1}B & \cdots & CA^{2(n-1)}B
\end{pmatrix}
\]

Now by assumption \(\text{rank } R_o = n\), \(\text{rank } R_c = n\), and using the Sylvester inequality for the matrices \(R_o\) and \(R_c\), (see Appendix A), we have \(\text{rank } (R_o) + \text{rank } (R_c) - n \leq \text{rank } (R_o R_c) \leq \min \{\text{rank } R_o, \text{rank } R_c\}\). Thus we obtain \(\text{rank } R_o R_c = n\).

For the realization \((\hat{A}, \hat{B}, \hat{C}, D)\) we have

\[
\hat{R}_o \hat{R}_c =
\begin{pmatrix}
\hat{C} \hat{B} & \cdots & \hat{C} \hat{A}^{n-1}B \\
\vdots & \ddots & \vdots \\
\hat{C} \hat{A}^{n-1}B & \cdots & \hat{C} \hat{A}^{2(n-1)}B
\end{pmatrix}
\]

and in view of Eq. (3.36), \(\hat{R}_o \hat{R}_c = R_o R_c\). Since the system \((\hat{A}, \hat{B}, \hat{C}, D)\) is of dimension \(v\), the rank of \(\hat{R}_o \hat{R}_c\) is at most \(v\), whereas the rank of \(R_o R_c\) is \(n > v\). This is a contraction since \(\hat{R}_o \hat{R}_c = R_o R_c\).
Example 3.8

Consider the model of the aircraft given in CAD Ex. 3.1. Assume that the only control input is the aileron angle $\delta_\alpha$ and that the only measurable state is the roll angle $\gamma(t)$, that is,

$$
\begin{pmatrix}
20 \\
0 \\
0 \\
0
\end{pmatrix},
\begin{pmatrix}
0 & 0 & 0 & 1
\end{pmatrix}
$$

Decompose the system to obtain a controllable and observable subsystem. Obtain the transfer function of the overall system and verify that it is equal to the transfer function of the controllable and observable subsystem.

**Solution** It can easily be verified that the system is uncontrollable but observable. Using the procedure discussed in Sec. 3.5, we have

$$
R = \begin{pmatrix}
20 & -200 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 20 & 0 & 0
\end{pmatrix}
$$

where the first two columns of $R$ are the only linearly independent columns of the controllability matrix $R_c = (b, Ab, A^2b, A^3b)$, and the last two columns are chosen to make the matrix $R$ nonsingular. Applying the transformation $z = Px$, $P = R^{-1}$, we obtain

$$
\begin{pmatrix}
\dot{z}_1 \\
\dot{z}_2
\end{pmatrix} = \begin{pmatrix}
0 & 0 & 0 & -0.5 \\
1 & -10 & 0 & 0 \\
0 & 0 & -0.7 & 9 \\
0 & 0 & -1 & -0.7
\end{pmatrix}
\begin{pmatrix}
z_1 \\
z_2
\end{pmatrix} + \begin{pmatrix}
1 \\
0 \\
0 \\
0
\end{pmatrix} u
$$

$$
y = \begin{pmatrix}
0 & 20 & 0 & 0
\end{pmatrix}
$$

The controllable and observable subsystem is

$$
\begin{pmatrix}
0 & 0 \\
1 & -10
\end{pmatrix} z_1 + \begin{pmatrix}
1 \\
0
\end{pmatrix} u
$$

$$
y = \begin{pmatrix}
0 & 20
\end{pmatrix} z_1
$$

The transfer-function of this subsystem is

$$
G_1(s) = \begin{pmatrix}
0 & 20
\end{pmatrix} \begin{pmatrix}
s & 0 \\
-1 & s + 10
\end{pmatrix}^{-1} \begin{pmatrix}
1 \\
0
\end{pmatrix}
$$

$$
= \frac{20}{s(s + 10)}
$$

Now consider the transfer function of the overall system, which can be obtained using the Fadeev algorithm,

$$
G(s) = C(sI - A)^{-1}b = \frac{20(s^2 + 1.4s + 9.49)}{s^4 + 11.4s^3 + 23.49s^2 + 94.9s} = \frac{20}{s(s + 10)}
$$
Thus the transfer-function of the overall system and of the controllable and observable subsystem are identical. This is due to the cancellations of uncontrollable modes $s = 0.7 \pm j3.0$.

3.9 STABILITY OF LINEAR SYSTEMS

In this section, we discuss the stability of a linear system that is subjected to initial conditions and to forcing inputs.

3.9.1 Zero-Input Stability

The stability of a system when subjected to initial conditions only, that is, $u(t) \equiv 0$, is called zero-input stability.

Consider the linear system

$$x(t) = A(t) x(t), \quad x(t_0) = x_0$$  \hspace{1cm} (3.37)

A point $x_e$ in the state space is called an equilibrium point of the system if $x(t) = x_e$ for all $t \geq t_0$. Since $x_e$ is constant, we have from Eq. (3.37),

$$A(t) x_e = 0$$  \hspace{1cm} (3.38)

One obvious equilibrium point satisfying Eq. (3.38) is $x_e = 0$. However, if $A(t)$ has a zero eigenvalue, the system will have an infinite number of eigenvectors satisfying $A(t)x_e = 0$, and all these eigenvectors are the equilibrium points. In the analysis to follow, we will only consider the equilibrium point at the origin, that is, $x_e = 0$.

Consider the following definitions of stability.

**Definition 3.1.** The equilibrium state $x_e = 0$ of the system Eq. (3.37) is stable if for any given value $\epsilon > 0$, there exists a scalar $\delta(t_0, \epsilon)$ such that $\|x(t_0)\| < \delta(t_0, \epsilon)$ implies $\|x(t)\| < \epsilon$ for all $t \geq t_0$, where $\|x(t)\|$ is the norm of $x(t)$, that is, a measure of the distance of point $x(t)$ from the origin of state space.

If the equilibrium point $x_e = 0$ of a system satisfies this definition, then its state remains within $\epsilon$, in norm, of the origin provided the initial state $x(t_0)$ is less than $\delta$ in norm. Note that $\delta \leq \epsilon$ and that in general $\delta$ is a function of $\epsilon$ and the initial time $t_0$. This stability is referred to as system stability in the sense of Lyapunov (isL).

**Definition 3.2.** The system Eq. (3.37) is asymptotically stable if, (a) it is stable isL and (b) for any $t_0$, there exists a number $\delta(t_0) > 0$ such that whenever $\|x(t_0)\| < \delta(t_0)$, we have $\lim_{t \to \infty} \|x(t)\| = 0$.

Note that Definition 3.2 refers to the system behavior as $t \to \infty$, and therefore, it is called asymptotic stability. If $\delta$ is not a function of the initial time $t_0$, the system is said to be uniformly asymptotically stable.
Definition 3.3 If \( \delta(t_0) \) can be made arbitrarily large, that is, if for all \( x(t_0) \), \( \lim_{t \to \infty} \|x(t)\| = 0 \) the system is said to be globally asymptotically stable (or asymptotically stable in the large).

Theorem 3.8. Consider the unforced system Eq. (3.37):

1. The system is stable isL if and only if there exists a constant \( c \), which may depend on \( t_0 \), such that
   \[
   \| \phi(t, t_0) \| \leq c \quad \text{for all } t \geq t_0
   \]
   where \( \phi(t, t_0) \) is the transition matrix.

2. The system is asymptotically stable if and only if Eq. (3.39) holds and
   \[
   \lim_{t \to \infty} \| \phi(t, t_0) \| = 0 \quad \text{for all } t_0
   \]

3. The system is uniformly asymptotically stable in the large if and only if positive constants \( c_1 \) and \( c_2 \) exist such that
   \[
   \| \phi(t, t_0) \| \leq c_1 e^{-c_2(t-t_0)} \quad \text{for all } t \geq t_0 \text{ and all } t_0
   \]

Proof

1. The solution to Eq. (3.37) is
   \[
   x(t) = \phi(t, t_0) x_0
   \]
   Hence
   \[
   \|x(t)\| = \|\phi(t, t_0) x_0\| \leq \|\phi(t_0)\| \|x_0\| \]
   Since \( \|\phi(t, t_0)\| \leq c \), the condition of Definition 3.1 can be satisfied for any \( t > 0 \) by setting \( \delta(t_0, \epsilon) = \frac{\epsilon}{c} \). In this case, we have from Definition 3.1, Eqs. (3.39) and (3.42), \( \|x(t)\| \leq \epsilon \) and thus condition Eq. (3.39) is sufficient for stability isL. We now show that the condition is also necessary. Suppose that \( x_e = 0 \) is stable isL but \( \phi(t, t_0) \) is not bounded for \( t \geq t_0 \). This implies that an element of \( \phi(t, t_0) \) say \( \phi_{ij}(t, t_0) \) exists which is not bounded. Now choose the initial condition as \( x(t_0) = x_0 = (0, \ldots, 0, 1, 0, \ldots, 0)^T \) where the \( i \)th element of \( x_0 \) is 1 and the remaining elements are zero. In this case, the \( i \)th element of \( x(t) = \phi(t, t_0) x_0 \) will be \( \phi_{ij}(t, t_0) \) which is unbounded and therefore the system is not stable, which contradicts the assumption.

2. The proof of part 2 of the Theorem is similar to that of part 1 and will not be repeated.

3. To prove the necessity of part 3 of the Theorem, we substitute Eq. (3.41) in Eq. (3.42),
\[ \|x(t)\| \leq c_1 e^{-c_2(t-t_0)} \|x_0\| \]

Now \(\|x_0\| \leq \delta\) and thus
\[ \|x(t)\| \leq c_1 \delta e^{-c_2(t-t_0)} \quad \text{for all } t \geq t_0 \text{ and all } t_0 \]

Hence \(\|x(t)\| \to 0\) as \(t \to \infty\).

To prove the sufficiency, we note that if the system is uniformly asymptotically stable in the large, then for any \(\delta > 0\) and any \(t > 0\), there exists a time \(\tau\), independent of \(t_0\), such that \(\|x_0\| < \delta\) implies \(\|x(t)\| < \epsilon\) for all \(t \geq t_0 + \tau\). Let \(\|x_0\| = 1\) and \(t = e^{-1}\), then we have
\[ \|x(t_0 + \tau)\| = \|\phi(t_0 + \tau, t_0)x_0\| \leq \|\phi(t_0 + \tau, t_0)\| \leq e^{-1} \]

Now for any positive integer \(l\), we have
\[ \|\phi(t_0 + l\tau, t_0)\| < e^{-l} \]

Choose \(l\), \(c_2\), and \(c_1\) such that \(t_0 + l\tau \leq t \leq t_0 + (l + 1)\tau\), \(c_2 = \frac{1}{\tau}\) and \(c_1 = \|\phi(t, t_0 + l\tau)\|\), respectively. Note that \(c_1\) is bounded due to asymptotic stability of the system. Thus
\[ \|\phi(t, t_0)\| \leq \|\phi(t, t_0 + l\tau)\| \cdot \|\phi(t_0 + l\tau, t_0)\| \]
\[ = c_1 e^{-1} \|\phi(t_0 + l\tau, t_0)\| \]
\[ \leq c_1 e^{-1} e^{-l} = c_1 e^{-(l+1)} = c_1 e^{-t_0 + (l+1)\tau - t_0} = c_1 e^{-c_2(t-t_0)} \]

or
\[ \|\phi(t, t_0)\| \leq c_1 e^{-c_2(t-t_0)} \]

which is Eq. (3.41).

The following theorem is stated for stability of linear discrete-time systems. The proof of this theorem is similar to the proof presented for Theorem 3.8.

**Theorem 3.9** Consider the linear discrete-time system \(x(k + 1) = A(k)x(k)\), \(x(k_0) = x_0\) with the transition matrix \(\phi(k, k_0)\):

1. The system is stable isL if and only if there exists a constant \(c\), which may depend on \(k_0\), such that
   \[ \|\phi(k, k_0)\| \leq c \quad \text{for all } k \geq k_0 \]

2. The system is asymptotically stable if and only if in addition to part 1, the following condition also holds,
   \[ \lim_{k \to \infty} \|\phi(k, k_0)\| = 0 \quad \text{for all } k_0 \]
3. The system is uniformly asymptotically stable in the large if and only if constants 
\(c_1 > 0\) and \(|c_2| < 1\) exist such that 
\[
\| \phi (k, k_0) \| \leq c_1 c_2^{(k - k_0)} \quad \text{for all } k \geq k_0 \text{ and all } k_0
\]

For linear time-invariant systems, we can write \(\phi(t, t_0) = e^{At}\). Now each element
of \(e^{At}\) can be expressed as \(\sum_{i=1}^{n} m_i e^{\lambda_i t}\) where \(m_i\) is the multiplicity of the eigenvalue \(\lambda_i\) (see Sec. 2.6). Note that complex eigenvalues occur in conjugate pairs. In particular,
a pair of distinct eigenvalues with zero real parts, that is, \(\lambda_{1,2} = \pm j\omega\) results in 
terms \((\sin \omega t)\) in the expression for \(e^{At}\), which are bounded. However, repeated 
eigenvalues with zero real parts result in terms such as \(t \sin \omega t\) and \((t \cos \omega t)\) which
are unbounded. Thus, we can state the following result.

**Lemma 3.7.** Consider the linear time-invariant system \(\dot{x}(t) = Ax(t)\), \(x(t_0) = x_0\) and let the eigenvalues of \(A\) be \(\lambda_i, i = 1, 2, \ldots, n\).

1. The system is stable ISL if and only if \(\Re \lambda_i \leq 0, i = 1, 2, \ldots, n\) and the
eigenvalues with zero real parts are distinct, where \(\Re \lambda_i\) denotes the real part
of \(\lambda_i\).
2. The system is asymptotically stable if \(\Re \lambda_i < 0, i = 1, 2, \ldots, n\).

Note that asymptotic stability of a linear time-invariant system implies uniform
asymptotic stability in the large. It is important to note that a linear time-varying
system can be unstable even if \(\Re \lambda_i < 0\) for all \(t\), as the following example demonstrates.

**Example 3.9**

Consider the system
\[
\dot{x}(t) = \begin{pmatrix}
-1 + 1.5 \cos^2 t & 1 - 1.5 \sin t \cos t \\
-1 - 1.5 \sin t \cos t & -1 + 1.5 \sin^2 t
\end{pmatrix} x(t)
\]

Determine its stability.

**Solution** The transition matrix is
\[
\phi(t, 0) = \begin{pmatrix}
e^{0.5t} & e^{-t} \sin t \\
e^{-0.5t} \sin t & e^{-t} \cos t
\end{pmatrix}
\]

Clearly, this system is unstable due to the unbounded term \(e^{0.5t}\) in the transition matrix.
The eigenvalues of \(A(t)\) are obtained from
\[
\det [\lambda I - A(t)] = \lambda^2 + 0.5 \lambda + 0.5
\]
and \(\Re \lambda_i = -0.25, i = 1, 2\). It is seen that the system is unstable even though all
eigenvalues have negative real parts.
As with the continuous time-invariant systems, stability of linear time-invariant
discrete system \( x(k + 1) = Ax(k) \) can be determined from the location of the
eigenvalues of \( A \). Suppose that the eigenvalues of \( A \) are distinct and that a
nonsingular transformation \( z(k) = P x(k) \) has been applied to obtain the
diagonalized system matrix \( \hat{A} = P A P^{-1} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n) \). The transition matrix of the original system is

\[
\phi(k) = P^{-1} \hat{\phi}(k) P = P^{-1} \hat{A}^k P = P^{-1} \text{diag}(\lambda_1^k, \lambda_2^k, \ldots, \lambda_n^k) P
\]

It is evident that \( \phi(k) \) is bounded provided that \( |\lambda_i| \leq 1, i = 1, 2, \ldots, n \) and that
\( \lim_{k \to \infty} \phi(k) = 0 \) provided that \( |\lambda_i| < 1, i = 1, 2, \ldots, n \).

**Lemma 3.8.** Consider the system \( x(k + 1) = Ax(k), x(k_0) = x_0 \).

1. The system is stable isL if \( |\lambda_i| \leq 1, i = 1, 2, \ldots, n \) and the eigenvalues with
   \( |\lambda_i| = 1 \) are simple.
2. The system is asymptotically stable if and only if \( |\lambda_i| < 1, i = 1, 2, \ldots, n \). Note that \( |\lambda_i| < 1 \) is the locus of the points inside a circle of unit radius in
   the complex plane.

Part 1 of Lemma 3.8 can also be extended to systems with repeated eigenvalues (See
Prob. 3.20).

**CAD Example 3.4**

Determine stability of the following two systems.

\[
a. \quad \dot{x}(t) = \begin{pmatrix} 1 & 0.2 & 0 & -1 & 2 \\ 1.1 & 2 & -3 & 0 & 0 \\ 0 & -0.7 & 2 & 0.2 & 0.6 \\ 0.1 & 0 & 0 & 0.7 & 1.8 \\ 3.5 & 12 & 1 & 0 & 0 \end{pmatrix} x(t)
\]

\[
b. \quad x(k + 1) = \begin{pmatrix} 0 & 1 & 0 & 0.2 \\ 0.6 & 0 & 1 & 0 \\ -0.1 & 1 & 0 & 0 \\ 0 & 2 & 0.2 & 0.8 \end{pmatrix} x(k)
\]

Below is a **CONTROL.lab** solution of this problem.

\[
\text{\textcircled{1}} \text{A1}=[1.2 0 -1 2; 1.1 2 -3 0 0; 0 -7 2 .2 .6; .1 0 0 .7 1.8; 3.5 12 1 0 0];
\]

\[
\text{\textcircled{2}} \text{EIG(A1)}
\]

\[
\text{ANS} =
\begin{align*}
3.7443 + 1.3541i \\
3.7443 - 1.3541i
\end{align*}
\]

\[
\text{1 For details on CONTROL.lab/VAX, see Appendix B.}
\]
1.2492 - 0.0000i
-1.5189 + 0.8417i
-1.5189 - 0.8417i

// SYSTEM 1 is unstable
\( a2 = \begin{bmatrix} 0 & 1 & 0 & .2; & .6 & 0 & 1 & 0; & -.1 & 1 & 0 & 0; & 0 & 2 & .2 & .8 \end{bmatrix} \)
\( \text{eig}(a2) \)

\[ \text{ANS} = \]

-1.2485
1.3830
0.0614
0.6041

// SYSTEM 2 is unstable

### 3.9.2 Zero-State Stability

The zero-state stability is concerned about the stability of the system

\[
\begin{align*}
\dot{x}(t) &= A(t)x(t) + B(t) u(t) \\
y(t) &= C(t)x(t) + D(t) u(t)
\end{align*}
\]  
(3.43)

with zero initial condition, that is, \( x(t_0) = 0 \). It is assumed that \( A(t), B(t), C(t), \) and \( D(t) \) are continuous and bounded for \( t \geq t_0 \).

**Definition 3.4** The system Eq. (3.43) with \( x(t_0) = 0 \) is said to be bounded input bounded output (BIBO) stable if for all \( t_0 \) and for all bounded inputs \( u(t), t \geq t_0 \), the output \( y(t), t \geq t_0 \) is bounded.

**Theorem 3.10.** The system Eq. (3.43) is BIBO stable if and only if there exists a finite constant \( c \) such that

\[ \int_{t_0}^{t} \|G(t, \tau)\|_1 \, d\tau \leq c \quad \text{for any } t_0 \text{ and all } t \geq t_0 \]  
(3.44)

where \( G(t, \tau) = C(\tau)\phi(t, \tau)B(\tau) \) and the norm \( \|G(t, \tau)\|_1 = \max_j \sum_{i=1}^{n} |g_{ij}(t, \tau)| \) is used, where \( g_{ij}(t, \tau) \) is a typical element of \( G(t, \tau) \).

**Proof.** Sufficiency: Suppose that Eq. (3.44) holds and that \( u(t) \) is bounded, that is, \( \|u(t)\| \leq c_1 \) where \( c_1 \) is a finite constant. The output of system Eq. (3.43) is

\[ y(t) = \int_{t_0}^{t} C(\tau)\phi(t, \tau)B(\tau)u(\tau) \, d\tau + D(t) u(t) \]
\[\|y(t)\| \leq \int_{t_0}^{t'} \|C(\tau)\phi(t, \tau)B(\tau)u(\tau)\| d\tau + \|D(t)u(t)\|\]
\[\leq \int_{t_0}^{t'} \|C(\tau)\phi(t, \tau)B(\tau)\| \|u(\tau)\| d\tau + \|D(t)\| \|u(t)\|\]

Hence
\[\|y(t)\| \leq c_1 \int_{t_0}^{t} \|G(t, \tau)\| d\tau + c_1 \|D(t)\| \quad \text{for all } t_0 \text{ and } t \geq t_0\]

Since \(D(t)\) is assumed to be bounded, \(\|D(t)\| \leq c_2\), where \(c_2\) is a finite number. Thus, we have \(y(t) \leq c_1 c + c_1 c_2\) and the output is bounded.

Necessity: This is proved by contradiction. Suppose the system is BIBO but Eq. (3.44) is not satisfied. This implies that for any constant \(c\), there exists a time \(t_1\) such that
\[\int_{t_0}^{t} \|G(t_1, \tau)\| d\tau = \int_{t_0}^{t} \max_j \sum \sum_{i=1}^{n} \|g_{ij}(t_1, \tau)\| d\tau > c\]

which in turn implies that there exists at least one element of \(G(t_1, \tau)\) say \(g_{ij}(t_1, \tau)\), such that \(\int_{t_0}^{t} |g_{ij}(t_1, \tau)| d\tau > c\). Since for BIBO stability, any bounded input must result in bounded output, we choose the \(j\)th element of the vector \(u(t)\) as \(u_j(t) = \text{sgn} \ [g_{ij}(t_1, \tau)]\). The \(i\)th output is now
\[y_i(t_i) = \int_{t_0}^{t} g_{ij}(t, \tau) \text{sgn} \ [g_{ij}(t_1, \tau)] d\tau = \int_{t_0}^{t} |g_{ij}(t_1, \tau)| dt > c\]

Furthermore, since \(c\) can be chosen to be arbitrarily large, the output is not bounded and the system is not BIBO stable which is a contradiction.

For linear time-invariant systems, it is convenient to express the conditions for BIBO stability in terms of the transfer-function matrix.

**Lemma 3.9.** Consider an \(m\)-input \(r\)-output multivariable system described by \(Y(s) = G(s) U(s)\) where \(G(s)\) is an \(r \times m\) proper rational function matrix. The system is BIBO stable if and only if all poles of every element of \(G(s)\) have negative real parts.

**Proof.** The \(i\)th output is:
\[Y_i(s) = \sum_{j=1}^{m} g_{ij}(s) U_j(s) \quad i = 1, 2, \ldots, r \quad (3.45)\]

Consider a typical element of \(G(s)\) say \(g_{ij}(s)\) which can be expanded by partial fraction into sum of finite number of terms of the form \(a/(s - \lambda_k)^k + b\) where \(\lambda_k\) is a pole
with multiplicity \( k \), and \( a \) and \( b \) are constant numbers. Now \( g_y(t) \) is the sum of terms \( t^{k-1} e^{\lambda_k t} \) and \( \delta(t) \) where \( \delta(t) \) is the delta function. It can be verified that
\[
\int_{0}^{t} t^{k-1} e^{\lambda_k \tau} d\tau \text{ is finite if and only if } \text{Re} \lambda_k < 0.
\]
Furthermore, each output is the sum of \( m \) terms given in Eq. (3.45) and thus for BIBO stability, all elements of \( G(s) \) must have poles with negative real parts.

It is important to note that BIBO stability of the linear time-invariant system \((A, B, C, D)\) does not always guarantee asymptotic stability. The latter requires the eigenvalues of the matrix \( A \), that is, roots of \( \det(sI - A) = 0 \) to have negative real parts. On the other hand, BIBO stability requires the poles of \( G(s) = [C \det(sI - A)]/\det(sI - A) = W(s)/p(s) \) to have negative real parts. In forming \( G(s) \), there may be pole-zero cancellations between \( p(s) \) and all elements of \( W(s) \). If any cancelled pole has a positive real part but all the remaining poles have negative real parts, the system is BIBO stable but not asymptotically stable.

**Example 3.10**

Determine stability of the system
\[
\dot{x}(t) = \begin{pmatrix} -1 & 0 \\ 1 & 1 \end{pmatrix} x(t) + \begin{pmatrix} -2 \\ 1 \end{pmatrix} u(t)
\]
\[
y(t) = \begin{pmatrix} 1 & 0 \end{pmatrix} x(t)
\]

**Solution** We have
\[
p(s) = |sI - A| = (s - 1)(s + 1)
\]

The eigenvalues of \( A \) are \(-1\) and \(+1\) and thus the system is not asymptotically stable, that is, if \( u(t) = 0 \) and the system is subjected to initial conditions, \( x(t) \) grows unbounded. However, \( G(s) = C(sI - A)^{-1}B = -2/(s + 1) \) and thus the system is BIBO stable.

The condition for BIBO stability of discrete-time systems is similar to the continuous case, except that here we replace the integral in Eq. (3.44) by summation. Thus the discrete-time system
\[
x(k + 1) = A(k)x(k) + B(k)u(k)
\]
\[
y(k) = C(k)x(k) + D(k)u(k)
\]
is BIBO stable if and only if
\[
\sum_{j = k_0}^{k-1} \|G(k, j)\|_1 \leq c \quad \text{for all } k_0 \text{ and all } k \geq k_0
\]

where \( c \) is a finite constant and \( G(k, k_0) = C(k) \phi(k, k_0) B(k) \). The requirement for BIBO stability of linear time-invariant discrete systems, is that all poles of all elements of \( G(z) = C(zI - A)^{-1}B + D \) be inside the unit circle (the reader can show this as an exercise).
3.9.3 Total Stability

A system can be BIBO stable and yet blow up internally, as seen in Ex. 3.10. A more practical requirement for a linear system is total stability which is defined as follows.

**Definition 3.5.** A linear system is said to be totally stable if for any initial state and for any bounded input, both the state and the output are bounded.

We note that total stability is a stronger requirement than asymptotic stability or BIBO stability.

**Theorem 3.11.** The system Eq. (3.43) is totally stable if and only if

1. \( \| \phi(t, t_0) \| < c_1 \)

and

2. \( \int_{t_0}^{t} \| \phi(t, \tau) B(\tau) \| d\tau \leq c_2 \) for any \( t_0 \) and all \( t \geq t_0 \)

**Proof.** The response of Eq. (3.43) is

\[
x(t) = \phi(t, t_0) x(t_0) + \int_{t_0}^{t} \phi(t, \tau) B(\tau) u(\tau) d\tau
\]

Using Theorem 3.8 (part 1) and Theorem 3.10, we conclude that \( x(t) \) is bounded if and only if parts 1 and 2 of Theorem 3.11 are satisfied. Since \( C(t) \) and \( D(t) \) are bounded, \( y(t) = C(t) x(t) + D(t) u(t) \) is also bounded.

In the case of linear time-invariant systems, total stability requires that the eigenvalues of \( A \) and the poles of \( G(s) = [C \ adj(sI - A) B + D \ det(sI - A)]/\ det(sI - A) = W(s)/p(s) \) have negative real parts. Furthermore, if the system is controllable and observable, there are no common pole zero cancellations between all elements \( W(s) \) and \( p(s) \), as discussed in Sec. 3.8. As a result, for controllable and observable linear time-invariant systems, poles of the system \( G(s) \) and eigenvalues of \( A \) are identical. Hence, for asymptotic stability, BIBO stability, and total stability of a controllable and observable linear time-invariant system, it is necessary and sufficient that the eigenvalues of \( A \) have negative real parts.

3.10 LYAPUNOV STABILITY ANALYSIS

The stability of linear time-invariant systems is determined by the location of eigenvalues of \( A \) and/or the poles of the transfer-function, as discussed before. Methods for determining whether the roots of a polynomial are located in the stable region, without finding the roots, are provided by Routh-Hurwitz criterion for continuous
systems and by Jury-Blanchard criterion for discrete systems. In addition, Nyquist criterion provides a method of determining BIBO stability of a closed-loop feedback system from the transfer-function of the open-loop system. These methods can be found in standard textbooks on control systems.

The stability of a linear time-varying system is determined from its transition matrix. However, except for very simple examples, the determination of the transition matrix of a time-varying system is extremely complex, if not impossible. The Lyapunov method provides sufficient conditions for zero-input stability of linear and nonlinear time-varying systems, without requiring the evaluation of the transition matrix or the solution of state equations.

The Lyapunov method makes use of Lyapunov functions $V(x, t)$ which are defined as follows.

**Definition 3.6.** A scalar function $V(x, t)$ is called a Lyapunov function if for all $t \geq t_0$ and all vectors $x$ in the neighborhood of the origin, it satisfies the following conditions:

1. $V(0, t) = 0$
2. $V(x, t), \frac{\partial V(x, t)}{\partial t}$ and $\frac{\partial V(x, t)}{\partial x_i}$, $i = 1, 2, \ldots, n$ all exist and are continuous
3. $V(x, t) > \alpha (||x||) > 0$ for all $x \neq 0$ and $t \geq t_0$, where $\alpha (\cdot)$ is a nondecreasing function of $||x||$ and $\alpha (0) = 0$.

Condition 3 implies that $V(x, t)$ is bounded below by a nonzero function. The following theorem provides a sufficient condition for asymptotic stability of a general nonlinear time-varying system described by $\dot{x} = f(x, t)$.

**Theorem 3.12.** The equilibrium state $x_e = 0$ of the system $\dot{x}(t) = f(x, t)$, $f(0, t) = 0$ is asymptotically stable if a Lyapunov function $V(x, t)$ can be found such that $\dot{V}(x, t) < 0$ for all $x \neq 0$ and $t \geq t_0$.

The proof of this theorem is rather involved and can be found in Vidyasagar (1978). For uniform asymptotic stability, condition 3 of Definition 3.6 must be replaced by $\beta (||x||) \geq V(x, t) \geq \alpha (||x||)$ where $\beta (\cdot) > 0$ is a nondecreasing function with $\beta (0) = 0$. This implies that $V(x, t)$ must be bounded both from the above and below by nonzero functions. Furthermore, for uniform asymptotic stability in the large, the statements "neighborhood of origin" and "nondecreasing" in Definition 3.6 must be replaced by "anywhere" and "increasing," respectively.

It must be noted that many Lyapunov functions may exist for a particular system. However, the problem is to find one such function. If a Lyapunov function that satisfies the condition of Theorem 3.12 can be found, asymptotic stability is established. However, if our search for such a Lyapunov function is not successful, we cannot conclude instability. In order to establish instability, the following theorem may be used (see Vidyasagar 1978).
Theorem 3.13. The equilibrium state $x_e = 0$ of the system $\dot{x} = f(x, t), f(0, t) = 0$ is unstable if a Lyapunov function $V(x, t)$ can be found such that $\dot{V}(x, t) > 0$ and $\dot{V}(0, t) = 0$ for all $x \neq 0$ in the neighborhood of 0.

Example 3.11

Determine the stability of the system

$$\dot{x}(t) = \begin{pmatrix} 0 & 1 \\ -\frac{1}{t+3} & -15 \end{pmatrix} x(t)$$

Choose the Lyapunov function $V(x, t) = \frac{1}{2} [x_1^2 + (t + 3) x_2^2]$

Solution

$$\dot{V}(x, t) = \frac{\partial V(x, t)}{\partial x} \frac{dx}{dt} = \frac{\partial V}{\partial t} + \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} \dot{x}_i$$

$$= \frac{1}{2} x_2^2 + x_1 \dot{x}_1 + (t + 3) x_2 \dot{x}_2$$

$$= -\frac{1}{2} (30t + 89) x_2^2 \quad t \geq 0$$

Since $\dot{V}(x, t) < 0$ for all $x \neq 0$ and $t \geq 0$, the origin is asymptotically stable. Note that asymptotic stability is not uniform since for $t < -\frac{89}{30}$, we have $\dot{V}(x, t) > 0$. However, asymptotic stability is in the large, since the system is linear.

Lyapunov’s stability theorems can also be used to investigate stability of discrete-time systems described by

$$x(k + 1) = f[x(k), k], \quad f(0, k) = 0 \quad (3.46)$$

Theorems 3.12 and 3.13 can be applied to system Eq. (3.46) by noting that $V[x(t), t]$ and $\dot{V}[x(t), t]$ for discrete-time systems are $V[x(k), k]$ and $\Delta V[x(k), k] = V[x(k + 1), k + 1] - V[x(k), k]$, respectively.

For linear systems, the Lyapunov method provides both necessary and sufficient conditions for stability. However, a brief discussion on quadratic forms is needed for stating Lyapunov stability theorem of linear systems. (See also Appendix A.)

A class of scalar functions that play an important role in Lyapunov stability analysis is the quadratic form. A particularly useful quadratic form for the Lyapunov function is $V(x, t) = x^T(t) P(t) x(t)$, where $P(t)$ is a real and symmetric $n \times n$ matrix. In order that $V(x, t)$ be a Lyapunov function, $V(x, t)$ must be positive definite. Sylvester criterion provides conditions on $P(t)$ for the positive definiteness of $V(x, t) = x^T(t) P x(t)$. According to this criterion, $V(x, t) = x^T(t) P(t) x(t)$ is positive definite if and only if all principle minors of $P(t)$ are positive. Such a matrix $P(t)$ is called a positive definite matrix.
Theorem 3.14 The linear system \( \dot{x}(t) = A(t) x(t) \), where \( A(t) \) is continuous and bounded for \( t \geq t_0 \), is uniformly asymptotically stable, if and only if given a positive definite real matrix \( Q(t) \), there exists a symmetric positive definite real matrix \( P(t) \) which satisfies the equation

\[
\dot{P}(t) + A^T(t) P(t) + P(t) A(t) = -Q(t) \quad t \geq t_0
\]  

(3.47)

Proof. We only prove the sufficiency. The complete proof may be found in Vidyasagar (1978). If \( P(t) \) is positive definite, then \( V(x,t) = x^T(t) P(t) x^T(t) \) is a Lyapunov function. We have

\[
\dot{V}(x,t) = x^T(t) P(t) x(t) + x^T(t) P(t) x(t) + x^T(t) \dot{P}(t) x(t)
\]

\[
= x^T(t)(A^T(t) P(t) + P(t) A(t) + \dot{P}(t)) x(t)
\]

\[
= -x^T(t) Q(t) x(t)
\]

(3.48)

Since \( Q(t) \) is assumed to be positive definite, \( \dot{V}(x,t) \) is a negative definite quadratic form and the system is asymptotically stable.

For linear time-invariant systems, the matrices \( A, P, \) and \( Q \) are constant, and Eq. (3.47) reduces to

\[
A^T P + PA + Q = 0
\]

(3.49)

which is an algebraic equation. Thus, the linear time-invariant system \( \dot{x}(t) = A x(t) \) is asymptotically stable, if and only if given a positive definite constant matrix \( Q \), there exists a positive definite matrix \( P \) which is the solution of Eq. (3.49).

Example 3.12

Consider the linear system

\[
\begin{pmatrix}
    \dot{x}_1 \\
    \dot{x}_2
\end{pmatrix} =
\begin{pmatrix}
    0 & 1 \\
    -1 & -1
\end{pmatrix}
\begin{pmatrix}
    x_1 \\
    x_2
\end{pmatrix}
\]

Determine if the system is asymptotically stable.

Solution This system is asymptotically stable since the eigenvalues of \( A \) are \( -\frac{1}{2} \pm j \frac{\sqrt{3}}{2} \). Now, we use the Lyapunov method to verify this result. Let us choose \( Q = I \) and determine \( P \) from Eq. (3.49)

\[
\begin{pmatrix}
    0 & -1 \\
    1 & -1
\end{pmatrix}
\begin{pmatrix}
    P_1 & P_2 \\
    P_2 & P_3
\end{pmatrix} +
\begin{pmatrix}
    P_1 & P_2 \\
    P_2 & P_3
\end{pmatrix}
\begin{pmatrix}
    0 & 1 \\
    -1 & -1
\end{pmatrix} =
\begin{pmatrix}
    -1 & 0 \\
    0 & -1
\end{pmatrix}
\]

to obtain

\[
\begin{pmatrix}
    P_1 & P_2 \\
    P_2 & P_3
\end{pmatrix} =
\begin{pmatrix}
    3 & 1 \\
    2 & 2 \\
    2 & 1 \\
    1 & 1
\end{pmatrix}
\]

For asymptotic stability, the acquired \( P \) must be positive definite. We check the principal minors
Thus \( P \) is positive definite and the system is asymptotically stable.

**CAD Example 3.5**

Determine asymptotic stability of the system

\[
\dot{x} = \begin{pmatrix}
1 & 2 & 0.1 \\
7 & -0.3 & 2 \\
1 & 0 & -1.5
\end{pmatrix} x(t)
\]

using the Lyapunov method. A MATLAB solution for this problem follows:

\[
\begin{align*}
&\text{a} = [1 2 .1 ; 7 -.3 2 ; 1 0 -1.5]; \\
&q = \text{eye}(3); \\
&p = \text{lyapunov}(a,q)
\end{align*}
\]

**WARNING.**

UNSTABLE EIGENVALUE - SOLUTION MAY NOT BE POSITIVE-DEFINITE.

\[
P =
\begin{pmatrix}
-0.3772 & -0.0595 & -0.0375 \\
-0.0595 & 1.3705 & 0.1638 \\
-0.0375 & 0.1638 & 0.3084
\end{pmatrix}
\]

\[
\text{eig (a)}
\]

\[
\text{ANS} =
\begin{pmatrix}
4.2484 & -0.0000i \\
-3.1358 & +0.0000i \\
-1.9126 & -0.0000i
\end{pmatrix}
\]

**PROBLEMS**

3.1 Determine the linear dependence of the functions

\[
f_1(t) = (1 \ e^t), \quad f_2(t) = (t \ e^t)
\]

over \([0, 1]\).

3.2 Show that the state \( x(t_0) \) can be transferred to \( x(t_1) \), if and only if \( x(t_0) \) can be transferred to the origin of state space, that is, \( x(t_1) = 0 \).
3.3 Determine the controllability of
\[ \dot{x} = \begin{pmatrix} 0 & -1 \\ 0 & t \end{pmatrix} x + \begin{pmatrix} 0 \\ 2 \end{pmatrix} u \]

3.4 The controllability of time-invariant systems is invariant under nonsingular state transformation. Can this result be extended to time-varying systems?

3.5 Consider the system \( \dot{x}(t) = A \ x(t) + B \ u(t) \), where \( A \) has distinct eigenvalues. Show that the system is controllable, if and only if all rows of \( M^{-1}B \) are nonzero, where \( M \) is the modal matrix, i.e., a matrix whose columns are eigenvectors of \( A \).

3.6 Show that the transfer-function matrix of the systems Eqs. (3.30) and (3.31) is \( G(s) = \hat{C}_1 (sI - \hat{A}_{11})^{-1} \hat{B}_1 + D \).

3.7 The observable continuous-time system \( \dot{x}(t) = \begin{pmatrix} 0 & -a \\ a & 0 \end{pmatrix} x(t), \ a > 0, \ y(t) = (1 \ 0)x(t), \) is discretized with a sampling period of \( T \), to obtain the discrete-time system
\[ x(k + 1) = \begin{pmatrix} \cos aT & -\sin aT \\ \sin aT & \cos aT \end{pmatrix} x(k), \ y(k) = (1 \ 0)x(k) \]

Discuss the observability of the acquired discrete-time system.

3.8 Determine the observability of the system
\[ \dot{x}(t) = \begin{pmatrix} -1 & 2 \\ 0 & -2 \end{pmatrix} x(t) + \begin{pmatrix} \sin t \\ 1 \end{pmatrix} u(t) \]
\[ y(t) = \begin{pmatrix} e^t \\ e^t \end{pmatrix} ^T x(t) \]

3.9 Develop a computer program to check the controllability and observability of the single-input single-output system
\[ \dot{x}(t) = A \ x(t) + b \ u(t) \]
\[ y = c \ x(t) \]
by checking the rank condition. Let the order of the system be \( n \leq 10 \).

3.10 Develop a computer program to check the controllability and observability of the system of Prob. 3.9, by determining poles-zero cancellations.

3.11 Consider the discrete-time system \( x(k + 1) = Ax(k) + Bu(k) \). Prove the controllability condition directly without using the concept of linear dependence of time functions. In other words, show that for every initial state \( x_0 \), there exists a control sequence \( u(k), k = 0, 1, \ldots, n - 1 \) which drives the system to \( x(k_1) = x_1 \), where \( x_1 \) is any desired state, if and only if rank \( R_c = n; \ R_c = (B \ AB \cdots A^{n-1}B) \).

Furthermore, show that the required control sequence is
\[ \begin{pmatrix} u(n - 1) \\ u(n - 2) \\ \vdots \\ u(0) \end{pmatrix} = (R_c R_c^T)^{-1} R_c (x_1 - A^n x_0) \]
3.12 Let $A$ and $B$ be $n \times n$ and $n \times m$ matrices. Show the rank $(B, AB, \ldots, A^{n-1}B) = \text{rank } (B AB \cdots A^{n-\mu}B)$, where $\mu = \text{rank } B$.

3.13 Consider the system

$$\dot{x}(t) = \begin{pmatrix} 0 & 1 & 0 \cdots & 0 \\ 0 & 0 & 1 \cdots & 0 \\ & \vdots & \ddots & \vdots \\ 0 & 0 & 0 \cdots & 1 \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} u(t)$$

Show that the system is controllable for all $p_0, p_1, \ldots, p_{n-1}$.

3.14 Determine the observability of the system

$$\dot{x} = \begin{pmatrix} 1 & 0.2 & -0.1 & 1.5 \\ 0 & -1.2 & -0.5 & -0.7 \\ 0.3 & 0.4 & 0.7 & 2.1 \\ 2.1 & 0 & 3.1 & 0.1 \end{pmatrix} x(t) + \begin{pmatrix} 0.1 \\ 0.3 \\ -0.7 \\ 1 \end{pmatrix} u(t)$$

$$y(t) = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0.1 & 0 & 0.6 & 0.8 \end{pmatrix} x(t)$$

3.15 Determine controllability and observability of the system

$$\dot{x}(t) = \begin{pmatrix} 0 & 0 & -6 \\ 1 & 0 & -11 \\ 0 & 1 & -6 \end{pmatrix} x(t) + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} u(t)$$

$$y(t) = (1 \ 0 \ 0) x(t)$$

3.16 Develop a computer program to decompose an uncontrollable system into two subsystems; one that is controllable and one that is uncontrollable. Let the order of the original system be $n \leq 10$ and use the procedure of Lemma 3.5. Develop a computer program to decompose an unobservable system into an observable and unobservable subsystem. Use the procedure given in Lemma 3.6 and let $n \leq 10$.

3.17 Show that the transfer-function matrix of the system Eq. (3.33) is given by Eq. (3.35).

3.18 Consider the linear scalar system

$$\dot{x}(t) = a(t) x(t) \quad t \geq 0$$

where $a(t)$ is a continuous function

a. Verify that the solution is $x(t) = x(t_0) \exp \left[ \int_{t_0}^t a(\tau) d\tau \right]$

b. Show that the system is stable isL, if and only if

$$\max \int_{t_0}^t a(\tau) d\tau \triangleq m(t_0) < \infty$$

show that in this case $\delta (t_0, \epsilon) = \frac{\epsilon}{m(t_0)}$.

3.19 Show that the system $x(k + 1) = A x(k), x(t_0) = x_0$ is stable isL, if and only if all the eigenvalues of $A$ are inside the unit circle in the complex plane and the eigenvalues on
the unit circle have Jordan blocks of orders no higher than unity. Demonstrate the idea by applying it to $A_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ which has two Jordan blocks of order 1, and $A_2 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ which has one Jordan block of order 2.

3.20 Suppose that the system $\dot{x}(t) = A \ x(t)$ is asymptotically stable. Show that the solution to $A^T P + P A = -Q$ is given by

$$P = \int_0^\infty e^{At} Q e^{A^T t} \ dt$$

where $Q$ is a given real symmetric matrix.

3.21 Consider the system $\dot{x}(t) = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} x(t)$

Show that the system is not stable although the eigenvalues of $A(t)$ have negative real parts. Hint: the transition matrix is

$$\phi(t, 0) = \begin{pmatrix} e^{-t} & (e^t - e^{-t})/2 \\ 0 & e^{2t} \end{pmatrix}$$

3.22 Show that the system $\dot{x}(t) = \frac{-x}{2t} + u(t), \quad t \geq 1, \quad y(t) = x(t)$ is asymptotically stable. Is this system also BIBO stable?

3.23 Write a computer program to solve the Lyapunov Eq. (3.49), where $Q$ is a given symmetric matrix and $n \leq 10$.

3.24 Prove the following result: The linear system $\dot{x}(t) = A(t) x(t)$ has eigenvalues whose real parts are less than $\sigma$, if and only if given a positive definite matrix $Q$, there exists a positive definite matrix $P$ which satisfies the equation $PA + A^T P - 2\sigma P = -Q$. (Hint: Let $A = \dot{A} - \sigma I$ and apply Theorem 3.14).
4
State
Feedback and
Observer
Design

4.1 INTRODUCTION

This chapter is primarily concerned with the design of compensators using state variable feedback to shift the poles or eigenvalues of a linear time-invariant system to the desired locations for stabilization or response shaping.

In Sec. 4.2, we consider the concept of state variable feedback and derive the closed-loop system matrix description. Assuming that all the state variables are accessible for feedback, the state feedback is utilized to arbitrarily assign the closed-loop poles for controllable open-loop systems. Many algorithms are available for realizing this closed-loop pole assignment. If we exclude the iterative methods, a majority of direct algorithms assume that the open-loop system has been transformed to a special form using a similarity transformation. Techniques of this type, which are provided in Sec. 4.3, are based on the companion form and the Hessenberg form. Generally speaking, direct algorithms result either in a full rank feedback gain matrix or rank unity feedback gain matrix known as *dyadic form*. Both will be considered in the context of the present chapter.

In many practical situations, it is not possible to access all state variables for feedback implementation. When the controlled system has inaccessible state variables and a state feedback system is to be designed, we can derive an estimate of the state vector to utilize as a feedback signal rather than the true state vector. In Sec. 4.4, we introduce the deterministic state estimator known as the *Luenberger Observer* under the assumption of observability. Both full-order and reduced-order observers are considered and appropriate design techniques for small and large-size systems
are provided. Finally, in Sec. 4.5, we apply the state feedback to the output of an observer and establish the separation property which shows that the state feedback and observer can be designed independently without destroying the stability property of the closed-loop system. A discussion of robustness in observer-based controller design concludes the section.

4.2 STATE FEEDBACK CONCEPT

Let us consider the general linear time-invariant system

\[ \dot{x}(t) = A \, x(t) + B \, u(t) \]  \hspace{1cm} (4.1)
\[ y(t) = C \, x(t) + D \, u(t) \]  \hspace{1cm} (4.2)

where \( x(t) \) is the \( n \times 1 \) state vector, \( u(t) \) is an \( m \times 1 \) vector of inputs, \( y \) is an \( r \times 1 \) vector of outputs, and \( A, B, C, \) and \( D \) are constant matrices of dimensions \( n \times n, n \times m, r \times n, \) and \( r \times m, \) respectively. If a feedback control law of the form

\[ u(t) = v(t) + K \, x(t) \]  \hspace{1cm} (4.3)

is introduced, where \( v(t) \) is a reference input vector and \( K \) is an \( m \times n \) real constant feedback gain matrix. The dynamical equation of the state feedback system is shown in Fig. 4.1 and is of the form

\[ \dot{x}(t) = (A + BK) \, x(t) + B \, v(t) \]  \hspace{1cm} (4.4)
\[ y(t) = (C + DK) \, x(t) + Dv(t) \]  \hspace{1cm} (4.5)

Thus, the stability of the feedback system depends on the eigenvalues of \( A + BK, \) controllability depends on the pair \( (A + BK, B) \), and observability depends on the pair \( (A + BK, C + DK) \). Let us consider an example to gain some insight about the qualitative effects of the feedback.

![Figure 4.1 State variable feedback system.](image-url)
Example 4.1
Consider the controllable, observable, and stable system

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -3 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 2 \end{bmatrix} x$$

Investigate the effects of feedback on controllability and observability.

Solution Introduce a feedback signal of the form $u = v + [4 \quad -2]x$. Then it is easy to verify that the new system

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 1 & -6 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v$$

$$y = \begin{bmatrix} 1 & 2 \end{bmatrix} x$$

is controllable and observable but not stable. Next, introduce a feedback signal of the form $u = v + [2.5 \quad 2.5]x$. Then the system becomes

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -0.5 & -1.5 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v$$

$$y = \begin{bmatrix} 1 & 2 \end{bmatrix} x$$

which is controllable and stable, but not observable. Finally, let the feedback law to be in the general form $u = v + [K_1 \quad K_2]x$. Then the feedback system is obtained as

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -3 + K_1 & -4 + K_2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v$$

$$y = \begin{bmatrix} 1 & 2 \end{bmatrix} x$$

and the rank of controllability matrix is 2 independent of $K_1$ or $K_2$.

This example shows that observability and stability could be altered by the use of state feedback but not controllability. Applications of state feedback in linear control systems are stabilization, response shaping, asymptotic tracking, decoupling, and achieving cyclicity, to name a few. In this text, we concentrate mainly on stabilization.

The state feedback system given by Eqs. (4.4) and (4.5) is controllable for any feedback gain matrix $K$, if and only if the system described by Eqs. (4.1) and (4.2) is controllable. A consequence of this fact is that the controllability of a linear time-invariant system is invariant under any linear state feedback.

4.3 POLE PLACEMENT

In Ex. 4.1, we generated an unstable system by applying state feedback to an asymptotically stable system. In many design problems, we are interested in the opposite effect, that is, to stabilize an unstable system through state feedback. In
fact, one of the most important properties of state feedback is that it can be used to control the eigenvalues of the closed-loop control system. This control of the eigenvalues is called eigenvalue assignment, pole assignment, or simply pole placement.

The problem of pole placement has been of fundamental importance in the control system discipline for many years and numerous algorithms have been proposed in the literature to solve this problem. These algorithms result in feedback gain which are broadly speaking either dyadic (i.e., have rank equal to 1) or have full rank. The basic problem of pole placement by state feedback is to find a feedback control law

\[ u(t) = G v(t) + K x(t) \]  \hspace{1cm} (4.6)

for the system

\[ \dot{x}(t) = A x(t) + B u(t) \]  \hspace{1cm} (4.7)
\[ y(t) = C x(t) + D u(t) \]  \hspace{1cm} (4.8)

such that the resulting closed-loop system

\[ \dot{x}(t) = (A + BK) x(t) + BG v(t) \]  \hspace{1cm} (4.9)
\[ y(t) = (C + DK) x(t) + DG v(t) \]  \hspace{1cm} (4.10)

has a desired set of eigenvalues \( \{\lambda_i, \quad i = 1, \ldots, n\} \). The dimensions of variables have been defined previously. The matrices \( A, B, C, D, G, \) and \( K \) are respectively \( n \times n, \quad n \times m, \quad r \times n, \quad r \times m, \quad m \times m, \) and \( m \times n, \) where \( G \) is an adjustment gain to be used, if necessary, as a nonsingular transformation on the input. It is well known that for a single-input controllable system there is a unique \( K \) that achieves this goal. A controllable multi-input system, however, has no unique solution, and therefore, the designer has certain flexibility in choosing the feedback gain matrix \( K \). This degree of freedom can be used to achieve some other goals such as response shaping or robust solution to the problem.

Two general approaches are available in solving pole placement problems. The first approach is iterative and will not be discussed in this text with one exception (see Chap. 5). The second approach is direct and several reliable algorithms are provided. These algorithms either require that the open-loop system be transformed to a special form considered as design coordinate or no transformation is required. However, before discussing these techniques, let us consider coefficient matching technique as follows:

1. Compute the characteristic polynomial \( p_k(s) \) of \( A + BK \) in terms of the unknown \( nm \) components of the \( K \) matrix.

2. Compute the desired characteristic polynomial using the given set of desired eigenvalues \( \{\lambda_i, \quad i = 1, \ldots, n\} \) as \( p_k(s) = (s - \lambda_1)(s - \lambda_2) \cdots (s - \lambda_n) \).

\[ ^1 \text{We are assuming that the system is minimal (see Sec. 3.8), that is, controllable and observable so that the set of poles is the same as the set of eigenvalues.} \]
3. Set $p_k(s) = p_d(s)$ and equate the coefficients of the powers of $s$ to generate a set of $n$ equations to solve for the $nm$ components of $K$.

This set of $n$ equations is usually not linear in the components of $K$. Let us illustrate this with one example.

**Example 4.2**
Consider the system

\[
\dot{x} = \begin{bmatrix} 0 & -1 \\ -2 & -1 \end{bmatrix} x + \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix} u
\]

\[
y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} x
\]

Investigate the pole placement problem for eigenvalues at $-1$ and $-2$.

**Solution** System is controllable and observable, that is, irreducible (minimal), but unstable since the eigenvalues of $A$ are $1$ and $-2$. We need to position the closed-loop poles at $\lambda_1 = -1$, $\lambda_2 = -2$. Then by using the control law

\[
u = v + Kx = v + \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} x
\]

we have

\[
p_k(s) = \det [sI - (A + BK)] = \det \begin{bmatrix} s + k_{21} & k_{22} + 1 \\ 2 + 2k_{11} + k_{21} & s + (1 + 2k_{12} + k_{22}) \end{bmatrix}
\]

and

\[
p_d(s) = (s + 1)(s + 2) = s^2 + 3s + 2
\]

If we expand the determinant to obtain $p_k(s)$ and match its coefficients with the ones in $p_d(s)$, we will obtain a set of nonlinear equations. Since the system has two inputs, there is no unique solution for $K$. One possible solution is $k_{11} = -2$, $k_{12} = 1$, $k_{21} = 0$, and $k_{22} = 0$, which is dyadic, i.e., $k$ is rank deficient. A possible full rank solution can also be given by $k_{11} = -1.5$, $k_{12} = 0.5$, $k_{21} = 0$, and $k_{22} = 1$.

**Theorem 4.1.** If the open-loop dynamical system Eq. (4.7) and Eq. (4.8) are controllable, then by a feedback control law of the form Eq. (4.6), the eigenvalues of the closed-loop dynamical system Eq. (4.9), Eq. (4.10) can be arbitrarily assigned.

**Proof.** The proof of this theorem can be found in Wonham (1979) from a geometric point of view and in many standard texts such as Chen (1984) and Kailath (1980) from an algebraic point of view. The methods of subsec. 4.3.2 and 4.3.3 constitute constructive proofs of this theorem for single-input and multi-input systems, respectively. The basic idea of the proof follows from the simple argument that the
controllability of the pair \((A, B)\) assures the existence of a similarity transformation which transforms the system to the controllable canonical form. Then, the required feedback gain can be easily obtained to guarantee the desired eigenvalues for the closed-loop system.

Realizing this closed-loop pole assignment, a class of direct algorithm requires an initial transformation of the system to the appropriate design coordinates. Two such coordinate transformations are discussed.

4.3.1 Coordinate Transformation

The key to the successful design of most control problems lies in appropriate system transformation to a particular form. Canonical forms have been briefly discussed in Chap. 2 for the special case of (single-input single-output) SISO systems. In conjunction with the pole placement design of (multi-input multi-output) MIMO systems, we concentrate on two design coordinates. The multi-input phase variable or companion form also known as controllable canonical form, which gives explicit information about structural indices, is considered first. Then the block Hessenberg form, which is more preferable from a computational point of view, will be discussed.

**Controllable canonical form.** If the system Eq. (4.7), Eq. (4.8) is controllable, then the controllability matrix

\[
R_c = [B \ AB \ \cdots A^{n-1}B]
\]  

(4.11)

has \(n\) linearly independent columns. By choosing these independent columns, or their linear combination, as basis various controllable canonical forms, can be obtained Luenberger (1967). Here, let the nonsingular \(n \times n\) transformation matrix \(P_c\) be constructed from the columns of \(R_c\) as follows:

**Algorithm 4.1.**

1. Search the columns of \(R_c\) from left to right and select those vectors that are linearly independent from previously selected ones.
2. Arrange the \(n\) linearly independent vectors to form a new matrix \(T_c\) as

\[
T_c = [b_1 \ Ab_1 \ \cdots A^{\mu_1-1}b_1; \ \cdots; \ b_m Ab_m \ \cdots A^{\mu_m-1}b_m]
\]  

(4.12)

where the integers \(\mu_1, \ldots, \mu_m\) are the controllability indices with \(\mu_c = \max_i \mu_i; i = 1, \ldots, m\) defined as controllability index and \(\sum_{i=1}^n \mu_i = n\).
3. Let \(T_c^{-1}\) be represented in terms of its rows as

\[
T_c^{-1} = \begin{bmatrix}
t_1^T \\
t_2^T \\
\vdots \\
t_n^T
\end{bmatrix}
\]  

(4.13)

\(^2\) See also Sec. 2.5.
\[(61.4) \quad u \cdot \ldots \cdot 1 = (f \neq 1) \begin{bmatrix} x & \ldots & x & x \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \ldots & 1 & 0 \\ 0 & \ldots & 0 & 1 \end{bmatrix} = \eta_1^\forall \]

\[(8.4) \quad u \cdot \ldots \cdot 1 = ! \begin{bmatrix} x & \ldots & x & x & \lambda \\ 1 & \ldots & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 1 & 0 & 0 \\ 0 & \ldots & 0 & 1 & 0 \end{bmatrix} = \eta_1^\forall \]

and

\[\mathcal{A} = \mathcal{C}_1^\forall \mathcal{C} \mathcal{C}_1^\forall \mathcal{C} = \mathcal{C}_1^2 \mathcal{C} = \mathcal{C}_1^2 \]

\[\begin{bmatrix} b^w \\ \vdots \\ b \end{bmatrix} = b^2 \mathcal{A} = b^2 \begin{bmatrix} \mathcal{C}_1^w \mathcal{C}_1^w \ldots \mathcal{C}_1^w \\ \mathcal{A}_1^w \mathcal{A}_1^w \ldots \mathcal{A}_1^w \\ \vdots \\ \mathcal{A}_1^w \mathcal{A}_1^w \ldots \mathcal{A}_1^w \\ \mathcal{C}_1^w \mathcal{C}_1^w \ldots \mathcal{C}_1^w \end{bmatrix} = \mathcal{C}_1^2 \mathcal{A} \mathcal{A} = \mathcal{C}_1^2 \mathcal{A} \]

where

\[(7.1.17) \quad (i)n^2 \mathcal{A} + (i)2x^2 \mathcal{C} = (i)\mathcal{C} \]

\[(7.1.16) \quad (i)n^2 \mathcal{B} + (i)2x^2 \mathcal{A} = (i)\mathcal{B} \]

Then, from the system to the block controllable canonical

\[\begin{bmatrix} 1-n\mathcal{A}_1^w \\ \mathcal{A}_1^w \\ \vdots \\ \mathcal{A}_1^w \\ 1-n\mathcal{A}_1^w \\ \vdots \\ \mathcal{A}_1^w \\ \mathcal{A}_1^w \\ \mathcal{A}_1^w \end{bmatrix} = \mathcal{A} \]

\[\text{4. From the required transformation matrix, } \mathcal{P}^\forall \text{ as}
\]

\[(9.1.14) \quad u \cdot \ldots \cdot 1 = ! \begin{bmatrix} 1 = f \\ \vdots \end{bmatrix} = \eta_1^\forall \]

and let \( \gamma^t \) be the \( \mathcal{A} \)th row of \( \mathcal{J} \), where
The matrix $A_C$ is in block companion form and the matrix $C_C$ has no special form under this transformation. We also note that the nonzero entries, $x$, in the matrix $B_C$ can be replaced by zeros using a nonsingular transformation applied to the system inputs. It has been recognized that the Luenberger construction procedure as outlined previously leads to the controllable canonical form in a generic sense. In general, however, it is difficult to see $B_i$ in the form of Eq. (4.20); rather, the zero block above the $\mu_i$ row may have nonzero entries. Therefore, the transformed system cannot be said to be in the canonical form for this and other reasons. We shall provide a method due to Popov (1972), for obtaining a controllable canonical form in a general sense. For simplicity of notation, let us describe the procedure via an example.

Suppose the controllability matrix has been constructed and $\mu_1 = 3, \mu_2 = 2$ so that

$$A^3 b_1 = \alpha_1 b_2 + \alpha_2 A b_2 + \alpha_3 b_1 + \alpha_4 A b_1 + \alpha_5 A^2 b_1$$

$$A^2 b_2 = \beta_1 b_2 + \beta_2 A b_2 + \beta_3 b_1 + \beta_4 A b_1 + \beta_5 A^2 b_1$$

The Popov construction first puts all terms with nonzero power of $A$ on the left and then factor out $A$ as

$$A\left(A^2 b_1 - \alpha_5 Ab_1 - \alpha_4 b_1 - \alpha_3 b_2\right) = \alpha_1 b_2 + \alpha_3 b_1$$

Now repeat this operation with $e_{11}$ as

$$A\left(\alpha_5 b_1\right) = e_{11} + \alpha_4 b_1 + \alpha_2 b_2$$

and then with $e_{12}$ as

$$Ab_1 = e_{12} + \alpha_5 b_1$$

Doing the same for the next chain, we obtain

$$A\left(A b_2 - \beta_2 b_2 - \beta_5 Ab_1 - \beta_4 b_1\right) = \beta_1 b_2 + \beta_3 b_1$$

and

$$A\left(b_2 - \beta_3 b_1\right) = e_{21} + \beta_4 b_1$$
It is easy to show that
\[
\begin{pmatrix}
e_{11} & e_{12} & e_{13} & e_{21} & e_{22}
\end{pmatrix}
\]
form the required inverse transformation matrix \( P_c^{-1} \).

**Example 4.3**

Consider the controllable system
\[
A = \begin{bmatrix}
-1 & -1 & 0 \\
0 & -2 & 1 \\
0 & 0 & -3
\end{bmatrix}, \quad B = \begin{bmatrix}
1 & 0 \\
0 & 1 \\
1 & 0
\end{bmatrix}
\]
\[
C = \begin{bmatrix}
1 & 0 & 0 \\
0 & -1 & 1
\end{bmatrix}
\]

It is desired to transform it to controllable canonical form.

**Solution**  System has the controllability matrix \( R_C \) given by
\[
R_C = \begin{bmatrix}
b_1 & b_2 & A b_1 & A b_2 & A^2 b_1 & A^2 b_2 \\
1 & 0 & -1 & -1 & 0 & 3 \\
0 & 1 & 1 & -2 & -5 & 4 \\
1 & 0 & -3 & 0 & 9 & 0
\end{bmatrix}
\]

using the steps (1) to (4) we obtain
\[
T_C = [b_1 \quad Ab_1 \quad b_2] = \begin{bmatrix}
1 & -1 & 0 \\
0 & 1 & 1 \\
1 & -3 & 0
\end{bmatrix}
\]

with controllability indices \( \mu_1 = 2, \mu_2 = 1 \) and
\[
T_C^{-1} = \frac{1}{2} \begin{bmatrix}
3 & 0 & -1 \\
1 & 0 & -1 \\
-1 & 2 & 1
\end{bmatrix} = \begin{bmatrix}
t_1^r \\
t_2^r \\
t_3^r
\end{bmatrix}
\]

with \( k_1 = \mu_1 = 2, k_2 = \mu_1 + \mu_2 = 3 \). Therefore,
\[
P_C = \begin{bmatrix}
t_2^r \\
A t_2^r A \\
t_3^r
\end{bmatrix} = \frac{1}{2} \begin{bmatrix}
1 & 0 & -1 \\
-1 & -1 & 3 \\
-1 & 2 & 1
\end{bmatrix}
\]

and
\[
A_C = P_C A P_c^{-1} = \begin{bmatrix}
0 & 1 & 0 \\
17 & -18 & -3 \\
-\frac{2}{4} & 0 & -6
\end{bmatrix}
\]

\[
B_C = P_C B = \begin{bmatrix}
0 & 0 \\
1 & -\frac{1}{2} \\
0 & 1
\end{bmatrix}
\]
\[ C_c = C P_c^{-1} = \begin{bmatrix} \frac{7}{2} & 1 \\ 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \\ 1 & -\frac{1}{2} \end{bmatrix} \]

For this example, the Popov construction procedure is not necessary.

**Special case.** For single input systems, the nonsingular transformation matrix \( P_c \) can also be constructed by

\[
Q_c = P_c^{-1} = R_c \tilde{R}_c^{-1}
\]

where \( R_c \) is the controllability matrix and \( \tilde{R}_c^{-1} \) is given by

\[
\tilde{R}_c^{-1} = \begin{bmatrix}
    p_1 & p_2 & \cdots & p_{n-1} & 1 \\
p_2 & p_3 & \cdots & 1 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
p_{n-1} & 1 & \cdots & 0 & 0 \\
1 & 0 & \cdots & 0 & 0
\end{bmatrix}
\]

with \( p_i \)'s being the coefficients of the characteristic polynomial

\[
p(s) = s^n + p_{n-1}s^{n-1} + \cdots + p_1s + p_0.
\]

**Computational Aspects.** The reduction of a given system to a controllable canonical form requires steps in which Gaussian elimination without partial pivoting should be performed. This is known to be numerically unstable.\(^3\) On the other hand, the procedure of computing the required equivalence transformation uses the controllability matrix Eq. (4.11). If the dimension of the system \( n \) is large, the computation of \( A^kb \) for large \( k \) may transform the problem into a less well-conditioned\(^4\) problem.

In order to see this, assume that all eigenvalues \( \lambda_i \) of \( A \) are distinct with \( |\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \) and \( b \) is an \( n \times 1 \) vector for simplicity. Then, \( b \) can be written in terms of eigenvectors of \( A \) as

\[
b = \alpha_1v_1 + \alpha_2v_2 + \cdots + \alpha_nv_n
\]

and consequently,

\[
A^kb = \alpha_1\lambda_1^kv_1 + \alpha_2\lambda_2^kv_2 + \cdots + \alpha_n\lambda_n^kv_n
\]

where we have used the relation \( Av_i = \lambda_iv_i \). Now, if \( \lambda_1 \) is much larger than all other eigenvalues, then we have

\[
A^kb \to \alpha_1\lambda_1^kv_1 \text{ for large } k.
\]

\(^3\) A procedure for solving a problem or an algorithm is said to be numerically unstable, if numerical errors inside the procedure will not be amplified; otherwise, the algorithm is numerically stable.

\(^4\) If a problem is very sensitive to the variations of data, then the problem is ill conditioned; otherwise, it is well conditioned.
In other words, $A^k b$ tends to approach the same vector, $v_1$, as $k$ increases. This means that the transformation matrix $P_c$ becomes nearly singular and it is difficult to be constructed. One can also use a different argument based on the condition number of the matrix $A$ and arrive at the same conclusion. The condition number of a matrix is a robustness measure for the solution to be produced due to the changes in parameters. This may be defined as $\text{cond } A \triangleq \|A\|_2 \|A^{-1}\|_2 = \sigma_{\text{max}}/\sigma_{\text{min}}$, where the notation $\sigma$ is used for singular value. If $|\lambda_i| >> |\lambda_n|$, then $\sigma_{\text{max}} >> \sigma_{\text{min}}$ and $A$ has a large condition number. Since the multiplication of an ill-conditioned matrix is used in the process of generating the controllability matrix, large computational errors will be produced. It should be pointed out that the diagonal (Jordan) canonical form which displays the eigenvalues of the system has also been used in many design techniques. However, their treatments are beyond the scope of the present chapter, and therefore, shall concentrate on a numerically preferable orthogonal transformation.

**Controllable Hessenberg form.** The transformation to controllable canonical form for large size problems, in particular, is not recommended, since it requires in general a numerically unstable procedure. Instead, we shall introduce an efficient and numerically stable method to transform a state equation into the so-called upper block Hessenberg form. This important form is also known as condensed or staircase form and has several nice properties, as will be seen in our development.

Consider again, the controllable system Eq. (4.7), Eq. (4.8) and without loss of generality, assume that the matrix $B$ is of full column rank $m$. Then the system can be transformed by a sequence of orthogonal transformations, combined as an $n \times n$ nonsingular matrix $P_h$, that is, $x_h = P_h x$, into the controllable block Hessenberg form

$$\dot{x}_h(t) = A_h x_h(t) + B_h u(t) \quad (4.22)$$

$$y(t) = C_h x_h(t) + D_h u(t) \quad (4.23)$$

where

$$A_h = P_h A P_h^T = \begin{bmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1,k-1} A_{1k} & n_1 \\ A_{21} & A_{22} & A_{23} & \cdots & A_{2,k-1} A_{2k} & n_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & A_{k,k-1} A_{k,k} & n_k \end{bmatrix}$$

$$B_h = P_h B = \begin{bmatrix} B_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$C_h = C P_h^{-1} = [C_1 \ C_2 \ \cdots \ C_l]$$

$$D_h = D$$
The matrix $A_h$ is in upper block Hessenberg form, the block matrices $B_1$ and $\{A_{q,q-1}, q = 2, 3, \ldots, k\}$ are of full row rank, that is, $\rho(B_1) = n_1 = m$, $\rho(A_{q,q-1}) = n_q$ and may be assumed to be in upper row echelon form. Clearly, we have $m = n_1 \geq n_2 \geq \cdots \geq n_k$. The controllability indices $\{\mu_i, i = 1, 2, \ldots, m\}$ of the pair $(A, B)$ or correspondingly $(A_h, B_h)$ can be derived directly from $\{n_j, j = 1, 2, \ldots, k\}$. This is discussed in Patel (1981) and Van Dooren (1981). A recursive algorithm is outlined which shows the steps to be carried out for the construction of $P_h$.

Algorithm 4.2

1. Set $P_h = I_n$, $A_0 = A$, $B_0 = B$, $\tilde{n} = 0$, $j = 1$.
2. Find an orthogonal transformation $P_j$ such that
   \[
P_jB_{j-1} = \begin{bmatrix} X_j \\ \vdots \\ 0 \end{bmatrix} n_j
   \]
   where $\rho(X_j) = n_j$.
3. Compute
   \[
P_jA_{j-1}P_j^T = \begin{bmatrix} Y_j \\ Z_j \\ B_j \\ A_j \end{bmatrix}
   \]
   where $Y_j$ is an $n_j \times n_j$ matrix.
4. Update the transformation
   \[
P_h = \begin{bmatrix} I_{\tilde{n}} & 0 \\ 0 & P_j \end{bmatrix} P_h
   \]
5. Set $\tilde{n} = \tilde{n} + n_j$ if $\tilde{n} = n$, go to step 7.
6. Set $j = j - 1$ and go to step 2.
7. Define $P_h$, $A_h$, $B_h$, $C_h$ and stop.

To further illustrate their procedure, one can use a sequence of QR factorization available in MATLAB to carry out the steps as follows: Initialize the parameters $P_h = I_n$, $A_0 = A$, $B_0 = B$, $\tilde{n} = 0$, $j = 1$ and use $QR(B_0)$ to obtain

\[
B_0 = Q_1R_1
\]

or equivalently

\[
P_1B_0 = \begin{bmatrix} X_1 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} B_1^{(1)} \\ \vdots \\ 0 \end{bmatrix} n_1
\]

Compute
\[ P_1 A_0 P_1^T = \begin{bmatrix} Y_1 & Z_1 \\ B_1 & A_1 \end{bmatrix} = \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} \\ A_{21}^{(1)} & A_{22}^{(1)} \end{bmatrix} \]

and update \( P_h \) as

\[ P_h = \begin{bmatrix} I_0 & 0 \\ 0 & P_1 \end{bmatrix} I_n = P_1 \]

Set \( \bar{n} = n_1, j = 2 \) and use \( QR(B_1) \) to obtain

\[ P_2 B_1 = \begin{bmatrix} X_2 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} A_{21}^{(2)} \\ \vdots \\ 0 \end{bmatrix} n_2 \]

Then compute

\[ P_2 A_1 P_2^T = \begin{bmatrix} Y_2 & Z_2 \\ B_2 & A_2 \end{bmatrix} = \begin{bmatrix} A_{22}^{(2)} & A_{23}^{(2)} \\ A_{32}^{(2)} & A_{33}^{(2)} \end{bmatrix} \]

and update \( P_h \) as

\[ P_h = \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2 \end{bmatrix} P_1 \]

Now let us see the overall effect of this updated \( P_h \) at this stage.

\[ B_h = P_h B = \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2 \end{bmatrix} P_1 B = \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2 \end{bmatrix} \begin{bmatrix} B_1^{(1)} \\ 0 \end{bmatrix} = \begin{bmatrix} B_1^{(1)} \\ 0 \end{bmatrix} \]

\[ A_h = P_h A P_h^T = \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2 \end{bmatrix} P_1 A P_1^T \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2^T \end{bmatrix} = \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2 \end{bmatrix} \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} \\ A_{21}^{(1)} & A_{22}^{(1)} \end{bmatrix} \begin{bmatrix} I_{n_1} & 0 \\ 0 & P_2 \end{bmatrix}^T = \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} P_2^T \\ P_2 A_{21}^{(1)} & P_2 A_{22}^{(1)} P_2^T \end{bmatrix} \]
\[
\begin{bmatrix}
A_{11}^{(1)} & \quad & A_{12}^{(2)} \\
\hline
A_{21}^{(2)} & \quad & A_{22}^{(2)} & \quad & A_{23}^{(2)} \\
\hline
0 & \quad & A_{32}^{(2)} & \quad & A_{33}^{(2)}
\end{bmatrix}
\]

\[C_h = C P_h^T = [C_1 \quad C_2 \quad C_3]\]

Continuing this process, one arrives at the final form of Eq. (4.22), Eq. (4.23).

**CAD Example 4.1**

Consider the Distillation column example of Klein and Moore (1982) which is a fifth order system described by

\[
\dot{x} = \begin{bmatrix}
-0.1094 & 0.0628 & 0 & 0 & 0 \\
1.306 & -2.132 & 0.9807 & 0 & 0 \\
0 & 1.595 & -3.149 & 1.547 & 0 \\
0 & 0.0355 & 2.632 & -4.257 & 1.855 \\
0 & 0.0023 & 0 & 0.1636 & -0.1625
\end{bmatrix} x
\]

\[
+ \begin{bmatrix}
0 & 0 \\
0.0638 & 0 \\
0.0838 & -0.1396 \\
0.1004 & -0.206 \\
0.0063 & -0.0128
\end{bmatrix} u
\]

We use CONTROL.lab to obtain \( P_h \).

\(< >\text{AO} = A\)

\[
\begin{bmatrix}
-0.1094 & 0.0628 & 0.0000 & 0.0000 & 0.0000 \\
1.306 & -2.132 & 0.9807 & 0.0000 & 0.0000 \\
0.0000 & 1.595 & -3.149 & 1.5470 & 0.0000 \\
0.0000 & 0.0355 & 2.632 & -4.2570 & 1.8550 \\
0.0000 & 0.0023 & 0.0000 & 0.1636 & -0.1625
\end{bmatrix}
\]

\(< >\text{BO} = B\)

\[
\begin{bmatrix}
0.0000 & 0.0000 \\
0.0638 & 0.0000 \\
0.0838 & -0.1396 \\
0.1004 & -0.2060 \\
0.0063 & -0.0128
\end{bmatrix}
\]

\(< >\text{<Q1, R1> = QR(BO)}\)
R1 =
\[
\begin{bmatrix}
-0.01456 & 0.2229 \\
0.0000 & -0.1114 \\
0.0000 & 0.0000 \\
0.0000 & 0.0000 \\
0.0000 & 0.0000
\end{bmatrix}
\]

Q1 =
\[
\begin{bmatrix}
0.0000 & 0.0000 & -0.5992 & -0.7991 & -0.0499 \\
-0.4380 & -0.8763 & -0.1605 & 0.1199 & 0.0065 \\
-0.5754 & 0.1020 & 0.6497 & -0.4853 & -0.0302 \\
-0.6893 & 0.4700 & -0.4386 & 0.3315 & -0.0415 \\
-0.0433 & 0.0284 & -0.0275 & -0.0417 & 0.9974
\end{bmatrix}
\]

< > P1 = INV (Q1)

P1 =
\[
\begin{bmatrix}
0.0000 & -0.4380 & -0.5754 & -0.6893 & -0.0433 \\
0.0000 & -0.8763 & 0.1020 & 0.4700 & 0.0284 \\
-0.5992 & -0.1605 & 0.6497 & -0.4386 & -0.0275 \\
-0.7991 & 0.1199 & -0.4853 & 0.3315 & -0.0417 \\
-0.0499 & 0.0065 & -0.0302 & -0.0415 & 0.9974
\end{bmatrix}
\]

< > PH1 = P1

< > AT = P1 \times AO \times INV (P1)

AT =
\[
\begin{bmatrix}
-1.0971 & 0.8847 & -0.7953 & 1.3941 & -1.3114 \\
0.3002 & -2.6280 & 1.1761 & 0.4873 & 1.0099 \\
-0.5904 & -0.1790 & -3.5434 & 2.8877 & -0.8086 \\
0.4871 & 0.1914 & 2.5954 & -2.2899 & 0.6147 \\
-0.1591 & 0.1866 & -0.1249 & 0.0919 & -0.2515
\end{bmatrix}
\]

< > B1 = <AT(3,1) AT(3,2); AT(4,1) AT(4,2); AT(5,1) AT(5,2)>

B1 =
\[
\begin{bmatrix}
-0.5904 & -0.1790 \\
0.4871 & 0.1914 \\
-0.1591 & 0.1866
\end{bmatrix}
\]

< > A1 = <AT(3,3) AT(3,4) AT(3,5); AT(4,3) AT(4,4) AT(4,5); AT(5,3) AT(5,4) AT(5,5)>

... AT(5,4) AT(5,5)>
\[ A_1 = \begin{bmatrix} -3.5434 & 2.8877 & -0.8086 \\ 2.5954 & -2.2899 & 0.6147 \\ -0.1249 & 0.0919 & -0.2515 \end{bmatrix} \]

\[ < > < Q_2, R_2 > = Q R (B_1) \]

\[ R_2 = \begin{bmatrix} 0.7817 & 0.2165 \\ 0.0000 & -0.2380 \\ 0.0000 & 0.0000 \end{bmatrix} \]

\[ Q_2 = \begin{bmatrix} -0.7552 & 0.0651 & -0.6522 \\ 0.6231 & -0.2376 & -0.7452 \\ -0.2035 & -0.09692 & 0.1389 \end{bmatrix} \]

\[ < > P_2 = \text{INV}(Q_2) \]

\[ P_2 = \begin{bmatrix} -0.7552 & 0.6231 & -0.2035 \\ 0.0651 & -0.2376 & -0.9692 \\ -0.6522 & -0.7452 & 0.1389 \end{bmatrix} \]

\[ < > P H_2 = < \text{EYE}(2) < 0 \ 0 \ 0 \ 0 >; < 0 \ 0 \ 0 \ 0 > \ P_2 > \ast P_1 \]

\[ P H_2 = \begin{bmatrix} 0.0000 & -0.4380 & -0.5754 & -0.6893 & -0.0433 \\ 0.0000 & -0.8763 & 0.1020 & 0.4700 & 0.0284 \\ -0.0352 & 0.1946 & -0.7869 & 0.5462 & -0.2082 \\ 0.1992 & -0.0452 & 0.1869 & -0.0671 & -0.9586 \\ 0.9793 & 0.0162 & -0.0663 & 0.0333 & 0.1875 \end{bmatrix} \]

\[ < > A_1 T = P_2 \ast A_1 \ast \text{INV}(P_2) \]

\[ A_1 T = \begin{bmatrix} -5.7337 & 0.1300 & 0.0310 \\ 0.9399 & -0.2437 & 0.0008 \\ -0.3809 & 0.0291 & -0.1074 \end{bmatrix} \]

\[ < > B_2 = < A_1 T(3,1) \ A_1 T(3,2) > \]
B2 =
\[-0.3809 \quad 0.0291\]

< > A2 = <A1T(3,3)>

A2 =
\[-0.1074\]

< > <Q3, R3> = QR(B2)

R3 =
\[-0.3809 \quad 0.0291\]

Q3 =
\[1\]

< > P3 = INV(Q3)

P3 =
\[1\]

< > PH3 = EYE(5) * PH2

< > PH = PH3

PH =
\[
\begin{pmatrix}
0.0000 & -0.4380 & -0.5754 & -0.6893 & -0.0433 \\
0.0000 & -0.8763 & 0.1020 & 0.4700 & 0.0284 \\
-0.0352 & 0.1946 & -0.7869 & 0.5462 & -0.2082 \\
0.1992 & -0.0452 & 0.1869 & -0.0671 & -0.9586 \\
0.9793 & 0.0162 & -0.0663 & 0.0333 & 0.1875
\end{pmatrix}
\]

< > AH = PH * A * PH'

AH =
\[
\begin{pmatrix}
-1.0971 & 0.8847 & 1.7361 & 0.8879 & -0.7024 \\
0.3002 & -2.6280 & -0.7901 & -1.0180 & -0.9899 \\
0.7817 & 0.2165 & -5.7337 & 0.1300 & 0.0310 \\
0.0000 & -0.2380 & 0.9399 & -0.2437 & 0.0008 \\
0.0000 & 0.0000 & -0.3809 & 0.0291 & -0.1074
\end{pmatrix}
\]

< > BH = PH * B
\[ BH = \]
\[
\begin{array}{cc}
-0.1456 & 0.2229 \\
0.0000 & -0.1114 \\
0.0000 & 0.0000 \\
0.0000 & 0.0000 \\
0.0000 & 0.0000
\end{array}
\]

### 4.3.2 Pole Placement for Single-Input Systems

For single-input systems, the control distribution matrix and direct transmission matrix of Eq. (4.7), Eq. (4.8) reduce to \( n \times 1 \) and \( r \times 1 \) vectors, namely, \( B = b \) and \( D = d \), respectively. Consequently, the control law Eq. (4.6) reduces to \( u(t) = v(t) + k x(t) \), where \( k = [k_1 \; k_2 \; \cdots \; k_n] \). The design procedure which assigns the desired set of eigenvalues \( \{\lambda_i, \quad i = 1, \ldots, n\} \) of the closed-loop system is accomplished as follows.

**Algorithm 4.3.** (Design Coordinate Required)

1. Using the transformation matrix \( P_C \) specified by Eq. (4.21), transform the system into the controllable canonical form

\[ \dot{x}_c = A_c x_c + b_c u \]  \hspace{1cm} (4.24)
\[ y = C_c x_c + d_c u \]  \hspace{1cm} (4.25)

where

\[ A_c = P_C A P_C^{-1} = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-p_0 & -p_1 & -p_2 & \cdots & -p_{n-1}
\end{bmatrix}, \quad b_c = P_C b = \begin{bmatrix}
0 \\
0 \\
\vdots \\
0 \\
1
\end{bmatrix} \]

The matrix \( C_c = C P_C^{-1} \) has no special form and \( d_c = d \).

2. Use the unknown transformed feedback gain

\[ k_c = [k_{c1} \; k_{c2} \; \cdots \; k_{cn}] \]  \hspace{1cm} (4.26)

to form the closed-loop matrix

\[ A_c + b_c k_c = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-p_0 + k_{c1} & -p_1 + k_{c2} & -p_2 + k_{c3} & \cdots & -p_{n-1} + k_{cn}
\end{bmatrix} \]  \hspace{1cm} (4.27)
3. Using the set of desired closed-loop eigenvalues, obtain the corresponding closed-loop characteristic polynomial \( p_A(s) = (s - \lambda_1)(s - \lambda_2) \cdots (s - \lambda_n) = s^n + p_{n-1,d}s^{n-1} + \cdots + p_{0d} \) and translate it to the desired companion form of the closed-loop matrix

\[
A_d = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-p_{0d} & -p_{1d} & -p_{2d} & \cdots & -p_{n-1,d}
\end{bmatrix}
\] (4.28)

4. Equating \( A_C + b_ck_C \) from step 2 and \( A_d \) from step 3 yields

\[
k_{C1} = p_0 - p_{0d}
\]
\[
k_{C2} = p_1 - p_{1d}
\]
\[\vdots\]
\[
k_{Cn} = p_{n-1} - p_{n-1,d}
\] (4.29)

5. Compute the original feedback gain \( k \) from

\[
k = k_C P_C
\] (4.30)

Example 4.4

Consider the unstable single-input system

\[
\dot{x} = \begin{bmatrix}
-1 & 1 & -1 \\
0 & 2 & 0 \\
0 & 0 & 3
\end{bmatrix} x + \begin{bmatrix}
0 \\
1 \\
1
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix} x
\]

and assign the set of eigenvalues \( \lambda_1 = -1, \lambda_2 = -2, \lambda_3 = -3 \) so that the feedback control system is stabilized.

Solution Since the system is single-input, we use the steps of the Algorithm 4.3 as follows.

1. Compute the transformation matrix \( P_C \) using Eq. (4.21) as

\[
P_C = [R_C \hat{R}_C^{-1}]^{-1} = \begin{bmatrix}
-1 & 0 & 0 \\
-3 & -2 & 1 \\
-2 & -1 & 1
\end{bmatrix}^{-1} = \begin{bmatrix}
-1 & 0 & 0 \\
1 & -1 & 1 \\
-1 & -1 & 2
\end{bmatrix}
\]

and construct the controllable canonical form specified by Eq. (4.24), Eq. (4.25) as

\[
A_C = P_C A P_C^{-1} = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-6 & -1 & 4
\end{bmatrix},
\quad
b_C = P_C b = \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}
\]
2. Use Eq. (4.26) and form Eq. (4.27) as

\[ A_C + b_c k_C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 + k_{c1} & -1 + k_{c2} & 4 + k_{c3} \end{bmatrix} \]

3. Construct the desired closed-loop characteristic polynomial \( P_d(s) = (s + 1)(s + 2)(s + 3) = s^3 + 6s^2 + 11s + 6 \) and translate it to the stable matrix \( A_d \) with the companion form Eq. (4.28) as

\[ A_d = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix} \]

4. Compute \( k_C \) from Eq. (4.29), which yields

\[ k_C = [0 \quad -10 \quad -10] \]

5. Compute \( k \) from Eq. (4.30) to get

\[ k = [0 \quad 20 \quad -30] \]

Remark. There are other algorithms that are available based on both design coordinates. For high order systems, it is recommended to use design techniques based on Hessenberg form which are computationally preferable. (Miminis and Paige (1982), Petkov et al. (1984), Patel and Misra (1984), Murdoch and Shriba (1985), Datta (1987), Shafai and Bhattacharyya (1988)). Since such an algorithm is provided for general multi-input systems in Subsec. 4.3.4, we will not discuss its special case here.

**Algorithm 4.4** (Design Coordinate Not Required)

1. Compute the characteristic polynomial of the system

\[ p(s) = \det(sI - A) = s^n + p_{n-1} s^{n-1} + \cdots + p_0 \quad (4.31) \]

2. Compute the desired characteristic polynomial of the closed-loop system

\[ p_d(s) = (s - \lambda_1)(s - \lambda_2) \cdots (s - \lambda_n) \]

\[ = s^n + p_{n-1,d} s^{n-1} + \cdots + p_{0,d} \quad (4.32) \]

3. Define

\[ \bar{p} = [p_{n-1} \quad p_{n-2} \cdots \quad p_0] \]

\[ \bar{p}_d = [p_{n-1,d} \quad p_{n-2,d} \cdots \quad p_{0,d}] \]

4. Compute the feedback gain by Bass-Gura (1965) formula

\[ k = - [\bar{p}_d - \bar{p}] T^{-1} R_C^{-1} \quad (4.35) \]

where \( T \) is an upper triangular Toeplitz matrix.
\[ T = \tilde{R}_C^{-1} N = \begin{bmatrix}
1 & p_{n-1} & \cdots & p_2 & p_1 \\
0 & 1 & \cdots & p_3 & p_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 1 & p_{n-1} \\
0 & 0 & \cdots & 0 & 1
\end{bmatrix} \quad (4.36) \]

with \( \tilde{R}_C^{-1} \) as defined by Eq. (4.21) and \( N \) is a normal matrix with all off-diagonal elements equal to 1 and \( R_C \) is the controllability matrix of the system.

The derivation of formula Eq. (4.35) should be obvious from the determinant identity
\[
p_d(s) = \det [sl - (A + bk)] = \det (sl - A) \det [I - (sl - A)^{-1}bk] = p(s) [1 - k(sl - A)^{-1}b]
\]
or equivalently from
\[
p_d(s) - p(s) = -p(s) k (sl - A)^{-1}b \quad (4.37)
\]
by applying the resolvent formula (see Chap. 2) to \((sl - A)^{-1}\) and coefficient matching.

Ackermann (1972) noted that there is no need to compute Eq. (4.31) and gave an alternative formula as
\[
k = -q'_n p_d(A) \quad (4.38)
\]
where
\[
q'_n = [0 \ 0 \ \cdots \ 0 \ 1] \tilde{R}_C^{-1} \quad (4.39)
\]

It is evident from Eq. (4.35) and (4.38) that the feedback gain requires the inverse computation of the controllability matrix which may cause numerical difficulties; particularly, for ill-conditioned \( R_C \). Also, computation of \( p_d(A) \) needs numerical care for high-order systems. Consequently, Ackermann (1985) recommended techniques to improve the numerical properties of his formula. Another formula is given by Mayne and Murdoch (1970) which provides the feedback gains in terms of the eigenvalues. Using Eq. (4.37), we write
\[
\frac{p_d(s)}{p(s)} = 1 - \sum_{i=1}^{n} \frac{k_i b_i}{s - s_i} \quad (4.40)
\]
This means that one should make a partial fraction expansion on \( p_d(s)/p(s) \) and divide the coefficient of the term \((s - s_i)^{-1}\) by \( b_i \) to get
\[
k_i = -\frac{\prod_j \left(s_i - \lambda_j\right)}{b_i \prod_j (s_i - s_j)} \quad (4.41)
\]
where \( s_i \) are eigenvalues of \( A \) and \( \lambda_i \) are the desired eigenvalues. As pointed out previously, these formulas are appropriate for small size problems. If large size
problems are faced, then Algorithm 4.6 of the next section is recommended to be used here also for the special case of single-input systems. (See Prob. 4.10.)

Example 4.5
The model of a rigid satellite in a frictionless environment is given by

\[ T(t) = J \frac{d^2 \theta}{dt^2} \]

where \( T(t) \) is the applied torque caused by firing the thrusters, \( \theta(t) \) is the attitude angle, and \( J = 10 \) is the satellite's moment of inertia.

1. Write the state equation of the system with the states chosen as angular position and velocity.
2. Design a control system by pole placement such that the closed-loop system has a time constant \( \tau = 1 \text{ s} \), and a damping ratio \( \xi = 0.707 \).
3. Suppose there is no signal that appears in the rate path. What is the nature of the system response in this failure mode?

Solution

1. Let \( x_1 = \theta, x_2 = \dot{\theta}, \) and \( u = T \), then

\[
\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 10 \end{bmatrix} u \\
y = [1 \ 0] x
\]

2. Since \( \tau = 1/\xi \omega_n = 1 \) and \( \xi = 0.707 \), we have \( \omega_n = \sqrt{2} \) and the desired characteristic polynomial becomes \( p_d(s) = s^2 + 2\xi \omega_n s + \omega_n^2 = s^2 + 2s + 2 \). Now using Ackerman's formula we have

\[
k = -[0 \ 1][B \ AB]^{-1} p_d(A) = [-0.2 \ -0.2]
\]

3. In the failure mode, the closed-loop system becomes

\[
\dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 10 \end{bmatrix} v
\]

and the step response is undamped oscillating (sinusoid of sustained amplitude). Figure 4.2 shows the step response for all three cases of unstable open loop, stable closed loop, and closed loop under failure.

CAD Example 4.2
The following third-order system has open-loop poles at 6.8912 and \(-0.4096 \pm j 0.6471\).

\[
\dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 4 & 5 & 6 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} u
\]
Figure 4.2 Step responses for Example 4.5.
(a) State $x_1$.
(b) State $x_2$. 
Place the closed-loop poles of the system at $-8$ and $-3 \pm j$, using POLP of CONTROLlab.

\[
\begin{align*}
\text{A} &= \begin{bmatrix} 0 & 1 & 0; & 0 & 0 & 1; & 4 & 5 & 6 \end{bmatrix}; \\
\text{B} &= \begin{bmatrix} 0; & 1; & 0 \end{bmatrix}; \\
\text{LAM} &= \begin{bmatrix} -8 & 0; & -3 & 1; & -3 & -1 \end{bmatrix}; \text{Desired Poles} \\
\text{EIG}(A) \\
6.8912 + 0. \\
-0.4096 + 0.6471i \\
-0.4096 - 0.6470i \\
\end{align*}
\]

Note that the open loop system is unstable.

\[
\text{POLP(A,B,LAM)}
\]

The element of ROW Vector k:

\[
-0.20000d + 02 \quad -0.46000 + 2 \quad -0.45750 + 02
\]

The new matrix (A + B*K):

\[
\begin{array}{ccc}
-0.20000d + 02 & -0.45000d + 02 & -0.45750d + 02 \\
0. & d + 00 & 0. \\
-0.40000d + 01 & 0.50000d + 01 & -0.60000d + 01 \\
\end{array}
\]

### 4.3.3 Pole Placement for Multi-Input Systems

One possible approach to solve the multi-input pole placement problem is to reduce it to the single-input pole placement problem from which any of the methods discussed in Subsec. 4.3.2 could be employed. This approach imposes a distinct structural constraint on the feedback gain matrix; that is, its unity rank structure known as dyadic. The word dyad may be defined as the product of a column vector and a row vector which justifies the fact that a unity rank feedback gain matrix should be generated
as such. This observation forms the basis of the analytical developments of the dyadic methods for pole placement.

The state feedback matrix derived by any other approach will, in general, have full rank. The majority of the algorithms belonging to this category involve a significant amount of numerical computation. On the other hand, the dyadic algorithms are simpler and computationally lighter, however, the resulting closed-loop systems have poor disturbance rejection properties.

**Dyadic Methods.** To develop dyadic methods, we need the definition of a cyclic matrix. A matrix \( A \) is called cyclic if its characteristic polynomial is equal to its minimal polynomial. Consequently, a matrix \( A \) is cyclic, if and only if the Jordan form of \( A \) has one and only one Jordan block associated with each eigenvalue. We may also characterize a cyclic matrix as one that can be transformed to a one-block companion matrix by a similarity transformation. If a matrix \( A \) is cyclic, then there exists a vector \( b \) such that the vectors \( b, Ab, \ldots, A^{n-1}b \) span the \( n \)-dimensional real vector space or, equivalently \((A, b)\) is controllable. This property can be used to establish the following theorem (Wonham (1967) and Gopinath (1971)).

**Theorem 4.2.** Consider the \( n \)-dimensional linear time-invariant system Eq. (4.7), Eq. (4.8).

1. If \( \{A, B\} \) is a controllable pair and \( A \) is cyclic, then for almost any \( m \times 1 \) real vector \( p \) the pair \( \{A, Bp\} \) is controllable.
2. If \( \{A, B\} \) is a controllable pair and \( A \) is not cyclic, then almost any state feedback gain matrix \( K \) will make \( A + BK \) cyclic.

Consequently, in the general case, the design consists of the following steps:

1. Use an arbitrary state feedback \( u = w + K_1x \) such that \( A + BK_1 \) in
   \[
   \dot{x} = (A + BK_1)x + Bw
   \]  
   (4.42)
   is cyclic.
2. Use the state feedback \( w = v + K_2x \) where
   \[
   K_2 = pq
   \]  
   (4.43)
   with almost arbitrary choice of \( m \times 1 \) vector \( p \) such that \( \{A + BK_1, Bp\} \) in
   \[
   \dot{x} = [A + BK_1 + Bpq]x + Bv
   \]  
   (4.44)
   is controllable.
3. Apply algorithm 4.5 or 4.6 to obtain \( 1 \times n \) vector \( q \) such that the closed-loop system matrix Eq. (4.44) has a desired set of eigenvalues.
   This method shows that the resulting feedback gain matrix will be the sum of the feedback gains \( K_1 \) and \( K_2 \) that is, \( K = K_1 + pq \).
Example 4.6

Consider the system

\[
\dot{x} = \begin{bmatrix}
-1 & 0 & 0 \\
0 & -1 & 1 \\
0 & 0 & -1
\end{bmatrix} x + \begin{bmatrix}
1 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix} x
\]

It is desired to investigate the use of Theorem 4.2.

Solution  Although the system is stable, it is required to shift the eigenvalues to the new locations \(-2, -3, -4\) for the purpose of response improvement. Since the matrix \(A\) is not cyclic, we use

\[
K_1 = \begin{bmatrix}
-1 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

to make it cyclic. The resulting closed-loop system according to Eq. 4.42 will be

\[
\dot{x} = \begin{bmatrix}
-2 & 0 & 0 \\
0 & -1 & 1 \\
0 & 0 & -1
\end{bmatrix} x + \begin{bmatrix}
1 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} w
\]

Now we use the dyadic design. Choosing \(p = [1 \quad 0]^T\) or \(p = [0 \quad 1]^T\) does not provide us with a controllable pair of \(\{A + BK_1, Bp\}\). Thus, we choose a different \(p\) as

\[
p = \begin{bmatrix}
1 \\
1
\end{bmatrix}
\]

which results a controllable pair. Next, we use Algorithm 4.4 to obtain the feedback gain which shifts the eigenvalues to the desired locations according to Eq. (4.44). The characteristic polynomial of the system Eq. 4.42 is

\[
p(s) = (s + 1)^2(s + 2) = s^3 + 4s^2 + 5s + 2
\]

The desired characteristic polynomial of the closed-loop system \(p_d(s)\) is

\[
p_d(s) = (s + 2)(s + 3)(s + 4) = s^3 + 9s^2 + 26s + 24
\]

We define

\[
\tilde{p} = [4 \quad 5 \quad 2]
\]

\[
\tilde{p}_d = [9 \quad 20 \quad 24]
\]

and obtain \(k\) from Eq. 4.35 as

\[
k = -\begin{bmatrix} 5 & 21 & 22 \end{bmatrix} \begin{bmatrix} 1 & 4 & 5 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & -2 & 4 \\ 0 & 1 & -2 \\ 1 & -1 & 1 \end{bmatrix}^{-1} = [0 \quad -6 \quad -5]
\]

The resulting feedback gain is therefore
\[ K = K_1 + K_2 = K_1 + \begin{bmatrix} -1 & -6 & -5 \\ 0 & -6 & -5 \end{bmatrix} \]

To conclude, we should point out that the primitive FSF (full-state feedback) available in CONTROL.lab can be used to obtain MIMO pole placement dyadic method.

**Full rank methods**

**Algorithm 4.5** (Design Coordinate Required)

1. Using the transformation matrix \( P_C \) specified by Eq. (4.15), transform the system to Eq. (4.16), Eq. (4.17). If the desired controllable canonical form is not constructed, one can use the general scheme of Popov to achieve this goal.
2. Use Eq. (4.6) or equivalently
   \[ u = Gv + K_C x_C \]  \hspace{1cm} (4.45)
   to form
   \[ \dot{x}_C = (A_C + B_C K_C) x_C + B_C G v \]  \hspace{1cm} (4.46)
   where the nonsingular matrix \( G \) is selected to remove the nonzero entries of \( B_C \) so that
   \[ B_C G = \tilde{B}_C = \text{block diag} \{[0 \hspace{0.5cm} 0 \hspace{0.5cm} \cdots \hspace{0.5cm} 1]^T, \mu_i \times 1, i = 1, \cdots, m\} \]  \hspace{1cm} (4.47)
   and
   \[ B_C K_C = \tilde{B}_C \tilde{K}_C \]  \hspace{1cm} (4.48)
   with
   \[ \tilde{K}_C = G^{-1} K_C \]
3. Using the set of desired closed-loop eigenvalues, specify the desired closed-loop matrix \( A_d \) as
   \[ A_d = \text{block-diag} \{A_{d_1}, A_{d_2}, \cdots, A_{d_m}\} \]  \hspace{1cm} (4.50)
   where each \( A_{d_i} \) is in companion form and has the same size as the blocks \([A_C + \tilde{B}_C \tilde{K}_C]_i\).
4. Equate \( A_C + \tilde{B}_C \tilde{K}_C \) and \( A_d \) and determine the unknown elements of \( \tilde{K}_C \).
5. Compute the original feedback gain \( K \) from
   \[ K = K_C P_C = G \tilde{K}_C P_C \]  \hspace{1cm} (4.51)

Let us illustrate this algorithm by using a simple example.
Example 4.7

Consider the system

\[
\dot{x} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u
\]

\[
y = \begin{bmatrix} 1 & 3 & -1 \\ 0 & 1 & 0 \end{bmatrix} x
\]

which is controllable with the controllability indices \( \mu_1 = 1, \mu_2 = 2 \). Design a MIMO pole placement controller.

Solution

1. Using the transformation matrix \( P_C \) specified by Eq. (4.15) obtain the transformed system Eq. (4.16), Eq. (4.17) as

\[
P_C = \begin{bmatrix} 1 & 3 & -1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad P_C^{-1} = \begin{bmatrix} 1 & -4 & 1 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}
\]

\[
A_C = P_C A P_C^{-1} = \begin{bmatrix} -1 & 7 & 0 \\ 0 & 0 & 1 \\ 0 & -2 & 3 \end{bmatrix}
\]

\[
B_C = P_C B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
\]

\[
C_C = C P_C^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}
\]

2. Since \( B_C = \tilde{B}_C \), the matrix \( G \) is identity and \( u = v + K_c x \). Thus

\[
A_C + B_C K_C = \begin{bmatrix} -1 + k_{c_{11}} & k_{c_{12}} + 7 & k_{c_{13}} \\ 0 & 0 & 1 \\ k_{c_{21}} & -2 + k_{c_{22}} & 3 + k_{c_{23}} \end{bmatrix}
\]

3. Let the desired eigenvalues be \(-1, -2, -3\). Then the desired closed-loop matrix is

\[
A_d = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -6 & -5 \end{bmatrix}
\]

4. Equating \( A_C + B_C K_C \) and \( A_d \), we get

\[
k_{c_{11}} = k_{c_{13}} = k_{c_{21}} = 0, \quad k_{c_{12}} = -7, \quad k_{c_{22}} = -4, \quad k_{c_{23}} = -8
\]

5. Finally, the original feedback gain is obtained as

\[
K = K_c P_C = \begin{bmatrix} 0 & -7 & 0 \\ 0 & -12 & -8 \end{bmatrix}
\]
4.3.4 Pole Placement for High-Order Systems

In spite of the conceptual simplicity of the pole placement problem, the numerical considerations become particularly important when dealing with high-order systems. A natural way to resolve the problem is the decomposition approach (Shieh et al. (1983)). However, this and the majority of earlier pole placement algorithms (Fallside (1977)) require the transformation of a given system to the controllable canonical form. This transformation requires in general two steps in which the second step must be carried out by Gaussian elimination without partial pivoting and is numerically unstable. Most recent algorithms consider block controllable Hessenberg form which uses numerically preferable orthogonal transformation (Miminis and Paige (1982), Petkov et al. (1984,1986), Patel and Misra (1984), Chu (1986), Shafai and Bhattacharyya (1988)). Also expensive iterative algorithms have been proposed for improvement of robustness by making the eigenvector matrix maximally orthonormal or minimizing the condition number of the eigenvector matrix (Kautsky et al. (1985)). Here, our objective is to provide an efficient direct algorithm, which can be used for high-order systems. This is accomplished by reducing the multi-input system to a block Hessenberg form and reformulating the state feedback pole placement problem in terms of Sylvester's equation. Then, by assuming a priori, the structures of the unknown matrices, we decompose the associated Sylvester's equation and show that the multi-input pole assignment problem can be solved via a number of lower order subproblems in a numerically efficient manner. These subproblems are guaranteed to have solutions because they consist of pole assignment problems with at least as many inputs as states and are reduced to the solution of linear equations. Furthermore, the solutions to subproblems can be done in a parallel fashion, which makes the method attractive for parallel implementation.

The Algorithm set-up. In what follows, we reformulate the pole assignment problem in terms of Sylvester's equation and provide a new method for its solution.

We know that when the given system represented in the design coordinate Eq. (4.22), Eq. (4.23) is controllable, all the eigenvalues of \( A_h + B_hK_h \) can be assigned arbitrarily by a proper choice of \( K_h \). Equivalently, one may write

\[
A_h + B_hK_h = TFT^{-1}
\]  

(4.52)

that is, \( A_h + B_hK_h \) and \( F \) are similar and have the same set of eigenvalues, where \( T \) is a nonsingular matrix and \( \det [sI - (A_h + B_hK_h)] = \det [sI - F] \). The Eq. (4.52) can also be written as

\[
A_hT - TF = B_hG
\]  

(4.53)

where

\[
G = -K_hT
\]  

(4.54)

We are now looking for a nonsingular matrix \( T \) which satisfies the Sylvester Eq. (4.53) and determines at the same time the matrix \( G \).
Assume the structure of the matrix \( T \) to be of the form

\[
T = \begin{bmatrix}
I_{n_1} & T_{12} & T_{13} & \cdots & T_{1,k-1} & T_{1k} \\
0 & I_{n_2} & T_{23} & \cdots & T_{2,k-1} & T_{2k} \\
0 & 0 & I_{n_3} & \cdots & T_{3,k-1} & T_{3k} \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & I_{n_{k-1}} & T_{k-1,k} \\
0 & 0 & 0 & \cdots & 0 & I_{n_k}
\end{bmatrix}
\]  
(4.55)

and let

\[
F = \begin{bmatrix}
F_1 & 0 & 0 & \cdots & 0 & 0 \\
A_{21} & F_2 & 0 & \cdots & 0 & 0 \\
0 & A_{32} & F_3 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & F_{k-1} & 0 \\
0 & 0 & 0 & \cdots & A_{k,k-1} & F_k
\end{bmatrix}
\]  
(4.56)

which implies that its eigenvalues are the union of the eigenvalues for each diagonal block.

Furthermore, let the matrix \( G \) be appropriately partitioned as

\[
G = [G_1 \quad G_2 \quad G_3 \quad \cdots \quad G_{k-1} \quad G_k]
\]  
(4.57)

Then by substituting Eqs. (4.55), (4.56), (4.57) and \( A_h, B_h \) from Eq. (4.22) into Eq. (4.53), and equating the correspondingly positioned blocks on either side of this equation we get the following set of chain equations.

\[
A_{11} - F_1 - T_{12}A_{21} = B_1G_1
\]

\[
A_{11}T_{12} + A_{12} - T_{12}F_2 - T_{13}A_{32} = B_1G_2
\]

\[
A_{11}T_{13} + A_{12}T_{23} + A_{13} - T_{13}F_3 - T_{14}A_{43} = B_1G_3
\]

\[
\vdots
\]

\[
A_{11}T_{1,k-1} + A_{12}T_{2,k-1} + A_{13}T_{3,k-1} + \cdots + A_{1,k-1} - T_{1,k-1}F_{k-1} - \cdots - T_{1,k}A_{k,k-1} = B_1G_{k-1}
\]

\[
A_{11}T_{1k} + A_{12}T_{2k} + A_{13}T_{3k} + \cdots + A_{1k} - T_{1k}F_k = B_1G_k
\]

\[
A_{21}T_{12} + A_{22} - F_2 - T_{23}A_{32} = 0
\]

\[
A_{21}T_{13} + A_{22}T_{23} + A_{23} - T_{23}F_3 - T_{24}A_{43} = 0
\]

\[
\vdots
\]

\[
A_{21}T_{1,k-1} + A_{22}T_{2,k-1} + A_{23}T_{3,k-1} + \cdots + A_{2,k-1} - T_{2,k-1}F_{k-1} - \cdots - T_{2,k}A_{k,k-1} = 0
\]

\[
A_{21}T_{1k} + A_{22}T_{2k} + A_{23}T_{3k} + \cdots + A_{2k} - T_{2k}F_k = 0
\]
\[ A_{32} T_{23} + A_{33} - F_3 - T_{34} A_{43} = 0 \]
\[ \vdots \]
\[ A_{32} T_{2,k-1} + A_{33} T_{3,k-1} + \cdots + A_{3,k-1} - T_{3,k-1} F_{k-1} - T_{3k} A_{k,k-1} = 0 \]
\[ A_{32} T_{2k} + A_{33} T_{3k} + \cdots + A_{3,k-1} T_{k-1,k} + A_{3k} - T_{3k} F_k = 0 \]
\[ \vdots \]
\[ A_{k-1,k-2} T_{k-2,k-1} + A_{k-1,k-1} - F_{k-1} - T_{k-1,k} A_{k,k-1} = 0 \]
\[ A_{k-1,k-2} T_{k-2,k} + A_{k-1,k-1} T_{k-1,k} + A_{k-1,k} - T_{k-1,k} F_k = 0 \]
\[ A_{k,k-1} T_{k-1,k} + A_{k,k} - F_k = 0 \] (4.58)

The set of chain Eqs. (4.58) can be solved backwards for the unknown submatrices starting with the last equation. It should be pointed out that several approaches may be used at this stage. However, two distinct approaches will be discussed in the sequel.

**Approach 1.** (Specify \( F_i \) and solve for \( T_{ij} \).) Fix the defined matrix \( F \) in Eq. (4.56) with the assigned eigenvalues and obtain \( T_{k-1,k} \) from the last equation. Then, the remaining submatrices are determined by solving sets of linear equations. The problem of solving sets of linear equations has been treated extensively in the numerical analysis literature and it suffices to say that it can be solved in a numerically stable manner. In fact, the solutions to the sets of linear equations can be obtained very effectively because of the special structure of the coefficient matrices \((A_{q,q-1}, q = 2, 3, \ldots, k)\). Consequently, by substitution of \( T_{k-1,k} \) into the second from the last chain equation, one can obtain \( T_{k-2,k} \) and \( T_{k-2,k-1} \) simultaneously, that is, in parallel. Continuing the process of substitution with the remaining chain of equations, one can arrive at the second chain in which the submatrices \( T_{1k}, T_{1,k-1}, \ldots, T_{13} \) and \( T_{12} \) are determined. At this stage, all submatrices of \( T \) are obtained and the first chain can be used to evaluate the submatrices of \( G \) again by back substitutions because of the special nonsingular structure of \( B_1 \). This operation can also be accomplished in parallel as the equation for \( G_i \) are decoupled. Finally, Eq. (4.54) can be solved for the unknown matrix \( K_h \) directly by back substitutions since \( T \) is upper triangular, and equation \( K = K_h P_h \) specifies the feedback matrix \( K \) for the original system.

**Approach 2.** (Decomposition into lower order pole assignment problems.) The decomposition shows clearly that the last equation of Eq. (4.58) has the form of an eigenvalue assignment problem with the controllable pair \((A_{k,k}, A_{k,k-1})\), that is, to find \( T_{k-1,k} \) so that \( A_{k,k} + A_{k,k-1} T_{k-1,k} \) has \( n_k \) eigenvalues equal to certain specified values. Next, the second from the last chain equation is considered and its first equation can be interpreted as a pole assignment problem, assigning the eigenvalues
of $A_{k-1,k-1} - T_{k-1,k}A_{k-1} + A_{k-1,k-2}T_{k-2,k-1}$ to those of $F_{k-1}$. This set of assignment problems can be solved recursively in a numerically efficient manner. Note that the $j$th chain requires the solution of $k + 1 - j$ matrix equations which can be done in parallel as in the case of Approach 1. Once $T_{k-1,k}$ is obtained, $F_k$ can be evaluated and the process is continued until one can arrive at the first chain, in which the submatrices of $G$ are computed. Finally, $K_h$ can be obtained from Eq. (4.54) and the feedback matrix of the original system can be recovered from $K = K_hP_h$.

**Algorithm 4.6** (Design Coordinate Required)

1. Transform the system to the controllable block Hessenberg form Eq. (4.22), Eq. (4.23) using $P_h$ defined by the transformation algorithm.
2. Specify $K_h$ and define the matrices $T$, $F_h$, and $G$ as Eqs. (4.55), (4.56), and (4.57).
3. Construct the chain Eq. (4.58).
4. Obtain $T$ based on Approach 1, or $T$ and $F_h$ by using Approach 2.
5. Compute $G$ from the first chain of Eq. (4.58).
6. Compute $K_h$ from Eq. (4.54) and $K = K_hP_h$.

**Example 4.8**

Consider a system $\dot{x}(t) = Ax(t) + Bu(t)$ which has been transformed to the following block Hessenberg form

$$ \dot{x}_h(t) = A_hx_h(t) + B_hu(t) $$

with

$$ A_h = \begin{bmatrix}
-1 & 1 & 0 & 0 & 1 & -1 & 0 & 1 \\
0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
& & & \hline
1 & -1 & 1 & -3 & -1 & 1 & -2 & -5 \\
0 & 0 & 1 & -1 & -2 & 0 & 1 & -1 \\
& & & \hline
0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 3 & 1 \\
& & & \hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & -4
\end{bmatrix} \quad B_h = \begin{bmatrix}
1 & 1 & -1 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
& & & \hline
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
& & & \hline
0 & 0 & 0
\end{bmatrix} $$

Suppose it is desired to assign the eigenvalues $-1$, $-1$, $-1$, $-1 \pm j$, $-2$, $-2$, $-3$.

**Solution** For this example the structure of the matrices $T$, $F$, and $G$ are given by

$$ T = \begin{bmatrix}
I_3 & T_{12} & T_{13} & T_{14} \\
0 & I_2 & T_{23} & T_{24} \\
0 & 0 & I_2 & T_{34} \\
0 & 0 & 0 & I_1
\end{bmatrix} \quad F = \begin{bmatrix}
F_1 & 0 & 0 & 0 \\
A_{21} & F_2 & 0 & 0 \\
0 & A_{23} & F_3 & 0 \\
0 & 0 & A_{43} & F_4
\end{bmatrix} $$

$$ G = [G_1 \quad G_2 \quad G_3 \quad G_4] $$
Let us take the first approach and select arbitrarily the matrices $F_1$, $F_2$, $F_3$, and $F_4$ according to the specified eigenvalues as

$$
F_1 = \begin{bmatrix}
-1 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & -1
\end{bmatrix} \quad F_2 = \begin{bmatrix}
0 & 1 \\
-2 & -2
\end{bmatrix} \quad F_3 = \begin{bmatrix}
-2 & 0 \\
0 & -2
\end{bmatrix} \quad F_4 = -3
$$

Then the chain Eq. (4.58) can be set up as follows:

\[
\begin{align*}
A_{11} - F_1 - T_{12}A_{21} &= B_1G_1 \\
A_{11}T_{12} + A_{12} - T_{12}F_2 - T_{13}A_{32} &= B_1G_2 \\
A_{11}T_{13} + A_{12}T_{23} + A_{13} - T_{13}F_3 - T_{14}A_{43} &= B_1G_3 \\
A_{11}T_{14} + A_{12}T_{24} + A_{13}T_{34} + A_{14} - T_{14}F_4 &= B_1G_4 \\
A_{21}T_{12} + A_{22} - F_2 - A_{23}A_{32} &= 0 \\
A_{21}T_{13} + A_{22}T_{23} + A_{23} - T_{23}F_3 - T_{24}A_{43} &= 0 \\
A_{21}T_{14} + A_{22}T_{24} + A_{23}T_{34} + A_{24} - T_{24}F_4 &= 0 \\
A_{32}T_{23} + A_{33} - F_3 - T_{34}A_{43} &= 0 \\
A_{32}T_{24} + A_{33}T_{34} + A_{34} - T_{34}F_4 &= 0 \\
A_{43}T_{34} + A_{44} - F_4 &= 0
\end{align*}
\]

Solving these equations recursively for the unknown submatrices $T_1$, starting from the last chain equation, we obtain

\[
\begin{align*}
T_{34} &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad T_{24} = \begin{bmatrix} -7 \\ -7 \end{bmatrix} \quad T_{23} = \begin{bmatrix} -4 & -4 \\ -1 & -4 \end{bmatrix} \\
T_{14} &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad T_{13} = \begin{bmatrix} -1 & -0.5 \\ 1 & 0.5 \\ -4 & -12 \end{bmatrix} \quad T_{12} = \begin{bmatrix} 0.5 & 2.5 \\ -0.5 & -2.5 \\ -2 & -3 \end{bmatrix}
\end{align*}
\]

Then we obtain directly the submatrices of $G$ from the first chain as

\[
\begin{align*}
G_1 &= \begin{bmatrix} 1 & 0 & -2 \\ 0.5 & 0.5 & 5 \\ 2 & -1 & 6 \end{bmatrix} \\
G_2 &= \begin{bmatrix} 12.5 & 11.5 \\ -10 & -10 \\ -2.5 & 1.5 \end{bmatrix} \\
G_3 &= \begin{bmatrix} -3 & -5.5 \\ -6 & -22 \\ -7 & -23.5 \end{bmatrix} \\
G_4 &= \begin{bmatrix} -7 \\ 1 \\ 0 \end{bmatrix}
\end{align*}
\]

Finally, $K$ can be recovered from Eq. (4.54) by back substitution as follows:

\[
K_1 = -G_1 = \begin{bmatrix} -1 & 0 & 2 \\ -0.5 & -0.5 & -5 \\ -2 & 1 & -6 \end{bmatrix}
\]
\[ K_2 = -G_2 - K_1T_{12} = \begin{bmatrix} -8 & 3 \\ 0 & -5 \\ -8 & -12 \end{bmatrix} \]

\[ K_3 = -G_3 - K_1T_{13} - K_2T_{23} = \begin{bmatrix} -25 & -15 \\ -19 & -58 \\ -64 & -130 \end{bmatrix} \]

\[ K_4 = -G_4 - K_1T_{14} - K_2T_{24} - K_3T_{34} = \begin{bmatrix} -55 \\ 22 \\ -10 \end{bmatrix} \]

yielding

\[ K_h = \begin{bmatrix} -1 & 0 & 2 & -8 & -3 & -25 & -15 & -55 \\ -0.5 & -0.5 & -5 & 0 & -5 & -19 & -58 & 22 \\ -2 & 1 & -6 & -8 & -12 & -64 & -130 & -10 \end{bmatrix} \]

**CAD Example 4.3**

Let us consider the system of CAD Ex. 4.1, and design the controller such that eigenvalues \(-1 \pm j, -1, -0.5, \) and \(-0.2\) are assigned using the previous algorithm.

One can apply PP2MIMO program of CAD-MCS to solve this problem. Here, we use CONTROL.lab, as we used in CAD Ex. 4.1, in order to preserve the continuity. The pair \((A,B)\) is transformed to \((A_h, B_h)\) as given in CAD Ex. 4.1. Let the desired eigenvalues be distributed to \(F_1, F_2, \) and \(F_3\). Then we have

\[
\begin{align*}
f1 \\
F1 &= \\
&= \\
&= \\
&\begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
f2 \\
F2 &= \\
&= \\
&= \\
&\begin{bmatrix} -1.0000 & .0000 \\ .0000 & -.5000 \end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
f3 \\
F3 &= \\
&= \\
&= \\
&= -.2000
\end{align*}
\]

\[
\begin{align*}
t \\
T &= \\
&= \\
&= 
\end{align*}
\]
130  Chap. 4  State Feedback and Observer Design

\[
\begin{bmatrix}
1.0000 & .0000 & 4.8527 & -.4558 & 1.4090 \\
.0000 & 1.0000 & 3.9184 & 1.0796 & .9608 \\
.0000 & .0000 & 1.0000 & .0000 & .2418 \\
.0000 & .0000 & .0000 & 1.0000 & -.0184 \\
.0000 & .0000 & .0000 & .0000 & 1.0000 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
.0000 & .0000 & .0000 & .0000 & .0000 \\
\end{bmatrix}
\]

\[
g
\]

\[
G = \begin{bmatrix}
44.0719 & 25.5104 & 37.2935 & 33.5736 & 47.1152 \\
6.8478 & 10.9462 & 47.9942 & 31.2378 & 27.5781 \\
\end{bmatrix}
\]

\[
k = -g\text{inv}(t)\ast p
\]

\[
K = \begin{bmatrix}
-42.2154 & 96.2089 & -197.8922 & 170.2621 & -36.5383 \\
\end{bmatrix}
\]

\[
norm(k)
\]

\[
\text{ANS} = \begin{bmatrix}
285.6901 \\
\end{bmatrix}
\]

\[
ac = a + b\ast k
\]

\[
AC = \begin{bmatrix}
-.1094 & .0628 & .0000 & .0000 & .0000 \\
-.7729 & 7.0260 & -16.5839 & 13.5863 & -4.8768 \\
-.1587 & 5.8120 & -12.5903 & 9.5486 & -4.4916 \\
-.0125 & .3671 & -.9580 & 1.0319 & -.5591 \\
\end{bmatrix}
\]

\[
m
\]

\[
M = \begin{bmatrix}
\end{bmatrix}
\]

\[
\text{COLUMNS 1 THRU 4}
\]

\[
\begin{bmatrix}
-.0196 - .0204i & .0113 - .0506i & -.9561 - .1369i & .6662 + .3325i \\
.6030 - .0238i & -.9650 + .5369i & 1.3905 + .1991i & -9.4326 - 4.7073i \\
.5896 + .2352i & -.6976 + .9289i & .9681 + .1386i & -5.7148 - 2.8520i \\
\end{bmatrix}
\]
\[
\begin{align*}
.3655 + .3036i & \quad -.2822 + .8225i & \quad .3523 + .0504i & \quad -1.7780 & \quad .8873i \\
.0541 - .0577i & \quad -.1394 - .0387i & \quad -.1159 - .0166i & \quad -.3848 & \quad .1921i \\
\end{align*}
\]

\text{COLUMNS  5 THRU 5}

\[
\begin{align*}
-.1917 & \quad -.1956i \\
1.1895 & \quad + 1.2140i \\
-1.1218 & \quad -1.1448i \\
-1.3728 & \quad -1.4011i \\
1.6193 & \quad + 1.6526i \\
\end{align*}
\]

\(\text{cond}(m)\)

\[
\text{ANS} = 298.6754
\]

\(< >\)

\(\text{x}\)

\[
\begin{align*}
\text{x} & = \\
\end{align*}
\]

\text{COLUMNS  1 THRU 4}

\[
\begin{align*}
-.0196 & \quad -.0204i & \quad .0062 & \quad -.0276i & \quad -.4779 & \quad -.0684i & \quad .0532 & \quad + .0266i \\
.6030 & \quad -.0238i & \quad -.5273 & \quad + .2934i & \quad .6950 & \quad + .0995i & \quad -.7537 & \quad -.3761i \\
.5896 & \quad + .2352i & \quad -.3812 & \quad + .5076i & \quad .4839 & \quad + .0693i & \quad -.4567 & \quad -.2279i \\
.3655 & \quad + .3036i & \quad -.1542 & \quad + .4494i & \quad .1761 & \quad + .0252i & \quad -.1421 & \quad -.0709i \\
.0541 & \quad -.0577i & \quad -.0762 & \quad -.0212i & \quad -.0579 & \quad -.0083i & \quad -.0308 & \quad -.0153i
\end{align*}
\]

\text{COLUMNS  5 THRU 5}

\[
\begin{align*}
-.0499 & \quad -.0510i \\
.3099 & \quad + .3163i \\
-.2922 & \quad -.2983i \\
-.3577 & \quad -.3650i \\
.4219 & \quad + .4305i
\end{align*}
\]

\(\text{cond}(x)\)

\[
\text{ANS} = 68.3633
\]

\(< >\)

The feedback gain matrix for MIMO case is not unique. Alternate \(k_i\)'s for eigenvalue assignment along with their norms, the condition numbers of normalized eigenvector matrix and the computed closed-loop eigenvalues are also provided below. It is interesting to point out that \(k_4\) obtained from direct algorithm, PP2MIMO of CAD-MCS, yields similar results as iterative algorithms of Kautsky et al. (1985).
\[ k_1 = \]
\[
\begin{array}{cccccc}
-47.6900 & 102.0100 & -213.7000 & 179.8600 & -42.5520 \\
\end{array}
\]

eig(a + b * k1)

\[
\text{ans} =
\]
\[
\begin{array}{cccc}
-0.9999 & + & 1.0000i \\
-0.9999 & - & 1.0000i \\
-1.0001 \\
-0.2000 \\
-0.4999 \\
\end{array}
\]

\[ \text{norm}(k1) = 311.5 \quad \text{cond}(x) = 39.4 \]

\[ \text{Method 2/3 of Kautsky et al. (1985)} \]
\[ k_2 \]

\[
\begin{array}{cccccc}
-159.6800 & 69.8440 & -165.2400 & 125.2300 & -45.7480 \\
-99.3480 & 7.9892 & -14.1580 & -5.9382 & -1.2542 \\
\end{array}
\]

eig(a + b * k2)

\[
\text{ans} =
\]
\[
\begin{array}{cccc}
-1.0008 & + & 1.0015i \\
-1.0008 & - & 1.0015i \\
-0.5003 \\
-0.9984 \\
-0.2000 \\
\end{array}
\]

\[ \text{norm}(k2) = 283.1 \quad \text{cond}(X) = 66.1 \]

\[ \text{Method of Chu (1986)} \]
\[ k_3 \]

\[ k_3 = \]
- 102.0000 10.1400 - 31.7400 22.8300 - 3.3310
- 76.7400 - 64.0900 130.6000 - 119.5000 41.2200

\[ \text{eig}(a + b^*k3) \]

\[ \text{ans} = \]

\[ -1.0016 + 0.9887i \]
\[ -1.0016 - 0.9887i \]
\[ -0.9891 \]
\[ -0.5032 \]
\[ -0.1986 \]

\[ / / \text{norm}(k3) = 207.4 \quad \text{cond}(X) = 2753.2 \]

\[ / / \text{Algorithm 4.6 using PP2MIMO of CAD-MCS} \]

\[ k4 \]

\[ k4 = \]

\[ -46.0855 \quad 121.3856 \quad -274.1013 \quad 276.4817 \quad -18.0365 \]
\[ -20.8666 \quad 56.7374 \quad -112.6042 \quad 106.9468 \quad -0.1208 \]

\[ \text{eig}(a + b^*k4) \]

\[ \text{ans} = \]

\[ -1.0001 + 1.0001i \]
\[ -1.0001 - 1.0001i \]
\[ -0.5000 \]
\[ -0.2000 \]
\[ -1.0001 \]

\[ / / \text{norm}(k4) = 443.2 \quad \text{cond}(X) = 79.1 \]

Let us conclude this section with a method (Porter and D'azzo (1977, 1978), D'azzo and Houpis (1988)) which assigns the eigenvalues as well as eigenvectors without the requirement of initial transformation to canonical form.

Suppose the desired eigenvalues are given by

\[ \lambda(A + BK) = \{\lambda_1, \lambda_2, \ldots, \lambda_n\} \]  \hspace{1cm} (4.59)

and the associated set of eigenvectors are denoted by

\[ v(A + BK) = \{v_1, v_2, \ldots, v_n\} \]  \hspace{1cm} (4.60)

Then we can write the obvious relation
\begin{align}
(A + BK) v_i &= \lambda_i v_i \\
\text{which can also be written more compactly as} \\
[A - \lambda_i I B] \begin{bmatrix} v_i \\ g_i \end{bmatrix} &= 0
\end{align}

(4.62)

where

\[ g_i = K v_i \]

(4.63)

The determination of the feedback gain \( K \) is clear that it requires the evaluation of the set of vectors \( [v_i^T g_i^T]^T \) which should lie in the kernel or nullspace of the matrix

\[ S(\lambda_i) = [A - \lambda_i I B] \]

(4.64)

Thus, the required steps for computing the feedback gain \( K \) are summarized as an algorithm.

\textbf{Algorithm 4.7.} (Design Coordinate Not Required)

1. Corresponding to each desired eigenvalue, construct the matrix \( S(\delta_i) \) given by (4.64).
2. Augment \( S(\lambda_i) \) with sufficient rows of zeros to form a square matrix and use elementary row operations to put it into Hermite form.
3. Interchange the rows, if necessary, so that the leading ones in each row are on the principal diagonal.
4. Replace zeros on the principal diagonal by \(-1\) and denote the new matrix by \( \hat{S}(\lambda_i) \).
5. Select the columns of the matrix \( \hat{S}(\lambda_i) \) which contains \(-1\) on the principal diagonal to span the nullspace of \( S(\lambda_i) \).
6. Set up Eq. (4.63) for all \( i \) and obtain \( K \) from

\[ K = [g_1 \ g_2 \ \cdots \ g_n] [v_1 \ v_2 \ \cdots \ v_n]^{-1} = GV^{-1} \]

(4.65)

An alternative approach to find \( \text{Ker} S(\lambda_i) \) is to form the augmented matrix

\[
\begin{bmatrix}
A - \lambda I & B \\
\hline
I_{n+m}
\end{bmatrix}
\]

and use elementary column operation to obtain \( m \) zero columns of this matrix. Then the columns below these zero columns span the \( \text{Ker} S(\lambda_i) \).

\textbf{Example 4.9}

Consider the controllable system

\[ \quad \]
\[
\dot{x} = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
1 & 1 & -1
\end{bmatrix} x + \begin{bmatrix}
1 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} u
\]
\[
y = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix} x
\]

which has the eigenvalues \(-1, -1, 1\). It is desired to shift the eigenvalues to \(-2, -3, -5\) using Algorithm 4.7.

Solution

Step 1.

\[
S(\lambda_i) = [A - \lambda_i I \ B] = \begin{bmatrix}
-\lambda_i & 1 & 0 & 1 & 0 \\
0 & -\lambda_i & 1 & 0 & 0 \\
1 & 1 & -\lambda_i - 1 & 0 & 1
\end{bmatrix}
\]

\[
\lambda_1 = -2
\]
\[
\lambda_2 = -3
\]
\[
\lambda_3 = -5
\]

Steps 2., 3., 4.

\[
\hat{S}(-2) = \begin{bmatrix}
1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & \frac{1}{3} & \frac{1}{3} \\
0 & 1 & 0 & \frac{1}{3} & \frac{2}{3} \\
0 & 0 & 0 & -\frac{2}{3} & \frac{4}{3} \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]

\[
\hat{S}(-3) = \begin{bmatrix}
1 & 0 & 0 & \frac{5}{16} & \frac{1}{16} \\
0 & 1 & 0 & \frac{1}{16} & -\frac{3}{16} \\
0 & 0 & 1 & -\frac{3}{16} & \frac{9}{16} \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]

\[
\hat{S}(-5) = \begin{bmatrix}
1 & 0 & 0 & \frac{19}{96} & \frac{1}{96} \\
0 & 1 & 0 & \frac{1}{96} & -\frac{5}{96} \\
0 & 0 & 1 & -\frac{5}{96} & \frac{25}{96} \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]
Step 5. In order to eliminate fractions, we multiply the last two columns of \( \hat{S}(\lambda_i) \) by 3, 16, and 96 respectively and generate

\[
\text{Ker } S(\lambda_i) = \text{span } [s_1(\lambda_i), s_2(\lambda_i)]
\]

where \( s_1 \) and \( s_2 \) represent the last two columns of \( \hat{S}(\lambda_i) \). The nullspace vectors can be selected by any desired linear combination of \( s_1(\lambda_i) \) and \( s_2(\lambda_i) \), that is,

\[
\begin{bmatrix}
    v_i \\
    g_i
\end{bmatrix} = \alpha_1 s_1(\lambda_i) + \alpha_2 s_2(\lambda_i)
\]

\[\lambda_1 = -2 : \begin{bmatrix} v_1 \\ g_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ -2 \\ -3 \\ 0 \end{bmatrix}\]

\[\lambda_2 = -3 : \begin{bmatrix} v_2 \\ g_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \\ -3 \\ -16 \\ 0 \end{bmatrix}\]

\[\lambda_3 = -5 : \begin{bmatrix} v_3 \\ g_3 \end{bmatrix} = \begin{bmatrix} 1 \\ -5 \\ 25 \\ 0 \\ -96 \end{bmatrix}\]

Step 6. Use Eq. (4.65) to compute \( K \) as

\[
K = \begin{bmatrix} -3 & -16 & 0 \\ 0 & 0 & -96 \end{bmatrix} \begin{bmatrix} 1 & 5 & 1 \\ 1 & 1 & -5 \\ -2 & -3 & 25 \end{bmatrix}^{-1}
\]

\[
= \begin{bmatrix} -3.1818 & 0.7273 & 0.2727 \\ -1.4545 & -10.1818 & -5.8182 \end{bmatrix}
\]

Suppose we are interested in a complex assignment. Then, to illustrate, we can avoid arithmetic completely if we use the following format.

For the purpose of illustration, let's consider \( \lambda_1 \) to be real and \( \lambda_2, \lambda_3 \) to be complex conjugates. Following the steps of the algorithm, we arrive at

\[
\begin{bmatrix}
    v_1 \\
    g_1
\end{bmatrix}, \begin{bmatrix}
    v_2 \\
    g_2
\end{bmatrix}, \begin{bmatrix}
    v_3 = v_2^* \\
    g_3 = g_2^* \end{bmatrix}
\]

which leads to the feedback gain \( K \) as
\[ K = [g_1 \quad Re(g_2) \quad Im(g_2)] [v_1 \quad Re(v_2) \quad Im(v_2)]^{-1} \]

This format of computing \( K \) can be extended for the case when there is more than one set of complex eigenvalues and eigenvectors.

### 4.3.5 Technical Discussion and Computational Aspects of Pole Placement Problem

In previous subsections, we discussed the eigenvalue (pole) placement problem and provided several algorithms to solve it. In all algorithms, we chose the eigenvalues of the closed-loop system arbitrarily. However, it is natural to ask how to choose a set of desired eigenvalues for the purpose of improving the response of the system. The improvement criteria is normally translated to a performance index and the problem is formulated as an optimization one. Let us explore this by the so-called regulator problem which will be considered in more details in Chap. 6. The basic problem is to obtain a control \( u(t) \) such that the state \( x(t) \), which has been displaced, returns to zero state as quickly as possible. It has been shown at least theoretically that we can use a finite input to instantaneously restore the state to zero. In practice, this means that we need very high (infinite) energy inputs. On the other hand, we can use the methodology of pole placement and obtain a finite energy input so that the state decays to zero as fast as desired. These facts suggest to consider a trade-off between the rate of state and the energy of the input. In the quadratic regulator problem, this can be achieved by choosing \( u \) such that the quadratic performance index

\[ J = \int_0^\infty [x^T(t)Q x(t) + u^T(t)R u(t)] \, dt \]

is minimized subject to the constraint

\[ \dot{x}(t) = A x(t) + B u(t) \]

It turns out that the unique feedback control law which results in a unique set of eigenvalues is given by

\[ u(t) = -R^{-1} B^T K x(t) \]

where \( K \) is the solution of the algebraic matrix Riccati equation. It has been long recognized that the linear quadratic regulator for single input systems has, among others, nice robustness properties against uncertainties (Anderson and Moore (1971)). These robustness properties have been generalized for multi-input systems as well (Safonov and Athans (1977), Safanov (1980)). Although new results in this direction continue to be found, we shall discuss some computational aspects of algorithms provided in previous subsections. In particular, we focus our attention on three algorithms. Algorithm 4.5 requires the transformation of \((A, B)\) into a controllable canonical form which requires the computation of \( A^k B \) for \( k = 1, \ldots, n - 1 \). For large \( k \), we may change the problem into a less well-conditioned problem, as discovered before. Thus, the algorithm is generally not satisfactory for higher-order
systems but it may be considered for small size problems. Algorithm 4.6 requires the transformation of the pair \((A, B)\) into a block Hessenberg form which uses numerically preferable orthogonal transformations. Although a few computationally reliable algorithms for eigenvalue assignment based on the Hessenberg form have been developed (Miminis and Paige (1982), Petkov et al. (1986), Patel and Misra (1984), Chu (1986)), they do need to be improved for better robust solutions, that is, the assigned eigenvalues are sensitive to perturbations in the closed-loop system. The robustness of the closed-loop system can be measured by the condition number of the eigenvector matrix associated with the assigned eigenvalues. This condition number should be as small as possible in order to improve robustness (Kautsky et al. (1985), Cavin and Bhattacharyya (1983)). On the other hand, to improve the transient response of the closed-loop system, the norm of the feedback matrix should be small (Bhattacharyya (1987)). This criteria is related to the requirement for robustness and the norm of the feedback matrix may be expected to be reasonably bounded if the condition number is small, but minimizing the feedback norm does not necessarily give the optimal sensitivity for the assigned eigenvalues (poles). On the other hand, the designer may use the available degrees of freedom to achieve some other goals. Therefore, in general the problem may be formulated as a multiobjective mathematical programming problem which requires expensive iterative algorithms. To avoid this and still obtain a numerically stable algorithm with a robust solution is not a simple problem. The Algorithm 4.6 (Shafai and Bhattacharyya (1988)) described in the previous section has attempted to achieve these goals directly to the extend possible. The initial step uses a numerically stable algorithm to transform the system to upper block Hessenberg form. A natural consequence of the decomposition is that the feedback matrix becomes small and it improves the conditioning or robustness of the higher-order system. Moreover, each smaller subproblem can be solved using available numerically stable methods with robust solutions. Two approaches to solve the so-called "chain equation" have been proposed. Although both approaches are simple and efficient, a trade-off between robustness and numerical stability is incorporated with them where the first approach is expected to result in a more robust solution. The matrix \(T\) is obviously nonsingular and will be better conditioned if its upper triangular part is minimized in some norm. Thus, in order to achieve this goal, the set of Eq. (4.58) should be solved carefully by the usual least-squares technique, that is, QR, such that the minimum norm requirement is established. With a well-conditioned \(T\) and the minimized solution, it is expected also from Eq. (4.54) that the norm of the feedback matrix \(K_h\) or equivalently \(K\) will be small. This in turn improves the dynamic response of the system as well. An additional feature of the algorithm is that the solution of each chain can be done in parallel and the more one proceeds to upper chains the more one uses this parallel mechanism. Thus, the algorithm is fast and can be tailored to present any future parallel implementations. Finally, Algorithm 4.7 provides a possibility of assigning eigenvalues as well as eigenvectors. An important advantage of this algorithm is the ability to select eigenvectors associated with each assigned eigenvalue for the purpose of response shaping and robust improvement. Since the set of eigenvectors can be
selected from the Kernel of nullspace of certain matrix, the robust improvement can be achieved by choosing them so that they are mutually orthogonal as much as possible. It is well known from numerical analysis that a set of orthonormal vectors generates a perfect conditional matrix. Consequently, the selection of such eigenvectors which in turn is used to compute the required feedback gain matrix leads to a robust solution for closed-loop systems.

4.4 OBSERVER DESIGN

In the previous section, we discussed the problem of pole assignment by using state feedback and provided several solution procedures for it. The assignment can be achieved under the assumption of controllability of the system and the feedback implementation requires availability of the states. In practice, this latter requirement is not always met because all the state variables are not accessible to direct measurement due to technological limitation or because the number of measuring devices is limited due to the economical constraint. Thus, in order to apply state feedback, an estimate of the state variables has to be obtained. The device for reconstructing such an estimate of the state vector is called a state estimator or a state observer. This latter terminology is used only for deterministic systems. In this section, we shall discuss various techniques to estimate the state vector by using the available inputs and outputs.

4.4.1 Full-Order State Observer Design

Consider the linear time-invariant system described by Eqs (4.7), (4.8). Here we assume, with no loss of generality, that the matrix $D$ is identically zero (If $D \neq 0$, we define the output as $y - Du$). Thus, we have

\begin{align}
\dot{x}(t) &= A \, x(t) + B \, u(t) \\
y(t) &= C \, x(t)
\end{align}

One possible way to obtain the state vector is to differentiate the output equation $n - 1$ times and use successively the state equation to obtain

\[ Y(t) = R_O \, x(t) + T \, U(t) \]

where $R_O$ is the observability matrix and $T$ is the Toeplitz transmission matrix defined as

\[ R_O = [C^T \ (CA)^T \ \cdots \ (CA^{n-1})^T]^T \]

\[ T = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{n-1} \end{bmatrix} \begin{bmatrix} \cdots & h_1 & h_0 \end{bmatrix} \]
with $h_0 = 0$, $h_i = CA^{i-1}B$ known as Markov parameters. The variables $Y(t)$ and $U(t)$ are defined by

$$
Y(t) = [y^T(t) \ y^T(t) \ \cdots \ (y^{(n-1)})^T(t)]^T
$$

$$
U(t) = [u^T(t) \ u^T(t) \ \cdots \ (u^{(n-1)})^T(t)]^T
$$

The system is assumed to be observable, that is, the matrix $R_O$ has full rank. Thus, $R_O^T R_O$ is nonsingular and it is not difficult to show that $x(t)$ can be obtained as

$$
x(t) = (R_O^T R_O)^{-1} R_O^T [Y(t) - T U(t)]
$$

This solution is not acceptable from the practical point of view. The reason is that noise is always present in $u$ and $y$ and differentiation causes considerable error in the process, leading to severe distortion of the estimated state. Furthermore, pure differentiators are not easy to realize. Another possible way to estimate the states is to simulate a model with accessible states, which has the same dynamic as the original system. Thus, with the complete knowledge of $A$, $B$, and $C$, one can construct a so-called open-loop estimator as shown in Fig. 4.3.

It is obvious that if the initial state $x(0)$ and $x_e(0)$ were the same, the open-loop estimator would provide an exact estimate of $x(t)$, namely, $x_e(t) = x(t)$ for all $t > 0$. However, this method has two drawbacks. First, we do not have the initial state $x(0)$ and even computable from Eq. (4.73) one should initialize the estimator each time for its operation. Second, if we let $\hat{x}(t)$ denote the state reconstruction error vector defined by

![Figure 4.3 An open-loop estimator.](image-url)
\[ x(t) = x(t) - x_e(t) \]  

then the dynamic behavior of \( \dot{x}(t) \) can be determined from

\[ \dot{x}(t) = \dot{x}(t) - \dot{x}_e(t) = Ax + Bu - (Ax_e + Bu) = A\dot{x}(t) \]

as

\[ \dot{x}(t) = e^{At} \dot{x}(0) \quad t \geq 0 \]

which would increase exponentially for unstable matrix \( A \) and \( \dot{x}(0) \neq 0 \).

To overcome these shortcomings, make use of both input and output of the system to drive the estimator. Since \( y = Cx \) and \( y_e = Cx_e \) are available in the open-loop estimator, compare them and use the difference as a correction term which would then be multiplied by an \( n \times r \) real constant matrix \( L \) and fed back into the input of the integrators of the estimator. This estimator, shown in Fig. 4.4, is a closed-loop estimator which was first introduced by Luenberger (1964) and it is commonly referred to as an asymptotic state estimator or simply a full-order state observer. These names are justified by the facts that the estimated state approaches the true state asymptotically and that the dimension of the observer is \( n \) as the system order. The dynamical equation of the state observer is given by

\[ \dot{x}_e(t) = A \dot{x}_e(t) + B u(t) + L [y(t) - Cx_e(t)] \]

where matrix \( L \) is yet to be determined. Thus, in order to design the observer, one needs to determine the unknown parameter \( L \) and fulfill the stability requirement. Obviously, the Eq. (4.77) can also be written as

\[ y(t) = Cx(t) + L \begin{bmatrix} 0 \\ x_e(t) \end{bmatrix} \]

**Figure 4.4** A closed-loop estimator (Luenberger Observer).
\[ \dot{x}_e(t) = (A - LC) x_e(t) + Ly(t) + Bu(t) \]  \hspace{1cm} (4.78)

and the configuration of Fig. 4.4 can be modified accordingly. To further illuminate the main advantage of the closed-loop estimator, let us define the state reconstruction error as

\[ \tilde{x}(t) = x(t) - x_e(t) \]  \hspace{1cm} (4.79)

Then, it is not difficult to obtain the error dynamic by subtracting Eq. (4.78) from Eq. (4.66) as

\[ \dot{\tilde{x}}(t) = (A - LC) \tilde{x}(t) \]  \hspace{1cm} (4.80)

which leads to the solution

\[ \tilde{x}(t) = e^{(A - LC)t} \tilde{x}(0) \]  \hspace{1cm} (4.81)

Now, if the observer gain matrix \( L \) is chosen such that the eigenvalues of \( A - LC \) have negative real parts, then the state error vector \( \tilde{x}(t) \) will decay exponentially as time approaches infinity even for large initial error. Thus, the steady state value of \( \tilde{x}(t) \) for any initial condition is zero, that is,

\[ \lim_{t \to \infty} \tilde{x}(t) = 0 \]

**Theorem 4.3.** An observer with arbitrary eigenvalues, described by Eq. (4.78), can be constructed for the system Eq. (4.66), Eq. (4.67) if and only if the system is observable.

**Proof.** To prove the theorem, let us transpose \( A - LC \), that is, \( (A - LC)^T = A^T - C^T L^T \) which is of the form \( \hat{A} + \hat{B} \hat{K} \), where \( \hat{A} = A^T, \hat{B} = C^T \), and \( \hat{K} = -L^T \). Since the determinant of any matrix is equal to the determinant of its transpose, we can write

\[ \det [(sI - (A - LC))^T] = \det [(sI - (A - LC))^T] \]

which shows that the eigenvalues of \( A - LC \) are exactly those of \( \hat{A} + \hat{B} \hat{K} \). Now, if \( (A, C) \) is an observable pair, then \( (A^T, C^T) = (\hat{A}, \hat{B}) \) is a controllable pair and we can find a real \( \hat{K} \) so that \( \hat{A} + \hat{B} \hat{K} \) has the set of \( n \) desired eigenvalues. This can be achieved by using any eigenvalue (pole) assignment algorithm provided in the previous section. Finally, \( L = -\hat{K}^T \) and the observer is completely determined.

Although the design of observer can be considered as the dual of pole assignment design, one can provide direct design procedures for the observable system. We leave this as an exercise, instead two appropriate design coordinates are provided, as in the case of pole placement, which are useful for direct design procedures for full-order observer (see Prob. 4.20) as well as for reduced-order observer design in the next subsection.
Observable canonical form.\textsuperscript{5} If the system Eq. (4.66), Eq. (4.67) is observable, then the observability matrix

\[
R_o = \begin{bmatrix}
C \\
CA \\
\vdots \\
CA^{n-1}
\end{bmatrix}
\] (4.82)

has \(n\) linearly independent rows. By using these independent rows or their linear combination chosen as basis, various observable canonical forms known also as Luenberger canonical forms can be obtained. Here, let the nonsingular \(n \times n\) transformation matrix \(P_o\) be constructed from the rows of \(R_o\) as follows:

**Algorithm 4.8**

1. Search the rows of \(R_o\) from top to bottom and select those rows which are linearly independent from previously selected ones.

2. Arrange the \(n\) linearly independent row vectors to form a new matrix \(T_o\) as

\[
T_o = \begin{bmatrix}
C_i \\
C_iA \\
\vdots \\
C_iA^{\nu_i-1} \\
\hline
C_r \\
C_rA \\
\vdots \\
C_rA^{\nu_r-1}
\end{bmatrix}
\] (4.83)

where the integers \(\nu_1, \ldots, \nu_r\) are the observability indices with

\[
\nu_o = \max_i \nu_i, \quad i = 1, \ldots, r
\]

defined as observability index and

\[
\sum_{i=1}^r \nu_i = n
\]

3. Let \(T_o^{-1}\) be represented in terms of its column as

\[
T_o^{-1} = [t_1 \quad t_2 \quad \cdots \quad t_n]
\] (4.84)

and let \(t_i\) be the \(i\)th column of \(T_o^{-1}\), where

\[
l_i = \sum_{j=1}^i \nu_j, \quad i = 1, \ldots, r
\] (4.85)

\textsuperscript{5} See also Sec. 2.5.
4. Form the required transformation matrix $P_O$ as

$$P_O = [t_{l_1} A t_{l_1} \cdots A^{n_1 - 1} t_{l_1} ; \ldots ; t_{l_r} A t_{l_r} \cdots A^{n_r - 1} t_{l_r}]^{-1} \quad (4.86)$$

Then $x_o(t) = P_O x(t)$ transforms the system to the block observable canonical form

$$\dot{x}_o(t) = A_o x_o(t) + B_o(t) \quad (4.87)$$

$$y(t) = C_o x_o(t) \quad (4.88)$$

where

$$A_o = P_o A P_o^{-1} = \begin{bmatrix} A_{11} & \cdots & A_{1r} \\ \vdots & & \vdots \\ A_{rl} & \cdots & A_{rr} \end{bmatrix}, \quad B_o = P_o B = \begin{bmatrix} B_1 \\ \vdots \\ B_r \end{bmatrix}$$

$$C_o = C P_o^{-1} = [C_1 \cdots C_r]$$

and

$$A_{ii} = \begin{bmatrix} 0 & 0 & \cdots & 0 & x \\ 1 & 0 & \cdots & 0 & x \\ 0 & 1 & \cdots & 0 & x \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & x \end{bmatrix} \quad v_i \times v_i \quad i = 1, \ldots, r \quad (4.89)$$

$$A_{ij} = \begin{bmatrix} 0 & x \\ x & \vdots \\ \vdots & \vdots \\ x & \vdots \\ x \end{bmatrix} \quad v_i \times v_j \quad (4.90)$$

$$C_i = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ x \\ \vdots \\ x \\ x \end{bmatrix} \quad \text{row } i \quad (4.91)$$

The matrix $A_o$ is in block companion form and the matrix $B_o$ has no special form under this transformation. We also note that the nonzero entries, $x$, in the matrix $C_0$ can be replaced by zeros using a nonsingular transformation applied to the system outputs.

As in the case of controllable canonical form, we may not in general obtain the transformed system Eqs. (4.87) to (4.91) by using the steps of this algorithm, instead, one should use Popov procedure to construct the required canonical form. This is illustrated for the following example.
Example 4.10

Consider the observable system

\[
A = \begin{bmatrix}
-1 & 1 & 2 & 1 & 0 \\
0 & -2 & 1 & 0 & 1 \\
0 & 0 & -3 & 0 & 0 \\
0 & 0 & 0 & -4 & 1 \\
0 & 0 & 0 & 0 & -5 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
1 & -1 \\
0 & 1 \\
1 & 1 \\
0 & 0 \\
-2 & 0 \\
\end{bmatrix}, \\
C = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Obtain its observable canonical form.

Solution  \(R_0\) is given by

\[
R_0 = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
0 & -2 & 1 & 0 & 1 & -1 \\
-1 & 1 & 2 & 1 & 5 & 0 \\
0 & 4 & -5 & 0 & -7 & 0 \\
1 & -3 & -7 & -5 & 27 & 1 \\
0 & -8 & 19 & 0 & 39 & -1 \\
-1 & 7 & 20 & 21 & -143 & -1 \\
0 & 16 & -65 & 0 & -203 & 0 \\
1 & -15 & -55 & -85 & 743 & 1 \\
\end{bmatrix}
\]

\(C_1\), \(C_2\), \(C_1A\), \(C_2A\), \(C_1A^2\), \(C_2A^2\), \(C_1A^3\), \(C_2A^3\), \(C_1A^4\), \(C_2A^4\)

The observable canonical form \(T_0\) is obtained

\[
T_0 = \begin{bmatrix}
C_1 \\
C_1A \\
C_1A^2 \\
C_2 \\
C_2A \\
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & -2 & 1 & 0 & 1 \\
0 & 4 & -5 & 0 & -7 \\
1 & 0 & 0 & 0 & 1 \\
-1 & 1 & 2 & 1 & 5 \\
\end{bmatrix}
\]

with observability indices \(v_1 = 3\), \(v_2 = 2\) and

\[
T_0^{-1} = \begin{bmatrix}
3 & 2.5 & 0.5 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 \\
5 & 3.5 & 0.5 & 0 & 0 \\
7 & 8 & 2 & 1 & 1 \\
-3 & -2.5 & -0.5 & 0 & 0 \\
\end{bmatrix}
\]

with \(l_1 = v_1 = 3\), \(l_2 = v_1 + v_2 = 5\). Therefore
\[ P_0 = \begin{bmatrix} t_3 & At_3 & A^2t_3 & t_3 & At_3 \end{bmatrix}^{-1} \]

\[
= \begin{bmatrix}
0.5 & 2.5 & -14 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 \\
0.5 & -1.5 & 4.5 & 0 & 0 \\
2 & -8.5 & 36.5 & 1 & -4 \\
-0.5 & 2.5 & -12.5 & 0 & 0 \\
\end{bmatrix}^{-1} \\
= \begin{bmatrix}
0 & 15 & 5 & 0 & 3 \\
0 & 8 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 \\
4 & -52.5 & -21.5 & 1 & -13.5 \\
1 & -13.5 & -5 & 0 & -4 \\
\end{bmatrix}
\]

and

\[
A_0 = P_0A_P_0^{-1} = \begin{bmatrix}
0 & 0 & -30 & 0 & 0 \\
1 & 0 & -31 & 0 & 0 \\
0 & 1 & -10 & 0 & 0 \\
0 & 0 & 55 & 0 & -4 \\
0 & 0 & 13 & 1 & -5 \\
\end{bmatrix}, \quad B_0 = P_0B
\]

\[
C_0 = CP_0^{-1} = \begin{bmatrix}
0 & 0 & 1 & 0 & 0 \\
0 & 5 & -26.5 & 0 & 1 \\
\end{bmatrix}
\]

It is clear that this transformed system does not have the required structure. Therefore, the Popov procedure can be used to obtain the observable canonical form. Since \( v_1 = 3 \) and \( v_2 = 2 \), we have two chains to generate as follows.

For the first chain, we have

\[
C_1A^3 = \alpha_1C_2 + \alpha_2C_2A + \alpha_3C_1 + \alpha_4C_1A + \alpha_5C_1A^2
\]

which can be solved for \( \alpha_i \) as

\[
[\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5] = [C_1A^3]^{-1}
\]

\[
= [0 \quad 0 \quad -30 \quad -31 \quad -10]
\]

and consequently, we obtain three out of five linearly independent basis vectors \( e_{11}, e_{12}, e_{13} \) as

\[
(C_1A^2 + 31C_1 + 10C_1A)A = -30C_1 \\
e_{11}
\]
\[(C_4A + 10C_1)A = e_{11} - 31C_1\]
\[e_{12}\]
\[C_1A = e_{12} - 10C_1\]
\[e_{13}\]

For the second chain we have
\[C_2A^2 = \beta_1C_2 + \beta_2C_2A + \beta_3C_1 + \beta_4C_1A + \beta_5C_1A^2\]

which can be solved for \(\beta_i\) as
\[
[\beta_1 \ \beta_2 \ \beta_3 \ \beta_4 \ \beta_5] = [C_2A^2]^{-1}
\]
\[
= [-4 \ -5 \ -1 \ -4.5 \ -1.5]
\]

and we obtain the remaining two vectors \(e_{21}, e_{22}\) as
\[(C_2A + 1.5C_1A + 4.5C_1 + 5C_2)A = -C_1 - 4C_2\]
\[e_{21}\]
\[(C_2 + 1.5C_1)A = e_{21} - 4.5C_1 - 5C_2\]
\[e_{22}\]

The required transformation matrix is then given by
\[
P_0 = \begin{bmatrix}
e_{11} \\
e_{12} \\
e_{13} \\
e_{21} \\
e_{22}
\end{bmatrix} = \begin{bmatrix}
0 & 15 & 5 & 0 & 3 \\
0 & 8 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 \\
4 & 2.5 & 3.5 & 1 & 1.5 \\
1 & 1.5 & 0 & 0 & 1
\end{bmatrix}
\]

and
\[
A_0 = P_0A P_0^{-1} = \begin{bmatrix}
0 & 0 & -30 & 0 & 0 \\
1 & 0 & -31 & 0 & 0 \\
0 & 1 & -10 & 0 & 0 \\
0 & 0 & 5 & 0 & -4 \\
0 & 0 & 3 & 1 & -5
\end{bmatrix}, \quad B_0 = P_0B
\]
\[
C_0 = C P_0^{-1} = \begin{bmatrix}
0 & 0 & 1 & 0 & 0 \\
0 & 0 & -1.5 & 0 & 1
\end{bmatrix}
\]

which is in observable canonical form as specified by Eqs. (4.89) to (4.91).
**Special Case.** For single-output systems, the nonsingular transformation matrix $P_0$ can also be constructed by

$$ P_0 = \tilde{R}_O^{-1} R_O $$  \hspace{1cm} (4.92) 

where $R_O$ is the observability matrix and $\tilde{R}_O^{-1}$ is given by

$$ \tilde{R}_O^{-1} = \begin{bmatrix} p_1 & p_2 & \cdots & p_{n-1} & 1 \\ p_2 & p_3 & \cdots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ p_{n-1} & 1 & 0 & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix} $$

with $p_i$ being the coefficient of the characteristic polynomial

$$ p(s) = s^n + p_{n-1}s^{n-1} + \cdots + p_1s + p_0. $$

**Observable Hessenberg form.** Let us directly present an algorithm which uses orthogonal coordinate transformation on the state Eqs. (4.66) and (4.67) to obtain an equivalent block observable Hessenberg form known also as staircase or condensed form. Singular value decomposition (SVD) is used and applied to the output matrix $C$, which is equivalent to carrying out coordinate transformation on the state and output vectors, that is, $\dot{y} = My$ and $\dot{x} = Tx$. Note that the matrices $M$ and $T$ are orthogonal, so that $M^{-1} = M^T$ and $T^{-1} = T^T$. The observability indices of the system can also be obtained in the process (Patel (1981)).

Thus, we show that a sequence of orthogonal transformations combined as an $n \times n$ singular matrix $P_h$, that is, $x_h = P_hx$, transforms the system into the block observable Hessenberg form

$$ \dot{x}_h(t) = A_h \, x_h(t) + B_h \, u(t) \hspace{1cm} (4.93) $$

$$ y(t) = C_h \, x_h(t) \hspace{1cm} (4.94) $$

where

$$ A_h = P_h A P_h^T = \begin{bmatrix} A_{11} & A_{12} & 0 & \cdots & 0 & 0 & \vdots \\ A_{21} & A_{22} & A_{23} & \cdots & 0 & 0 & \vdots \\ \vdots & \vdots & \ddots & \cdots & \vdots & \vdots & \vdots \\ A_{k-1,k} & A_{k-1,2} & A_{k-1,3} & \cdots & A_{k-1,k-1} & A_{k-1,k} & \vdots \\ A_{k,k} & A_{k,k-1} & A_{k,k-2} & \cdots & 0 & 0 & \vdots \\ n_1 & n_2 & n_3 & \cdots & n_{k-1} & n_k & \end{bmatrix} $$

$$ B_h = P_h B = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_{k-1} \\ B_k \end{bmatrix} $$

$$ C_h = C P_h^T = [C_1 \quad 0 \quad 0 \quad \cdots \quad 0 \quad 0] $$
Note that $D_h = D = 0$, if we consider the system Eqs. (4.66) and (4.67). Note also that in general we have

$$C_1 = \begin{bmatrix} \Sigma \\ 0 \end{bmatrix}$$

if $C$ is not assumed full rank. The observability indices $\{v_i, \quad i = 1, 2, \ldots, r\}$ of the pair $(A, C)$ or correspondingly $(A_h, C_h)$ can be derived from $\{n_j, \quad j = 1, 2, \ldots, k\}$. The recursive algorithm which shows the steps to be carried out for the construction of $P_h$ is outlined.

**Algorithm 4.9**

1. Set $i = 0$ and $v_1 = v_2 = \cdots = v_r = 0$, and let $l_0 = l$ be the numerical rank of $C$.

2. Find orthogonal matrices $M_0$ and $T_0$ by SVD such that

$$C T_0^T = M_0^T \begin{bmatrix} \Sigma_{l_0} & 0 \\ \hdashline 0 & 0 \end{bmatrix} \begin{bmatrix} l_0 \\ \hdashline \end{bmatrix}$$

and let $P_0 = T_0$.

3. Set $v_j = v_j + 1, \quad j = 1, 2, \ldots, l_0; \quad v = v_1 + v_2 + \cdots + v_{l_0}$ and construct

$$A_0 = P_0 A P_0^T = \begin{bmatrix} A_0^{11} & A_0^{12} \\ \hdashline A_0^{21} & A_0^{22} \end{bmatrix} \begin{bmatrix} l_0 \\ \hdashline \end{bmatrix}, \quad B_0 = P_0 B = \begin{bmatrix} B_0^1 \\ \hdashline B_0^2 \end{bmatrix} \begin{bmatrix} l_0 \\ \hdashline \end{bmatrix}$$

$$C_0 = C P_0^T$$

4. Set $i = i + 1$ and $C_i = A_{i-1}^{i+1}$ and find orthogonal matrices $M_i$ and $T_i$ by SVD such that

$$C_i T_i^T = M_i^T \begin{bmatrix} \Sigma_{l_i} & 0 \\ \hdashline 0 & 0 \end{bmatrix} \begin{bmatrix} l_i \\ \hdashline \end{bmatrix}$$

where $l_i$ is the numerical rank of $C_i$, determined from the number of its nonzero singular values.

5. Set $v_j = v_j + 1, \quad j = 1, 2, \ldots, l_i; \quad v = v_1 + v_2 + \cdots + v_{l_0}$ and let

$$A_i = P_i A_{i-1} P_i^T, \quad B_i = P_i B_{i-1}$$

$$C_i = C_{i-1} P_i^T = C_{i-1}$$

where
\[ P_i = \begin{bmatrix} I_v & 0 \\ 0 & T_i \end{bmatrix} \]

Then partition \( A_i \) and \( B_i \) as

\[ A_i = \{ A_i^q \}, \quad B_i = \{ B_i^q \}, \quad p = 1, \ldots, i + 2 \]
\[ q = 1, \ldots, i + 2 \]

where

\[ A_i^q \text{ and } B_i^q, \quad p = 1, \ldots, i + 1 \]
\[ q = 1, \ldots, i + 2 \]

are \( l_{q-1} \times l_{p-1} \) and \( l_{q-1} \times m \) matrices, respectively; \( A_i^{i+2p} \); \( p = 1, \ldots, i + 1 \) and \( B_i^{i+2} \) are \( (n - v) \times l_{p-1} \) and \( (n - v) \times m \) matrices, respectively; \( A_i^{q,i+2} \); \( q = 1, \ldots, i + 1 \) and \( (n - v) \times (n - v) \) matrices and \( A_i^{i+2,i+2} \) is an (n - v) \( \times \) (n - v) matrix.

6. Repeat steps 4 and 5 until \( v = n \).
7. Define \( P_h, A_h, B_h, C_h \) and stop. Note that \( P_h = P_k P_{k-1} \cdots P_1 P_0 \).

4.4.2 Reduced-Order State Observer Design

The full-order or the \( n \)-dimensional observer described in the previous subsection estimates all the state variables. However, the order of observer can actually be less than \( n \) because the observed output provides some linear combinations between the state variables specified by \( y(t) = C x(t) \), where \( C \) is assumed to be of full row rank \( r \). Therefore, it should be possible to reduce the dimension of the observer by the number of outputs of the system resulting in an \( n - r \) dimensional observer called reduced order state observer (Luenberger (1966, 1971)). To show this, traditionally it is common to apply a state coordinate transformation defined by

\[ \dot{x}(t) = P x(t) = \begin{bmatrix} C \\ R \end{bmatrix} x(t) \quad (4.95) \]

where \( R \) is any \( (n - r) \times n \) matrix which makes \( P \) nonsingular. The transformed system is then an output identifiable form written as

\[ \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \bar{B}_1 \\ \bar{B}_2 \end{bmatrix} u \quad (4.96) \]

\[ y = [I_r \quad 0] \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} \quad (4.97) \]

Although the transformation Eq. (4.95) seems to be very simple, from a computational point of view it is not recommended; instead, it is preferred to apply orthogonal transformation as used in the process of transforming a system to block Hessenberg
form. It should be pointed out that in this case it is sufficient to perform only steps 1, 2, and 3 of the transformation algorithm. Using $\hat{x}_1 = y$, we can write Eqs. (4.96) and (4.97) as

$$\dot{x}_2 = \tilde{A}_{22}x_2 + \tilde{B}u$$
$$\tilde{y} = \tilde{A}_{12}x_2$$

(4.98)

(4.99)

where $\tilde{B} = [\tilde{A}_{21} \tilde{B}_2]$, $\tilde{u} = \begin{bmatrix} y \\ u \end{bmatrix}$, and $\tilde{y} = \tilde{y} - \tilde{A}_{11}y - \tilde{B}_1u$. It is not difficult to show that the observability of the pair $\{A, C\}$ or equivalently $\{\tilde{A}, \tilde{C}\}$ implies and is implied by the observability of the pair $\{\tilde{A}_{22}, \tilde{A}_{12}\}$. Consequently, one can construct an $n - r$ dimensional observer for Eqs. (4.98) and (4.99) as

$$\dot{x}_{2e} = (\tilde{A}_{22} - \tilde{L}\tilde{A}_{12})x_{2e} + \tilde{L}\tilde{y} + \tilde{B}u$$

(4.100)

such that the eigenvalues of $\tilde{A}_{22} - \tilde{L}\tilde{A}_{12}$ are arbitrarily assigned by a proper choice of $\tilde{L}$. Note, however, that Eq. (4.100) involves differentiation of the output $y$ which is not desirable due to noise considerations. This can be eliminated by a variable transformation defined by

$$\tilde{z} = \dot{x}_{2e} - \tilde{L}y$$

(4.101)

which reduces Eq. (4.100) to

$$\dot{\tilde{z}} = (\tilde{A}_{22} - \tilde{L}\tilde{A}_{12})\tilde{z} + [(\tilde{A}_{22} - \tilde{L}\tilde{A}_{12})\tilde{L} +$$

$$(\tilde{A}_{21} - \tilde{L}\tilde{A}_{11})]y + (\tilde{B}_2 - \tilde{L}\tilde{B}_1)u$$

(4.102)

Having obtained $\tilde{L}$, $\tilde{z}$ is determined and we can find $x_{2e} = \tilde{z} + \tilde{L}y$. Finally, the estimate of the original state is given by

$$x_e = P^{-1}x_{2e} = P^{-1}\begin{bmatrix} y \\ \tilde{z} + \tilde{L}y \end{bmatrix} = P^{-1}\begin{bmatrix} I_r & 0 \\ \tilde{L} & I_{n-r} \end{bmatrix}\begin{bmatrix} y \\ \tilde{z} \end{bmatrix}$$

(4.103)

This derivation suggests that the general reduced order observer for the system Eqs. (4.96) and (4.97) be defined by

$$\dot{z}(t) = Fz(t) + Gy(t) + Hu(t)$$
$$x_e(t) = Mz(t) + Ny(t)$$

(4.104)

(4.105)

where $F$, $G$, and $H$ are, respectively, $(n - r) \times (n - r)$, $(n - r) \times r$, and $(n - r) \times m$ real constant matrices. The state $z(t)$ of the reduced-order observer is related to the state $x(t)$ of the system by

$$e(t) = z(t) - T x(t)$$

(4.106)

where $e(t)$ is the observer reconstruction error vector. Then the following theorem can be stated.
Theorem 4.4. The state of an observable system Eqs. (4.96) and (4.97) can be estimated with an \( n - r \) dimensional observer described by Eqs. (4.104) and (4.105) in the sense that \( z(t) - Tx(t) \rightarrow 0 \) as \( t \rightarrow \infty \) for any \( x(0), z(0), \) and \( u(t) \), if and only if the following conditions are satisfied

1. \( TA - FT = GC \) \hspace{1cm} (4.107)
2. \( MT + NC = I \) \hspace{1cm} (4.108)
3. \( H = TB \) \hspace{1cm} (4.109)
4. All the eigenvalues of \( F \) have negative real parts.

To prove this theorem, one should write the error dynamic as in the case of the full-order observer, which immediately deduce these conditions. It should also be pointed out that for the reduced-order observer, the relation between the actual state reconstruction error \( \hat{x} \) and the observer reconstruction error \( e \) is given by

\[
\hat{x} = Me + (MT + NC - I)x
\]

which by invoking Eq. (4.108) reduces to

\[
\hat{x} = Me
\]

The observer constraint Eq. (4.107) is responsible for the construction of the observer. Once the parameters \( T, F, \) and \( G \) are obtained from this equation, the parameters \( M, N \) are obtained from Eq. (4.108) as

\[
[M \ N] = \begin{bmatrix} T \\ C \end{bmatrix}^{-1}
\]

and \( H \) is obtained via Eq. (4.109).

Since the original contribution of Luenberger, observer design has been an active area of research. A tutorial study of observers for use in a wide variety of situations can be found in O’Reilly (1983). One should also consult the references therein. Arbel and Tse (1979), and Gupta and Fairman (1981) attempt to consider ways of proceeding with this design problem when the system order is large. Van Dooren (1984) provides a recursive algorithm which considers the numerical aspects of the design, and recent algorithm by Shafai (1988) goes along the same line, however, both numerical issues as well as large dimensionality of the system are considered.

In the sequel, two algorithms are provided for the design of reduced-order observers based on canonical forms discussed in Subsec. 4.3.1.

Algorithm 4.10. (Design Coordinate Required) This algorithm was originally proposed by Munro (1973) and it consists of the following steps.

1. Use the transformation matrix \( P_0 \) as specified by Eq. (4.86) to transform the system to observable canonical form Eqs. (4.87) and (4.88). If the desired form
is not constructed, the general scheme of Popov can be used to achieve this goal.

2. Apply a nonsingular transformation $S_O$ to the transformed output to remove the nonzero entries in $C_o$. Since the structure of the observer for the transformed system is

$$
\dot{z}_o = F_Oz_o + G_Oy + H_Ou
$$

The product $G_oC_o$ in the constraint equation

$$
T_oA_o - F_oT_o = G_oC_o
$$

can be replaced by $\tilde{G}_o\tilde{C}_o$ where

$$
\tilde{C}_o = S_o^{-1}C_o
$$

$$
\tilde{G}_o = G_oS_o
$$

Note that $\tilde{G}_o\tilde{C}_{oi} = [0 \tilde{g}_i]$, where $\tilde{g}_i$ is the $i$th column of $\tilde{G}_o$ for $i = 1, \ldots, r$.

3. Corresponding to the diagonal blocks $A_{ii}$ of $A_o$ with dimension $v_i \times v_i$, specify the blocks $F_{ii}$ of $F_o$ with dimension $(v_i - 1) \times (v_i - 1)$ having the desired eigenvalues as

$$
F_O = \text{block-diag}\left\{\begin{array}{cccc}
0 & 0 & \cdots & 0 \\
1 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 \\
\end{array}\right\} - \beta_{io} - \beta_{ii} - \beta_{i,n-1}, (v_i - 1) \times (v_i - 1), i = 1, \ldots, r
$$

4. Construct the matrix $T_o$ defined by

$$
T_o = \text{block diag} \{T_{ii} = [I_{v_i-1} \beta_i], (v_i - 1) \times v_i, i = 1, \ldots, r\}
$$

with $\beta_i$ representing the last column of $F_{ii}$.

5. Solve the observer constraint Eq. (4.114) for $\tilde{g}_i$ and specify $\tilde{G}_o$.

6. Determine $T = T_oP_o$ and compute the remaining observer parameters for the original system by Eqs. (4.116), (4.112), and (4.109).

**Example 4.11**

Consider a system $\dot{x} = Ax + Bu$, $y = Cx$, which is assumed to be in the observable canonical form $\dot{x}_o = A_o x_o + B_o u$, $y = C_o x_o$ with
\[ A_o = \begin{bmatrix} 0 & 2 & 0 & 4 \\ 1 & 1 & 0 & 1 \\ 0 & 3 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \quad B_o = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \]

\[ C_o = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \]

It is desired to design a reduced-order observer.

**Solution**

**Step 1.** In this case, \( P_o = I \) and \( n_1 = n_2 = 2 \).

**Step 2.** The nonsingular transformation matrix \( S_o \) applied to the output is given by

\[ S_o = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \]

so that from Eq. (4.115) we have

\[ \hat{C}_o = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

**Step 3.** Let the desired eigenvalues for the observer be \(-1, -2\). Then Eq. (4.117) implies that

\[ F_o = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix} \]

**Step 4.** The matrix \( T_o \) is constructed from Eq. (4.118) as

\[ T_o = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -2 \end{bmatrix} \]

**Step 5.** Solving the matrix Eq. (4.114) with \( G_oC_o = \hat{G}_o\hat{C}_o \) for the unknown columns \( \hat{g}_i \) of \( \hat{G}_o \), we get

\[ \hat{G}_o = \begin{bmatrix} 0 & 3 \\ 1 & -5 \end{bmatrix} \]

**Step 6.** It is obvious that \( T_o = T \) and Eqs. (4.116), (4.112), and (4.109) yield

\[ G = G_o = \hat{G}_oS_o^{-1} = \begin{bmatrix} -3 & 3 \\ 6 & -5 \end{bmatrix} \]

\[ [M \quad N] = \begin{bmatrix} T \quad C \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & -2 \\ 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \]

\[ H = TB = \begin{bmatrix} 1 & -1 \\ 1 & -2 \end{bmatrix} \]
**Algorithm 4.11.** (Design Coordinate Required) This algorithm is the dual version of the algorithm 4.6 which we recommended for large size problems. Here appropriate modifications are made for the design of a reduced-order observer (Shafai (1988)).

1. Transform the system to the observable Hessenberg form Eqs. (4.93) and (4.94) and let the observer for the transformed system be

\[ \dot{z}_h(t) = F_hz_h(t) + G_hy(t) + H_hu(t) \]  
\[ x_{he}(t) = M_hz_h(t) + N_hy(t) \]

where \( z_h(t) \) is an \( n - r \) dimensional vector.

2. Specify \( K \) and define the matrices \( T_h, F_h, \) and \( G_h \) as

\[
T_h = \begin{bmatrix}
T_{11} & I_{n_2} & & & \\
T_{21} & T_{22} & I_{n_3} & & \\
& & \ddots & \ddots & \\
& & & T_{k-2,1} & I_{n_{k-1}} \\
& & & T_{k-1,1} & T_{k-1,2} & T_{k-1,3} & \cdots & T_{k-1,k-1} & I_{n_k}
\end{bmatrix}
\]

(4.121)

\[
F_h = \begin{bmatrix}
F_1 & A_{23} \\
F_2 & A_{34} \\
& \ddots & \ddots \\
& & & \ddots & A_{k-1,k} \\
& & & & F_{k-1}
\end{bmatrix}
\]

(4.122)

\[
G_h = [G^T_1 \ G^T_2 \ G^T_3 \ \cdots \ G^T_{k-1}]^T
\]

(4.123)

where \( F_h \) implies that its eigenvalues are the union of the eigenvalues of each diagonal block.

3. Substitute Eqs. (4.121) to (4.123) and the transformed pair \( (A_h, C_h) \) from Eqs. (4.93) and (4.94) into

\[
T_hA_h - F_hT_h = G_hC_h
\]

(4.124)

We obtain the chain equation with similar but slightly different structure than Eq. (4.58) in Algorithm 4.6.

4. Obtain \( T_h \) based on approach 1 or \( T_h, F_h \) based on approach 2.

5. Compute \( G_h \) from the first chain equation.

6. Compute \( H_h = T_hB_h \) and \( M_h, N_h \) from

\[
[M_h N_h] = \left( T_h \right)^{-1}
\]

(4.125)

7. The estimate of the original state is given by

\[
x_e(t) = P_h^{-1} x_{he}(t)
\]

(4.126)
4.5 FEEDBACK CONTROL SYSTEMS USING OBSERVERS

The present section explores further the possibilities of linear feedback control law for systems with inaccessible state. In Sec. 4.3, the state feedback was used to assign the poles of the system. The control law was initially designed under the assumption that all of the states were available for feedback implementation. If the states are not accessible, then a full-order observer or a reduced-order observer as discussed in Subsec. 4.4.1 and 4.4.2 can be designed to estimate the states of the system so that the feedback implementation is realizable. This is possible if the system and the observer-based controller preserves the stability of the overall closed-loop system.

4.5.1 Observer-Based Controller Design

Let the \( n \)-dimensional system be defined by

\[
\begin{align*}
\dot{x}(t) &= A \, x(t) + B \, u(t) \quad (4.127) \\
y(t) &= C \, x(t) \quad (4.128)
\end{align*}
\]

where \( A, B, \) and \( C \) are \( n \times n, \ n \times m, \) and \( r \times n \) real constant matrices, respectively. First we design the control law under the availability of the state variables. Thus,

\[
u(t) = v(t) + K \, x(t) \quad (4.129)
\]

where \( k \) is an \( m \times n \) matrix has been applied to the system Eqs. (4.127) and (4.128) to obtain some desired behavior for the closed-loop system

\[
\begin{align*}
\dot{x}(t) &= (A + BK) \, x(t) + B \, v(t) \quad (4.130) \\
y(t) &= C \, x(t) \quad (4.131)
\end{align*}
\]

namely, to have a set of desired eigenvalues. Next, a stable observer of the form

\[
\begin{align*}
\dot{z}(t) &= F \, z(t) + G \, y(t) + H \, u(t) \quad (4.132) \\
xz(t) &= M \, z(t) + N \, y(t) \quad (4.133)
\end{align*}
\]

is designed under the observability assumption of the pair \( \{A, \ C\} \). Finally, the feedback gain \( K \) is applied to the estimated state \( xz \) by the control law

\[
u(t) = v(t) + K \, xz(t) \quad (4.134)
\]

which is shown in Fig. 4.5.

In this configuration, we considered the reduced-order observer. However, if we replace \( F, \ G, \ H, \ M, \) and \( N \) by \( A - LC, \ L, \ B, \ I, \) and \( O \) respectively, then \( z = xz \) and we have an identity observer-based controller configuration.

It is not difficult to write the composite dynamical equation of the system as
Figure 4.5  Block diagram of observer-based controller.

\[
\begin{bmatrix}
\dot{x} \\
\dot{z}
\end{bmatrix} = 
\begin{bmatrix}
A + BKNC & BKM \\
GC + HKNC & F + HKM
\end{bmatrix} 
\begin{bmatrix}
x \\
z
\end{bmatrix} + 
\begin{bmatrix}
B \\
H
\end{bmatrix} v
\]  
\hspace{1cm} (4.135)

\[
y = \begin{bmatrix}
C \\
0
\end{bmatrix} 
\begin{bmatrix}
x \\
z
\end{bmatrix}
\]  
\hspace{1cm} (4.136)

and by introducing an equivalent transformation of the form

\[
\begin{bmatrix}
x \\
e
\end{bmatrix} = 
\begin{bmatrix}
x \\
z - Tx
\end{bmatrix} = 
\begin{bmatrix}
I & 0 \\
-T & I
\end{bmatrix} 
\begin{bmatrix}
x \\
z
\end{bmatrix}
\]  
\hspace{1cm} (4.137)

Eqs. (4.135) and (4.136) reduce to

\[
\begin{bmatrix}
\dot{x} \\
\dot{e}
\end{bmatrix} = 
\begin{bmatrix}
A + BK & BKM \\
0 & F
\end{bmatrix} 
\begin{bmatrix}
x \\
e
\end{bmatrix} + 
\begin{bmatrix}
B \\
0
\end{bmatrix} v
\]  
\hspace{1cm} (4.138)

\[
y = \begin{bmatrix}
C \\
0
\end{bmatrix} 
\begin{bmatrix}
x \\
e
\end{bmatrix}
\]  
\hspace{1cm} (4.139)

It is obvious that the eigenvalues of the composite system are invariant under the equivalent transformation Eq. (4.137) and they are the union of those of \(A + BK\) and those of \(F\). This shows that the design of state feedback and the design of the observer can be performed independently without destroying the stability of the closed-loop system. This property is known as separation property.
It is interesting to note that the observer-based controller represented by Eqs. (4.132) and (4.133) may be viewed as a dynamic compensator (see Chap. 5) defined by

\[
\dot{z}(t) = F_1 z(t) + G_1 y(t) + H_1 v(t) \\
u(t) = K_1 z(t) + H_1 y(t) + v(t)
\]

(4.140) (4.141)

where

\[
F_1 = F + HKM \\
G_1 = G + HKN \\
K_1 = KM \\
H_1 = KN
\]

The order of the compensator in this case corresponds to the order of the observer \( l = n - r \).

A further reduction in the order is possible if one tries to generate some functions of the state, namely, \( u = Kx \). The structure of such an observer, known as a functional observer, is defined as

\[
\dot{z}(t) = F_1 z(t) + G_1 y(t) + H_1 u(t) \\
w(t) = M_1 z(t) + N_1 y(t)
\]

(4.142) (4.143)

where \( z(t) \) is assumed to have order \( l \) where \( 0 \leq l \leq n - r \). Again, it is not difficult to show that if

1. \( TA - FT = GC \)  
   (4.144)
2. \( MT + NC = K \)  
   (4.145)
3. \( H = TB \)  
   (4.146)
4. All the eigenvalues of \( F \) have negative real parts

then \( z(t) \) approaches \( T x(t) \) and \( w(t) \) approaches \( K x(t) \) as \( t \to \infty \).

One approach to solve this problem is to start with the lowest possible order, \( l = 0 \), and increase the order successively until all these conditions are satisfied. It turns out that for a single functional observer, the smallest order which guarantees the existence of a solution to the problem is given by \( v - 1 \), where \( v \) is the observability index of the system. However, if one seeks to obtain the observer of minimal order, then one should take the approach of successively increasing the order as pointed out above. See Prob. 4.28 and consult Fortmann and Williamson (1972), Murdoch (1973, 1974), Roman and Bullock (1975), Kimura (1978), Gupta et al. (1981), Fairman and Gupta (1980), Shafai and Carroll (1984) for further information. The functional observer problem may be considered as an equivalent problem of finding minimal-order dynamical compensator which stabilizes a given system.
Example 4.12
Consider the chemical reactor example discussed in Munro (1972).

\[
\begin{bmatrix}
1.3800 & -0.2077 & 6.7150 & -5.6760 \\
-0.5814 & -4.2900 & 0 & 0.6750 \\
1.0670 & 4.2730 & -6.654 & 5.8930 \\
0.0480 & 4.2730 & 1.3430 & -2.1040
\end{bmatrix}
\begin{bmatrix}
x(t)
\end{bmatrix}
\]

\[
B
+ \begin{bmatrix}
0 \\
5.679 \\
1.136 \\
1.136
\end{bmatrix}
\begin{bmatrix}
u(t)
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 \\
0 \\
0 \\
0 \\
1 \\
0
\end{bmatrix}
\begin{bmatrix}
x(t)
\end{bmatrix}
\]

It is desired to design an observer and a state feedback.

**Solution**  The eigenvalues of the system are

\[
\lambda_1 = -5.0566, \quad \lambda_2 = 1.991, \quad \lambda_3 = -8.6659, \quad \lambda_4 = 6.3508
\]

and indicate that the system is unstable as two of them are positive.

*Observer Design.*  Using the pole placement algorithm of the previous section with equivalent $A$ and $B$ matrices selected as

\[
A_{22} = \begin{bmatrix}
-6.654 & 1.343 \\
5.893 & -2.104
\end{bmatrix} \quad \text{and} \quad A_{12} = \begin{bmatrix}
6.715 & 0 \\
5.676 & 0.6750
\end{bmatrix}
\]

and arbitrarily selecting observer eigenvalues, say $\lambda_1 = -10, \lambda_2 = -12$ we obtain

\[
L^T = \begin{bmatrix}
0.4983 & 0.2 \\
15.7817 & 16.3425
\end{bmatrix}
\]

Knowing $L$, we can now calculate

\[
F = A_{22} - LA_{12} = \begin{bmatrix}
-10 & -1.9313 \\
0 & -12
\end{bmatrix}
\]

By inspection, the eigenvalues that were chosen for the observer have been attained. The remaining calculations are as follows to fully characterize the observer

\[
G = (A_{22} - LA_{12})L + (A_{21} - LA_{11}) = \begin{bmatrix}
4.1855 & -117.2993 \\
6.8735 & -121.6861
\end{bmatrix}
\]

\[
H = B_2 - LB_1 = \begin{bmatrix}
-88.4883 & -3.146 \\
-91.6731 & 0
\end{bmatrix}
\]
State Feedback. As shown, the original system is unstable. By adding state feedback, the eigenvalues can be moved to produce a stable closed-loop system. The design of the state feedback, as was shown in the previous paragraph, can be treated as a separate design to that of the observer. Arbitrarily again, choose some suitable eigenvalues as

\[ \lambda_1 = -5, \quad \lambda_2 = -2, \quad \lambda_3 = -1, \quad \lambda_4 = -9 \]

and by using the pole placement algorithm we obtain

\[
K = \begin{bmatrix}
0.40802 & -0.28198 & 0.17774 & -0.39672 \\
1.44130 & 0.28481 & 1.10672 & -0.56955
\end{bmatrix}
\]

Once \( K \) has been obtained all the terms of the modified \( A \) matrix of Eq. (4.135) can be calculated and gives

\[
A + BKNC = \begin{bmatrix}
1.3800 & -0.2077 & 6.7150 & -5.6760 \\
1.78814 & -26.7806 & 0 & 0.6750 \\
-4.3670 & -26.7872 & -6.6540 & 5.8930 \\
0.5220 & -0.2259 & 1.3430 & -2.104
\end{bmatrix}
\]

\[
BKM = \begin{bmatrix}
0 & 0 \\
-1.0093 & 2.2530 \\
3.2798 & -1.3411 \\
-0.2019 & 0.4507
\end{bmatrix}
\]

\[
(G + HKNC)C = \begin{bmatrix}
-38.6466 & 206.5830 & 0 & 0 \\
-31.3765 & 241.3702 & 0 & 0
\end{bmatrix}
\]

\[
F + HKM = \begin{bmatrix}
-29.2097 & 34.9656 \\
-16.2940 & 24.3686
\end{bmatrix}
\]

The closed-loop system as shown in Fig. 4.5 can now be simulated.

Results. Plots given in Fig. 4.6 show the simulation results for the estimated state versus the actual state for a unit step input on \( u_1(t) \) and \( u_2(t) = 0 \). The initial discrepancy between the estimated and actual states is due to the different initial conditions of the observer and the original system. If both systems had the same initial conditions, the observer would track from time zero. Note, that both states are increasing monotonically indicating an unstable system. Numerous results could be obtained for different input signals, for which the program can cater. Selection of the step response illustrates the transient characteristics well.
The second set of simulation results shown in Fig. 4.7 shows the estimated and true states of the observer-based state feedback system. Again the transient discrepancy between the states occurs due to the different initial conditions. The reference signal is zero, to which the system response eventually dies. Note in this case, the states are diminishing with time indicating that the system has been successfully stabilized.

If the initial conditions for the observer and original system are set to the same values, then the observer tracks the true states from time zero. In this case, there is no difference in response between the simple state feedback system and the complete observer, state feedback system.
4.5.2 Robustness with Observers

The term robustness refers to the preservation of desirable system properties, mainly stability and performance under structural and unstructural uncertainties. These uncertainties are caused by several reasons including model uncertainties, undesirable environmental effects such as induced parameter variations, noise, and similar types. We limit our discussion on stability robustness and highlight some important issues related to the design of observer-based controllers. A closed-loop system’s stability
property is deemed robust if it is maintained in the face of a specified class of model uncertainties; the larger the uncertainty class, the more robust the stability property.

Thus far, we learned that as a result of the separation principle, the realizations of control and observer are completely independent and that the overall observer-based controller system has all its $2n - r$ eigenvalues arbitrarily assigned in the left half of complex plane, preserving the stability. Before embarking on robustness analysis of observer-based controller, it is necessary to point out that in recent years many researchers directed their attention towards this issue. The exposition of the problem from a geometric point of view has been drawn by Bhattacharyya (1976) and with respect to measures for recovering full-state feedback regulator robustness has been drawn by Doyle (1978), Doyle and Stein (1979, 1981). A broader up-to-date design perspective on robust multivariable feedback control is to be found in the special issue of IEEE, Sain (1981), Bhattacharyya (1987), Dorato (1987), as well as Dorato and Yedavalli (1990). It is recalled that the closed-loop system under direct state feedback control and observer-based feedback control have the configurations shown in Figs. 4.8 and 4.9.
Note that the transfer function matrices $H_1(s)$ and $H_2(s)$ include our special case of $H_1(s) = I$, $H_2(s) = -K$ and further allows the inclusion of dynamic elements. It is easy to show that the overall transfer function matrices of both configurations are identical which asserts again the separation principle.

We continue our discussion by analyzing the effect of mismatches between the nominal plant and the actual plant. Suppose the parameters of the system are perturbed so that we have $A + \Delta A$, $B + \Delta B$, and $C + \Delta C$. Then it is not difficult to see that a new composite dynamical equation can be obtained instead of Eq. (4.138) in which the zero block is replaced by a nonzero block, say $\Delta X$. This indicates that in general, under perturbations, the separation principle is not valid anymore. Note that our discussion here is for a full-order observer-based controller and consequently, one should replace $F = A - LC$ and $M = I$ in Eq. (4.138). Now assume that in the state feedback controller of Fig. 4.8, the parameters are perturbed in such a way that their effect lies within the stability margins. These stability margins are known to be quite impressive in optimal linear quadratic regulator design. Thus, under the same circumstances, it is natural to inquire if such favorable robustness properties are preserved when observer is incorporated in the loop. Unfortunately, however, the two implementations show some differences even if there is no perturbation on the system parameters. To show this, one should recognize that the closed-loop transfer function matrices from the external input $v$ to the state $x$ are identical in both configurations (Figs. 4.8 and 4.9) because of the fact that the observer error dynamics are uncontrollable from the input $u'$ and therefore, unexcited by this input. Following the same reasoning, it is a simple exercise to verify that the return ratio matrix from signal $u'$ to signal $u$, loops broken at point XX, is also identical in both implementations. So far, identical closed-loop transfer function matrices and return difference matrices are reported for the state feedback and observer-based control configurations, promising equal robustness. However, if the loops are broken at point X, the return ratio matrices from the control signal $u''$ to the control signal $u'$ are, in general, different in the two configurations. Doyle and Stein (1979, 1981) showed that they are identical if the observer dynamic satisfies the condition

$$L[I + C(sl - A)^{-1}L]^{-1} = B[C(sl - A)^{-1}B]^{-1}$$

(4.147)

With reference to Fig. 4.8, the return-ratio matrix from $u''(s)$ to $u'(s)$ of the direct state feedback configuration is given by

$$u' = -H_1 H_2 \Phi B u''$$

(4.148)

where

$$\Phi(s) = (sl - A)^{-1}$$

(4.149)

is the system resolvent matrix. The return-ratio matrix from $u''(s)$ to $u'(s)$ of the observer-based controller configuration is derived as follows:

$$x_e = (\Phi^{-1} + LC)^{-1} [Bu' + LC \Phi B u'']$$

(4.150)

and since $u' = -H_1 H_2 x_e$, we obtain
\[ u' = -H_1 H_2 (\Phi^{-1} + LC)^{-1} [Bu' + LC \Phi B u''] \]  
(4.151)

and the application of Schur matrix inversion lemma to \((\Phi^{-1} + LC)^{-1}\) yields

\[ u' = -H_1 H_2 [\Phi - \Phi L (I + C \Phi L)^{-1} C \Phi] Bu' + LC \Phi B u'' \]  
= \(-H_1 H_2 \Phi [B - L (I + C \Phi L)^{-1} C \Phi B] u'\)  
\(-H_1 H_2 \Phi L (I + C \Phi L)^{-1} C \Phi B u''\)  
(4.152)

The reason for the difference between Eqs. (4.148) and (4.152) is that the observer error dynamics do get excited in response to input \(u''\) with loop broken at \(X\). However, if Eq. (4.147) is satisfied, the first term on the right hand side of Eq. (4.152) vanishes and the second term reduces to Eq. (4.148). Consequently, the return-ratio matrices at point \(X\) will be identical in both configurations resulting in equal robustness properties. According to Rynaski (1982), observers that satisfy the Doyle and Stein condition are called “Robust Observers.” To be more precise, since such an observer is used in connection with feedback control, we call the overall design as “Robust Observer Based Controller Design.”

To conclude our discussion on the robust observer, it remains to provide a method for selecting observer gain \(L\) such that Eq. (4.147) is fulfilled. The idea of high-gain output feedback control law (O’Reilly (1983)) suggest that an observer gain \(L(q)\) should be selected based on

\[ \frac{L(q)}{q} \rightarrow BW \]  
(4.153)

for any nonsingular matrix \(W\). This selection guarantees that the Eq. (4.147) will be satisfied asymptotically as \(q \rightarrow \infty\). The limiting poles of the full-order observer are the roots of the characteristic polynomial.

\[ \psi(s) = \det(sI - A) \det[I_r + qC(sI - A)^{-1} BW] \]  
(4.154)

which is the same as the determinant of the return difference matrix of the full-order observer, that is, \(\det[I_r + C(sI - A)^{-1} L]\). As \(q \rightarrow \infty\), \(r\) of these roots will approach the transmission or invariant zeros of

\[ p(s) = \det(sI - A) \det[C(sI - A)^{-1} B] = \det \begin{pmatrix} sI - A & B \\ C & 0 \end{pmatrix} \]  
(4.155)

and the remaining \(n - r\) observer poles tend to the infinite zeros of the system. Thus, the plant must be minimum phase in order to guarantee the stability of the observer. It is again clear that making all the roots of the observer error dynamics arbitrarily fast is not a correct task to perform from the point of view of robustness. Doyle and Stein (1981) suggest also a robust recovery procedure by selecting the following Kalman-type filter gain.

\[ L(q) = P(q) C^T R^{-1} \]  
(4.156)

where \(P(q)\) satisfies the algebraic Riccati equation\(^6\)

\(^6\) For a detail discussion on Riccati equations, see Chapter 6.
\[ A P + PA' + Q(q) - P C'R'C P = 0 \]  \hspace{1cm} (4.157)

for which

\[ Q(q) = Q_o + q^2 BVB' \]  \hspace{1cm} (4.158)

\[ R = R_o \]  \hspace{1cm} (4.159)

where \( Q_o \) and \( R_o \) are noise covariance matrices for the nominal system.

Finally, we should point out that the robustness based on reduced-order observer can equally be analyzed. In this case, overall robustness properties may be deduced in terms of the gain matrix \( \tilde{L} \) and the observer coefficient matrix \( \tilde{A}_{22} - \tilde{L}A_{12} \) defined by Eq. (4.100). This has been further discussed by Madiwale and Williams (1985) and Friedland (1986). Simply stated, the problem is the signal transfer from the control \( u \) to the observer through the control distribution matrix \( B \). To eliminate this feedback of the control matrix for the case of reduced-order observer, the matrix \( H = TB \) in Eq. (4.104) must be zero. Equivalently, by Eq. (4.102) \( \tilde{L} \) must be selected in order to satisfy \( \tilde{B}_2 - \tilde{L}B_1 = 0 \). Thus, for square system \( (r = m) \) and under the nonsingularity assumption of \( B_1 \), we have \( \tilde{L} = \tilde{B}_2B_1^{-1} \). Note that \( \tilde{C}_1 \) in Eq. (4.97) is \( I \) and the nonsingularity assumption is equivalent to \( \det \tilde{C}_1B_1 \neq 0 \) or \( \det CB \neq 0 \). It can be shown that with this selection of \( \tilde{L} \), the poles of the robust observer coincide with transmission zeros of the plant. Hence, it is again imperative to make the assumption of minimum phase for the plant. The more complex problem of the loop transfer recovery for nonminimum phase systems is also investigated by many researchers. We conclude by pointing out that the results of this chapter can equally be used for discrete-time systems (see Prob. 4.32).

**PROBLEMS**

4.1 Consider the system

\[
\begin{bmatrix}
    1 & 1 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & -1 & 1 \\
    0 & 0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
    x_1 \\
    x_2 \\
    x_3 \\
    x_4
\end{bmatrix}
+ 
\begin{bmatrix}
    0 \\
    1 \\
    1 \\
    0
\end{bmatrix} u
\]

\[ y = [1 \\ 1 \\ 1 \\ 1] x + [1] u \]

a. Is it possible to find a state feedback control law \( u = v + kx \) such that all the eigenvalues of the system are placed at \(-1\)? Is it possible to place them at \(-2\)?
b. If the answer to part a is yes, find the corresponding feedback gain.
c. Find the simplest possible feedback control law such that the system is unobservable.

4.2 The dynamical equation of a system is given by

\[
\begin{bmatrix}
    -1 & 0 \\
    1 & 1
\end{bmatrix}
\begin{bmatrix}
    x_1 \\
    x_2
\end{bmatrix}
+ 
\begin{bmatrix}
    0 \\
    1
\end{bmatrix} u
\]

\[ y = [1 \\ 1] x \]
Is it possible to stabilize the system by state feedback? If yes, find the required feedback gain $k$ such that all the eigenvalues are placed at $-1$.

4.3 Prove that the state feedback dynamical Eqs. (4.4) and (4.5) are controllable for any feedback gain $K$, if and only if the dynamical Eqs. (4.1) and (4.2) are controllable or

$$\rho(BAB \cdots A^{n-1}B) = \rho(B(A + BK)B \cdots (A + BK)^{n-1}B).$$

4.4 The simplified equations of motion for a helicopter can be described by

$$\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} = \begin{bmatrix}
-0.02 & -1.4 & 9.8 \\
-0.01 & -0.4 & 0 \\
0 & 1 & 0
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} + \begin{bmatrix}
9.8 \\
6.3 \\
0
\end{bmatrix} u$$

where $x_1$, $x_2$, $x_3$, and $u$ represent horizontal velocity, pitch rate, pitch angle, and rotor tilt angle.

a. Find the open-loop poles of the system.
b. Show that a state feedback control law $u = v + kx$ with

$$k = \begin{bmatrix}
-0.0627 \\
-0.4706 \\
-0.9999
\end{bmatrix}$$

shift the poles to $-2, -1 \pm j$.

4.5 Consider the inverted pendulum problem discussed in Ex. 2.1 and let $mg/M = 1$, $g/l = 5$, $1/M = 1$, and $1/2Ml = 2$. Then the state equation becomes

$$\dot{x} = \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 5 & 0
\end{bmatrix} x + \begin{bmatrix}
0 \\
1 \\
0 \\
-2
\end{bmatrix} u$$

$$y = [1 \ 0 \ 0 \ 0]x$$

Find the feedback control law $u = v + kx$ such that the closed-loop system has eigenvalues $-1, -2, -1 \pm j$.

4.6 The dynamic equations for the longitudinal motion of an aircraft is given by Rynaski (1982) as

$$\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\dot{x}_4
\end{bmatrix} = \begin{bmatrix}
-0.0507 & -3.861 & 0 & -32.17 \\
-0.00117 & -0.5164 & 1 & 0 \\
-0.000129 & 1.4168 & -0.4932 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix} x + \begin{bmatrix}
0 \\
-0.0717 \\
-0.1645 \\
0
\end{bmatrix} u$$

$$y = [0 \ 0 \ 1 \ 0]x$$

where the state vector $x = [x_1 \ x_2 \ x_3 \ x_4]^T$ consist of speed change, angle of attack, pitch rate, pitch, and the control $u$ represent elevator deflection. The system has an unstable pole at 0.724. It is required to stabilize the system such that the poles of the system are placed at $-1.25 \pm 2.1651j, -0.01 \pm 0.0995j$. Find the feedback gain $k$ of the control law by applying any of the single-input pole placement algorithms. (Use your favorite CAD package.)

4.7 Consider the system
\[
\dot{x} = \begin{bmatrix}
-2 & 1 & 2 \\
-1 & -2 & 2 \\
-2 & 0 & 2
\end{bmatrix} x + \begin{bmatrix}
0 \\
1 \\
0
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix} x
\]

Use dyadic method and your favorite package to transfer all poles of the system at \(-2\).

4.8 Let the state equation of a system be given by

\[
\dot{x} = \begin{bmatrix}
1 & 2 & 3 \\
2 & 3 & 4 \\
3 & 4 & 5
\end{bmatrix} x + \begin{bmatrix}
1 \\
0 \\
0
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix}
\]

Use Algorithm 4.5 to place the poles of the system at \(-4, -4, -4\).

4.9 The state equation of a system is given by

\[
\dot{x} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
1 & 2 & 0 & 0 & 0 \\
2 & 1 & 3 & 0 & 0 \\
1 & 0 & 0 & 4 & 0 \\
0 & 1 & 0 & 1 & 5
\end{bmatrix} x + \begin{bmatrix}
0 \\
1 \\
0 \\
0 \\
0
\end{bmatrix} u
\]

a. Place the eigenvalues of the system at \(-1, -2, -3, -4, -5\) by using Algorithm 4.5.

b. Change the matrix \(B\) to

\[
\begin{bmatrix}
1 & 0 & 1 & 0 & -2 \\
-1 & 1 & 0 & 0
\end{bmatrix}^T
\]

and repeat the same as part a. Identify the difference in procedure.

4.10 Develop a simplified version of the Algorithm 4.6 for single-input systems.
a. Write a program for the simplified algorithm.
b. Consider the \(20 \times 20\) bidiagonal matrix of Wilkinson which has ill-conditioned eigenvalues.

\[
A = \begin{bmatrix}
20 & & & & \\
20 & 19 & & & \\
20 & 18 & & & \\
& & & & \\
& & & & \\
& & & & \\
& & & & \\
& & & & \\
& & & & \\
& & & & \\
& & & & 20 & 1
\end{bmatrix}
\]

Pass the matrix without its first row through the Algorithm 4.6 to assign the eigenvalues of the original matrix \(A\). Compute the resulting first row and compare the eigenvalues of the assigned matrix with the ones of \(A\).
c. Repeat part b for the Frank matrices given below.
\[ A_1 = \begin{bmatrix} 4 & 3 & 2 & 1 \\ 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 12 & 11 & 10 & 9 & \cdots \\ 11 & 11 & 10 & 9 & \cdots \\ 10 & 10 & 10 & 9 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & 2 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \]

4.11 Consider the chemical reactor example of Munro (1972)

\[
\dot{x} = \begin{bmatrix} 1.38 & -0.2077 & 6.715 & -5.676 \\ -0.5814 & -4.29 & 0 & 0.675 \\ 1.067 & 4.273 & -6.654 & 5.893 \\ 0.048 & 4.273 & 1.343 & -2.104 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1.136 \\ -3.146 \end{bmatrix} u
\]

\[ y = \begin{bmatrix} 1 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \]

Use Algorithm 4.6 and your favorite package to stabilize the system with eigenvalues \(-0.2, \ -0.5, \ -5.0566, \ -8.6659\).

4.12 Consider the distillation column example of Klein and Moore (1982) which was displayed in CAD-Example 4.1. Use Algorithm 4.10 and your favorite package to stabilize the system with eigenvalues \(-0.2, \ -0.5, \ -1, \ -1 \pm j\).

4.13 Let the dynamical equation of a system be given in a multivariable controllable canonical form as

\[
\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} u
\]

\[ y = [1 \ -1 \ 3 \ 2 \ 0] x \]

Two different feedback gain matrices which cause the matrix \((A + BK)\) to have eigenvalues \(-1, \ -2 \pm j, \ -1 \pm 2j\) are found to be

\[ K_1 = \begin{bmatrix} -2 & 0 & 0 & 0 \\ -25 & -55 & -48 & -23 \\ -1 \end{bmatrix} \]

and

\[ K_2 = \begin{bmatrix} -7 & -9 & -5 & -1 & -1 \\ 0 & 0 & 0 & -4 & 0 \end{bmatrix} \]

The feedback gain \(K_2\) preserves the original block structure of companion form and its elements are smaller than the elements of \(K_1\).

a. Use your favorite package such as TIMDOM, CAD-MCS, CONTROL.lab, and so on, to obtain the zero input responses of the closed-loop system corresponding to \(K_1\) and \(K_2\). Compare your result and draw conclusions.

b. Use the Algorithm 4.6 and obtain a third feedback gain \(K_3\) such that \(A + BK_3\) has
the desired eigenvalues. Compare the feedback gain and the response of the closed-loop system with the ones obtained in part a.

4.14 Find the equivalence to transformation \( \dot{x} = Px \) such that the system \( \dot{x} = Ax + Bu, \ y = Cx \) is transformed to

\[
\dot{x} = \begin{bmatrix}
\bar{A}_{11} & \bar{A}_{12} \\
\bar{A}_{21} & \bar{A}_{22}
\end{bmatrix} x + \begin{bmatrix}
1_m \\
0
\end{bmatrix} u \\
y = [\bar{C}_1 \bar{C}_2] \dot{x}
\]

Assuming that all eigenvalues of \( \bar{A}_{22} \) are in the left half of complex plane, find the feedback control law which stabilizes the system and block diagonalize the closed-loop system matrix.

4.15 Let the system \( \dot{x} = Ax + Bu, \ y = Cx \) be stabilizable. Show that if in \( j \)th step of the Hessenberg transformation algorithm the submatrix \( A_{i,j-1} = 0 \), then the pole assignment can be completed by applying the Algorithm 4.6 to a submatrix of \( A_k \) with the size \( n_1 + n_2 + \cdots + n_{j-1} \). Investigate the case when \( j = 2 \). (Hint: See Prob. 4.14.)

4.16 Use Algorithm 4.7 to assign the given set of desired eigenvalues for the following systems.

a. \( \dot{x} = \begin{bmatrix}
-1 & 0 & 0 \\
1 & -2 & 0 \\
-1 & 0 & -4
\end{bmatrix} x + \begin{bmatrix}
0 & 0 & 1 \\
-1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix} u \quad \lambda(A + BK) = \{-4, -5, -6\}

b. \( \dot{x} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & -2 & 1
\end{bmatrix} x + \begin{bmatrix}
0 & 1 \\
0 & 0 \\
1 & 0
\end{bmatrix} u \quad \lambda(A + BK) = \{-2, -3, -4\}

c. \( \dot{x} = \begin{bmatrix}
-2 & 1 & 0 & 0 \\
-2 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} x + \begin{bmatrix}
0 & 0 \\
0 & 0 \\
1 & 1 \\
0 & 1
\end{bmatrix} u \quad \lambda(A + BK) = \{-5 \pm 2j, -6, -10\}

4.17 For the system of part a in Prob. 4.16, compute the zero input response with \( x(0) = [1 \ 0 \ 1]^T \). Use your favorite CAD package such as TIMDOM, CAD-MCS, CONTROL. Lab, and so on, to plot the response and compare it with the response of assigned system.

4.18 For the system of part b in Prob. 4.16 it is desired that the mode \( e^{-2t} \) not appear in \( x_1 \) and that the mode \( e^{-3t} \) not appear in \( x_1 \) and \( x_3 \). Select the eigenvectors that produce this result.

4.19 In Subsec. 4.4.2., we showed that the equivalence transformation defined by Eq. (4.95) transforms the system Eqs. (4.66) and (4.67) to Eqs. (4.96) and (4.97). Prove that the observability of the pair \( \{A, C\} \) or equivalently \( \{\bar{A}, \bar{C}\} \) implies and is implied by the pair \( \{\bar{A}_{22}, \bar{A}_{12}\} \).

4.20 Consider the system described by

\[
\dot{x} = \begin{bmatrix}
0 & 1 & -22 & -23 \\
0 & 0 & -11 & -6 \\
0 & 0 & -4 & 0 \\
1 & -2 & 0 & -6
\end{bmatrix} x + \begin{bmatrix}
0 & 0 \\
0 & 0 \\
0 & 1 \\
1 & 0
\end{bmatrix} u
\]
y = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 3 \end{bmatrix} x

a. Design a full-order observer for this system with eigenvalues at \(-5, -6, -7, -8\).
b. Design a reduced-order observer based on Algorithm 4.10 for this system with both eigenvalues at \(-6\).

4.21 Let a system be represented by its block observable Hessenberg form as Eqs. 4.93 and 4.94. Define the block structure matrices \(T_h, F_h,\) and \(G_h\) so that the constraint equation \(T_hA_h - F_hT_h = G_hC_h\) of the full-order observer is decomposed as in the dual case of pole assignment Algorithm 4.6.

4.22 For the system

\[
\begin{bmatrix}
-1.5 & 0.1 & 0 & 0.6 & 0.4 \\
-0.3 & 0 & 0.2 & -0.1 & 0.2 \\
-0.25 & 0 & -1 & -0.25 & 1 \\
0.1 & -0.2 & 0 & -2 & 0.5 \\
0.5 & 0 & 0.4 & 0 & 0.1
\end{bmatrix} x + \begin{bmatrix} 0 \\
1 \\
0 \\
0 \\
0 
\end{bmatrix} u
\]

\[
y = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} x
\]

design a reduced-order observer based on Algorithm 4.11.

4.23 Consider the problem of balancing a stick on your hand. There are several mechanical problems that have the character of complex balancing problems (Luenberger 1979 and Kailath 1980). Let the length of the stick be \(l\) and all of its mass \(m\) be concentrated at the top as shown in Fig. P4.23.

a. Using the obvious relations \(x(t) = p(t) + l \sin \theta(t)\), and the force component acting in the \(x\) direction \(mg \sin \theta(t) = F_x(t) = m\ddot{x}(t)\) with the assumption of small \(\theta\), derive the state equation of the system such that \(\theta = x_1, \theta = x_2\) represent the state variables, the input \(u(t) = \dot{\theta}(t)\) is the acceleration of the fingertip in the \(x\) direction and the output is chosen as \(y(t) = \theta(t)\).

b. Show that the system is controllable but unstable.

c. Use state feedback to stabilize the system with both desired eigenvalues at \(-1\).

d. Let us suppose that only position (stick’s angle \(\theta\)) can be measured. Construct a reduced-order observer with eigenvalue \(-2\) to produce an approximation to the velocity.

4.24 Let \(S_1\) be a linear time invariant system with input \(u(t)\) and output \(y(t)\). An observer \(S_2\)

![Figure P4.23: A broom balancer problem.](image)
is connected appropriately to $S_1$. Show that the resulting composite system is always uncontrollable from the input $u(t)$.

4.25 Show that the transfer function matrix of a closed-loop system incorporating a state observer is identical to the transfer function matrix of the closed-loop system employing direct state feedback.

4.26 Consider a system with the transfer function

$$g(s) = \frac{(s + 1)}{(s - 2)(s + 3)}$$

Design a reduced-order observer-based controller for this system such that overall closed-loop system has all its eigenvalues at $-3$.

4.27 Consider the system

$$\dot{x} = \begin{bmatrix} -1 & -2 & -2 \\ 0 & -1 & 1 \\ 1 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix} u$$

$$y = [1 \ 1 \ 0]x$$

Design a reduced-order observer-based controller for the system such that the eigenvalues of the observer are at $-2$ and $-3$ and the eigenvalues of the system are shifted to $-1$, $-2$, and $-3$. What is the allowable perturbation on $a_{11}$ element before the closed-loop system becomes unstable?

4.28 Consider the dynamical equation

$$\dot{x} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 2 \\ 1 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} x$$

Use your favorite CAD package to find a full-order observer, a reduced order observer, a functional observer for $kx$ with $k = [0 \ 1 \ 1 \ 2 \ -1]$. (Hint: For the last part of this problem consult reference Gupta et al. (1981).)

4.29 Consider again the system of Prob. 4.6. According to Rynaski (1982) a robust observer can be designed, if the observer poles are selected as $0$, $-0.0422$, $-1.0007$, and $-0.5860$. Consult related references and use your favorite package for convenience to design different robust observer-based controller for the system such that the stabilized system has the eigenvalues specified in Prob. 4.6. Display the zero input responses and compare the results.

4.30 Show that if the Doyle and Stein condition Eq. 4.147 is satisfied, then the transfer function from $u$ to $x$ in full state feedback configuration will be the same as the transfer function from $u$ to $\dot{x}$ in observer-based feedback system, promising equal robustness.

4.31 Show that the robust reduced-order observer is specified by
\[ \dot{z} = Fz + G y \]
\[ \dot{x} = Mz + Ny \]

with
\[
\begin{bmatrix} M & N \end{bmatrix} = \begin{bmatrix} T & C \end{bmatrix}^{-1}
\]
\[
\begin{bmatrix} F & G \end{bmatrix} = \begin{bmatrix} TAM & TAN \end{bmatrix}
\]

and \( T \) is obtained from the solution of algebraic equation \( B^T T = 0 \) using singular value decomposition of \( B^T \). Specify \( T \) in terms of right singular vectors (consult reference Shafai et. al. 1990).

Use the above method or the method discussed in subsection 4.5.2 to design the robust reduced-order observer for the system
\[
\dot{x} = \begin{bmatrix} 0 & 1 \\ -3 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u
\]
\[ y = [2 \ 1] x \]

so that the regulator specified by the control law \( u = -[50 \ 10] x + [50] v \) is recovered.

4.32 Consider the linear time-invariant discrete-time system
\[
x(k+1) = \begin{bmatrix} 1 & 0.0787 \\ 0 & 0.6065 \end{bmatrix} x(k) + \begin{bmatrix} 0.0043 \\ 0.0787 \end{bmatrix} u(k)
\]
\[ y(k) = [1 \ 0] x(k) \]

a. Find a control law \( u = v + kx \) so that all the eigenvalues of the resulting feedback system are at 0.
b. Show that, for any initial state, the zero-input response of the feedback system becomes identically zero for \( k \geq 2 \).c. Show that for a general \( n \)th order discrete system the zero-input response of the feedback system, with all eigenvalues placed at zero, becomes identically zero for \( k \geq n \). This implies that any initial state is driven to zero in (at most) \( n \) steps. The associated feedback control law is therefore called a dead-beat control law.
d. Design a dead-beat observer for this system and obtain the response of the observer-based control system.
5

Output Feedback and Compensator Design

5.1 INTRODUCTION

In the previous chapter, it was shown that for a controllable system the state feedback control law can shift the eigenvalues of the system to desired locations. When the state variables are not available for feedback, a state feedback law can be implemented through the use of observers. An alternative way for the design of feedback control systems, perhaps more desirable in certain situations, is to implement a feedback control law using available outputs rather than the complete states of the system. Thus, in this chapter, the design based on output feedback is considered. Methods for assigning the poles of the system via static and dynamic output feedback are presented. In the design of static output feedback, the access to the state variables of the system under control is not required and no additional dynamics is incorporated to the system. Although this is an advantage, it has the limitation that, in general, only a limited number of poles of the system may be arbitrarily specified. To overcome this difficulty and to achieve design requirements in addition to pole placement, dynamic output feedback can give extra freedom to the designer. Consequently, Secs. 5.3 and 5.4 are devoted to the design of dynamic output feedback using two- and three-term controllers as well as general compensators with desirable properties such as asymptotic tracking and disturbance rejection. In the case of output feedback, an observer is not explicitly required, however, an observer is often implicit in the design algorithms. Although it is more practical to use output feedback, the associated design techniques are often more involved. In Sec. 5.4, it is also shown that the feedback
compensator of multivariable systems may be reduced to an equivalent constant output feedback design considered in earlier sections.

5.2 STATIC OUTPUT FEEDBACK

This section is devoted to the design of static or proportional output feedback for the system

\[ \dot{x}(t) = A \, x(t) + B \, u(t) \]  
\[ y(t) = C \, x(t) + D \, u(t) \]

If the control law of the form

\[ u(t) = v(t) + K \, y(t) \]

is applied to the system Eqs. (5.1) and (5.2), the resulting output feedback system becomes

\[ \dot{x}(t) = (A + BK) \, x(t) + Bv(t) \]  
\[ y(t) = (C + DK) \, x(t) + Dv(t) \]

We would like to derive conditions under which the closed-loop system is stabilized through appropriate choice of $K$, where $K$ is an $m \times r$ matrix. Figure 5.1 shows the output feedback configuration.

It is obvious that what we can achieve with state feedback cannot be achieved by output feedback; otherwise, there would be no need for an observer to be introduced. Comparing Eqs. (5.4) and (5.5) with Eqs. (4.4) and (4.5) shows that the eigenvalues obtained by state feedback are identical to those by output feedback if

\[ K_s = K_0 C \]  

where we inserted the subscripts to distinguish $K_s$ and $K_0$ as the state and output feedback gains, respectively. For a given $K_s$ in Eq. (5.6), $K_s^T$ must lie in the range

![Figure 5.1 Output feedback system.](image-url)
space of $C^T$ in order to have a solution for $K_0$. (See Patel (1974), Porter (1977) and Sinha (1984) for more details). This is a restrictive condition and it leads to the limitation that not all poles of the system can be arbitrarily assigned in general.

Consequently, it is natural to investigate the number of assignable poles in both static and dynamic output feedback problems. Davison (1970) and Brasch and Pearson (1970) were among the first to investigate this problem. Using static output feedback, Davison (1970) has shown that max $(m, r)$ poles can be assigned arbitrarily close to desired values. For the case of dynamic output feedback, it has been shown by Brasch and Pearson (1970) that a dynamic compensator of degree $l = \min \{\mu_c - 1, \nu_o - 1\}$ is sufficient to allow arbitrary pole assignment. Here, $\mu_c$ and $\nu_o$ are the controllability and observability indices, respectively (see Chapter 4). This was an interesting result since it was also known that a functional observer of order $l = \nu_o - 1$ can stabilize a single-input system (Luenberger (1971)). It is clear that for a multivariable system, a dynamic compensator of degree $l = m(\nu_o - 1)$ can achieve the goal (Kailath (1980), O’Reilly (1983)). The first important results on static output feedback pole assignment were given by Kimura (1975) and Davison and Wang (1975). It was shown in these references that a sufficient condition for arbitrary pole assignment using static output feedback is that $(m + r - 1) \geq n$. Alternative proof of this result and methods to compute the feedback matrix were provided in Topalgu and Seborg (1975) and Seraji (1978). In particular, Seraji (1978) and Munro and Novin-Hirbod (1979) provide methods to design a full rank output feedback $K$ from a minimal sequence of dyads.\footnote{The word dyad was introduced in Sec. 4.3.3 with respect to state feedback gain matrix. Similarly, one can define it with respect to static and dynamic output feedback compensators.} Further, they show that when $m + r - 1 < n$, then a full rank dynamic compensator of degree $l \geq [n - (m + r - 1)]/\max\{m, r\}$ can be constructed from a minimal sequence of dyads, which will allow arbitrary pole assignment. Since then, the basic question of existence and computation of output feedback matrix for pole assignment for a given system has been considered by many researchers (e.g., Tarokh (1980), Brockett and Byrnes (1981), Kabamba and Longman (1982), Byrnes and Anderson (1984), Giannakopoulos and Karcanias (1985)) from the point of view of analytical and numerical improvements.

In this section, we present two algorithms for pole assignment with static output feedback. The first algorithm is direct (Seraji (1978)) and the second algorithm is iterative (Tarokh (1987)). As it is usually the case, the iterative methods give the possibility to stop the calculations at any point and examine the current solution for acceptable accuracy. This can not be done with direct methods. Also, roundoff errors do not accumulate since each approximation depends only on the previous one. In spite of these attractive features, iterative algorithms are known to be computationally intensive. One drawback is that the solutions produced by the iterative methods are strongly dependent on the initial choice. A bad initial choice will cause the number of iterations to increase which require more computation and consequently more time is needed to complete the solution.
5.2.1 A Direct Method for Pole Assignment by Output Feedback

It is always possible to assign \( \max(m, r) \) poles in a controllable and observable system by static output feedback. This is done by using a unity rank feedback matrix \( K = qk \) in the control law of Eqn. (5.3), where \( q \) and \( k \) are \( m \times 1 \) and \( 1 \times r \) gain vectors, respectively. On choosing the vector \( q \) such that the equivalent single input system \((A, Bq, C)\) is controllable, the \( r \) elements of \( k \) can be found to assign \( r \) closed-loop poles at desired locations. Note that a randomly selected \( q \) makes \((A, Bq, C)\) controllable (Reader may show this as an exercise). Alternatively, choosing a random \( k \), one can make the equivalent single output system \((A, B, kC)\) observable. The \( m \times 1 \) vector \( q \) is now chosen to assign \( m \) closed-loop poles.

In order to increase the number of assignable poles, in this section we describe an improved method to assign \((m + r - 1)\) poles of the closed-loop system using static output feedback. The gain matrix \( K \) in control law of Eq. (5.3) is constructed as the sum of two unity rank matrices \( K_1 \) and \( K_2 \) through the following three-step algorithm. For simplicity, we assume that \( D = 0 \) which reduces Eq. (5.5) to \( y = Cx \).

Algorithm 5.1

1. In this step, the \( m \times r \) unity rank feedback matrix \( K_1 = q_1k_1 \), where \( q_1 \) and \( k_1 \) are \( m \times 1 \) and \( 1 \times r \) vectors, respectively, is obtained such that \( m - 1 \) poles of the system are placed at \( \lambda_1, \ldots, \lambda_{m-1} \). The closed loop characteristic polynomial \( p_1(s) = \det(sI - A - BK_1C) \), using determinent identity \( \det(I - ab) = 1 - ba \) with \( a \) and \( b \) being \( n \times 1 \) and \( 1 \times n \) vectors, respectively, can be expressed as

\[
 p_1(s) = p_0(s) - k_1W_0(s)q_1 \tag{5.7}
\]

where \( W_0(s) = C \text{ adj}(sI - A)B \) and \( p_0(s) = \det(sI - A) \). The vector \( k_1 \) and one element of the vector \( q_1 \) are specified arbitrarily subject to the condition that \( \{A, k_1C\} \) is observable and the remaining \( m - 1 \) elements of \( q_1 \) are found by solving the set of \( m - 1 \) linear equations

\[
 p_1(\lambda_1) = p_0(\lambda_1) - k_1W_0(\lambda_1)q_1 = 0 \\
 \vdots \\
 p_1(\lambda_{m-1}) = p_0(\lambda_{m-1}) - k_1W_0(\lambda_{m-1})q_1 = 0 \tag{5.8}
\]

The unity rank output feedback matrix which places \( m - 1 \) poles of the system is then obtained as \( K_1 = q_1k_1 \).

2. In this step, we determine the \( m \times r \) unity rank output feedback matrix \( K_2 = q_2k_2 \) for the system \( \{A + BK_1C, B, C\} \) which preserves the \( m - 1 \) poles obtained in the first stage and places \( r \) additional poles at the specified locations \( \lambda_m, \ldots, \lambda_{m+r-1} \). The preservation of \( m - 1 \) poles is achieved by a suitable
choice of \( q_2 \), while the placement of \( r \) pole is effected by an appropriate choice of \( k_2 \). On applying the output feedback matrix \( K_2 \), the closed-loop characteristic polynomial

\[
p_2(s) = \det(sI - A - BK_1C - BK_2C)
\]

becomes

\[
p_2(s) = p_1(s) - k_2 W_1(s) q_2
\]

(5.9)

where \( W_1(s) = C \text{adj}(sI - A - BK_1C)B \) and \( p_1(s) = \det(sI - A - BK_1C) \). In order to preserve the \( m - 1 \) poles from the first step, we need \( p_2(\lambda_i) = 0 \) for \( i = 1, \ldots, m - 1 \). Since \( p_1(\lambda_i) = 0 \) for \( i = 1, \ldots, m - 1 \), from Eq. (5.9), we require

\[
k_2 W_1(\lambda_i) q_2 = 0 \quad i = 1, \ldots, m - 1
\]

(5.10)

For Eq. (5.10) to hold irrespective of the values of \( k_2 \), the vector \( q_2 \) must satisfy the equations

\[
W_1(\lambda_i) q_2 = 0 \quad i = 1, \ldots, m - 1
\]

(5.11)

It is easy to show that each \( W_1(\lambda_i) \) contains only one independent row denoted by \( w_i \). Thus Eq. (5.11) reduces to the \( m - 1 \) linear equations

\[
w_i q_2 = 0 \quad i = 1, \ldots, m - 1
\]

(5.12)

Equation (5.12) implies that in the \( m \)-dimensional vector space, the vector \( q_2 \) is orthogonal to the \( m - 1 \) linearly independent vectors \( w_1, \ldots, w_{m-1} \). Thus the vector \( q_2 \) for the second stage is chosen to satisfy Eq. (5.12) by specifying one element of \( q_2 \) arbitrarily and solving Eq. (5.12) for the remaining \( m - 1 \) elements. Once \( q_2 \) is found, the \( 1 \times r \) vector \( k_2 \) is obtained by solving the set of \( r \) linear equations

\[
p_2(\lambda_m) = p_1(\lambda_m) - k_2 W_1(\lambda_m) q_2 = 0
\]

\[
\vdots
\]

\[
p_2(\lambda_{m+r-1}) = p_1(\lambda_{m+r-1}) - k_2 W_1(\lambda_{m+r-1}) q_2 = 0
\]

(5.13)

The unity rank output feedback matrix which places the additional \( r \) poles is obtained as \( K_2 = q_2 k_2 \). Finally, obtain the output feedback matrix \( K \) as

\[
K = K_1 + K_2
\]

(5.14)

**Example 5.1**

Given the unstable system

\[
\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} u
\]
\[
    y = \begin{bmatrix}
        1 & 0 & 0 & 0 \\
        0 & 1 & 0 & 0
    \end{bmatrix} x
\]

It is desired to find the output feedback matrix \( K \) which stabilizes the system by placing its poles at \(-1, -2, -3,\) and \(-4\).

**Solution** Using Algorithm 5.1, we have

**Step 1.** We find \( K_1 \) which places two poles at \(-1\) and \(-2\) as follows:

\[
    p_1(s) = p_0(s) - k_1 w_0(s) q_1 = s^4 + 1 - k_1 \begin{bmatrix}
        s^3 & s & 1 \\
        -1 & s^2 & 1
    \end{bmatrix}
\]

Since \( m - 1 = r \), we specify \( q_1 \) and calculate \( k_1 \) for pole placement. Taking \( q_1 = [1 \quad 1 \quad 1]^T \) arbitrarily from Eq. (5.8) we have

\[
    p_1(-1) = k_{11} + k_{12} + 2 = 0
\]

\[
    p_1(-2) = 9k_{11} - k_{12} + 17 = 0
\]

so that \( k_1 = [-1.9 \quad 0.1] \) and \( K_1 \) is obtained as

\[
    K_1 = q_1 k_1 = \begin{bmatrix}
        1 \\
        1
    \end{bmatrix} \begin{bmatrix}
        -1.9 & -0.1 \\
        -1.9 & -0.1
    \end{bmatrix} = \begin{bmatrix}
        -1.9 & -0.1 \\
        -1.9 & -0.1
    \end{bmatrix}
\]

**Step 2.** In this step, we find \( K_2 \), which preserves the poles at \(-1, -2\) and places the poles at \(-3, -4\). From Eq. (5.11) the Eq. (5.12) results to

\[
    \begin{bmatrix}
        -1 & -0.9 & 0.9 \\
        0.9 & 0.2 & -0.1
    \end{bmatrix} q_2 = 0
\]

Taking \( q_{23} = 1 \) arbitrarily, we obtain \( q_2 = (1/61) [-9 \quad 71 \quad 61]^T \). Next, Eq. (5.13) is used to place \(-3, -4\). Thus

\[
    p_2(-3) = \frac{1}{61} (9 k_{21} - 67.1) - 7 = 0
\]

\[
    p_2(-4) = \frac{1}{61} (-27 k_{21} - 71 k_{22} + 85.4) - 12 = 0
\]

and we get \( k_2 = [54.9 \quad -29.98] \).

The second stage output feedback gain \( K_2 \) is

\[
    K_2 = q_2 k_2 = \frac{1}{61} \begin{bmatrix}
        -9 \\
        71 \\
        61
    \end{bmatrix} [54.9 \quad -29.8] = \begin{bmatrix}
        -8.1 & 4.42 \\
        63.9 & -34.9 \\
        54.9 & -29.98
    \end{bmatrix}
\]

The total output feedback matrix for the original system is

\[
    K = K_1 + K_2 = \begin{bmatrix}
        -10 & 4.32 \\
        62 & -35 \\
        53 & -30.08
    \end{bmatrix}
\]
CAD Example 5.1

In this CAD example, we use the primitive "PROP" of CONTROL.lab to solve the Ex. 5.1. Note that in PROP, the output feedback law is of the form \( u = -Ky \) which means that our \( K \) corresponds to \(-K\).

< >

HELP PROP

PROP  Finds the feedback matrix \( K \) for the observable and controllable
MIMO linear time-invariant system as shown

\[
\begin{align*}
    x &= Ax + Bu; \quad y = Cx \\
    u &= -KCx
\end{align*}
\]

PROP(A,B,C,POL,q1)

- \( A \) (nxn)
- \( B \) (nxm)
- \( C \) (1xn)
- \( POL \) ((m + r - 1) \times 2)
- \( q1 \) (mx1) \( m \geq 1 \)
- \( k1 \) (1xr) \( m < 1 \)

Where \( A \) and \( B \) are the system matrices, \( POL \) contains the \( m + r - 1 \leq n \) desired non-repeated poles; s. t. the REAL and IMAGINARY parts are in the first and second column, respectively. Vectors \( q1 \) and \( k1 \) are arbitrarily chosen by the user. If \( m \geq 1 \), then input column vector \( q1 \), otherwise input row vector \( k1 \).

< >

LOAD ('PROP')

< >

A

\[
A =
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-1 & 0 & 0 & 0
\end{bmatrix}
\]

< >

B

\[
B =
\begin{bmatrix}
1 & 0 & 0 \\
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

< >

C
\[ C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \]

\[ \text{POL} \]

\[ \text{POL} = \begin{bmatrix} -1 & 0 \\ -2 & 0 \\ -3 & 0 \\ -4 & 0 \end{bmatrix} \]

\[ \text{Q1} \]

\[ \text{Q1} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \]

\[ K = \text{PROP}(A,B,C,POL,Q1) \]

\[ K = \begin{bmatrix} 10.0000 & -4.3239 \\ -62.0000 & 35.0000 \\ -53.0000 & 30.0845 \end{bmatrix} \]

\[ \text{NOW WE WILL VERIFY THIS.} \]

\[ \text{AHAT} = A - B \times K \times C \]

\[ \text{AHAT} = \begin{bmatrix} -10.0000 & 5.3239 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 \\ 62.0000 & -35.0000 & 0.0000 & 1.0000 \\ 52.0000 & -30.0845 & 0.0000 & 0.0000 \end{bmatrix} \]

\[ \text{EIG(AHAT)} \]

\[ \text{ANS} = \begin{bmatrix} -4.0000 \\ -3.0000 \\ -2.0000 \\ -1.0000 \end{bmatrix} \]

\[ \text{POL} \]
5.2.2 An Iterative Method for Pole Assignment by Output Feedback

In this section, we describe an iterative method to assign closed-loop poles by static output feedback. The method enables one to assign more closed-loop poles compared to the two-step method of the previous section. However, the computation is now more intensive.

Let the characteristic polynomial of the open-loop system be

\[ p(s) = \det(sI - A) = s^n + p_1 s^{n-1} + \cdots + p_n \]  

(5.15)

Then, by applying output-feedback Eq. (5.3), the closed-loop characteristic polynomial will be

\[ \hat{p}(s) = \det(sI - \hat{A}) = s^n + \hat{p}_1 s^{n-1} + \cdots + \hat{p}_n \]  

(5.16)

where \( \hat{A} = A + BK C \).

Expanding Eq. (5.16), we obtain

\[ \hat{p}_i = p_i - L_i + \Phi_i(K) \quad i = 1, 2, \ldots, n \]  

(5.17)

where

\[ L_i = \text{tr} \left( K(p_{i-1}CB + p_{i-2}CAB + \cdots + CA^{i-1}B) \right) \]  

(5.18)

is a linear function in the elements of \( K \) and \( \Phi_i(K) \) is a nonlinear function in the elements of \( K \). Eq. (5.18) can also be written as

\[ L_i = (p_{i-1}e_0 + p_{i-2}e_1 + \cdots + e_{i-1})k \]  

(5.19)

where \( e_i \) is a \( 1 \times mr \) vector formed by placing the rows of \( CA^i B \) one next to the other and \( k \) is an \( mr \times 1 \) vector formed from \( K \) by stacking the columns of \( K \). Substituting Eq. (5.19) in Eq. (5.17), we obtain

\[ \hat{p}_i = p_i - (p_{i-1}e_0 + p_{i-2}e_1 + \cdots + e_{i-1})k + \Phi_i(K) \quad i = 1, 2, \ldots, n \]  

(5.20)

Next, consider the system Eqs. (5.4) and (5.5) with the characteristic polynomial Eq. (5.16) as an open-loop system and apply the feedback matrix \( \delta K \) where the elements of \( \delta K \), that is, \( \delta k_1, \delta k_2, \ldots, \delta k_{mr} \) are sufficiently small. The application of \( \delta K \) causes small changes \( \delta p_1, \delta p_2, \ldots, \delta p_n \) in the coefficients of the characteristic polynomial \( \hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n \). From Eq. (5.20) we have
\[ \hat{p}_i + \delta p_i = \hat{p}_i - (\hat{p}_{i-1} \hat{e}_0 + \hat{p}_{i-2} \hat{e}_1 + \cdots + \hat{e}_{i-1}) \delta k + \hat{\Phi}(\delta k) \ i = 1, 2, \ldots, n \]  

(5.21)

where \( \hat{e}_i \) is a \( 1 \times mr \) vector formed from the rows of the matrix \( C \hat{A}'B = C(A + BKC)'B \) and \( \hat{\Phi}(\delta k) \) is a nonlinear function containing second and higher order product terms. Equation (5.21) may be written as

\[
\begin{bmatrix}
\delta p_1 \\
\delta p_2 \\
\vdots \\
\delta p_n
\end{bmatrix} = -
\begin{bmatrix}
1 & 0 & \cdots & 0 \\
\hat{p}_1 & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
\hat{p}_{n-1} & \hat{p}_{n-2} & \cdots & 1
\end{bmatrix}
\begin{bmatrix}
\hat{e}_0 \\
\hat{e}_1 \\
\vdots \\
\hat{e}_{n-1}
\end{bmatrix} \delta k + 
\begin{bmatrix}
\hat{\Phi}_1(\delta k) \\
\hat{\Phi}_2(\delta k) \\
\vdots \\
\hat{\Phi}_n(\delta k)
\end{bmatrix}
\]

(5.22)

or

\[
\delta p = \hat{P} \hat{E} \delta k + \hat{\Phi}(\delta k)
\]

(5.23)

where \( \delta p \) and \( \hat{\Phi}(\delta k) \) are \( n \times 1 \) vectors and \( \hat{P} \) and \( \hat{E} \) are \( n \times n \) and \( n \times mr \) matrices defined in Eq. (5.22). It is recognized that the set of nonlinear equations in Eq. (5.23) has a solution if the matrix product \( \hat{P} \hat{E} \) is of full rank \( n \). Since the matrix \( \hat{P} \) is nonsingular, the matrix \( \hat{E} \) must be of full rank \( n \) and this in turn establishes the pole assignability condition via static output feedback which we state in the following theorem.

**Theorem 5.1.** Consider the \( n \)-dimensional linear time invariant system Eq. (5.1) and (5.2) and the set of desired poles \( \Lambda_d = \{\lambda_1, \lambda_2, \ldots, \lambda_n\} \). Then for almost all \( \Lambda_d \) there exists a constant output feedback matrix \( K \) such that \( \hat{A} + BKC \) has \( \Lambda_d \) as its eigenvalues if

\[
\text{rank} \hat{E} = \text{rank} \begin{bmatrix}
\hat{e}_0 \\
\hat{e}_1 \\
\vdots \\
\hat{e}_{n-1}
\end{bmatrix} = n
\]

(5.24)

where \( \hat{e}_i \) is a \( 1 \times mr \) vector formed from the rows of the matrix \( C(A + BKC)'B \), \( i = 0, 1, \ldots, n - 1 \) and \( \hat{K} \) is an arbitrary \( m \times r \) matrix.

This theorem does not provide a procedure to obtain the solution. However, for small \( \delta k \), \( \hat{\Phi}(\delta K) \) may be ignored and Eq. (5.23) approximates to

\[
P_j E_j \delta k_j = \delta p_j
\]

(5.25)

where \( P_j \) and \( E_j \) are the values of \( \hat{P} \) and \( \hat{E} \) evaluated for \( K = K_j \) in a typical step \( j \) of the iterative procedure. In each step, the incremental gain vector \( \delta k_j \) is found to make the characteristic coefficient vector \( p_j \) closer to its desired value \( p_d \) by an increment \( \delta p_j = [p_d - p_j] / M \) where \( P_d^T = (p_{1d}, p_{2d}, \ldots, p_{nd}) \) is generated from \( p_d(s) = (s - \lambda_1)(s - \lambda_2)(s - \lambda_n) = s^n + p_{1d}s^{n-1} + \cdots + p_{nd} \), the vector \( p_j^T = (p_{1j}, p_{2j}, \ldots, p_{nj}) \) is generated from the characteristic equation of the matrix \( A_j = A + BK_jC \), and \( M \) is a number chosen to make the elements of \( \delta p_j \) sufficiently
small. Note that in the first step with no previous feedback, we have \( E_1 = E \), where \( E \) is obtained from \( CA'B \). If rank \( E = n \), then the total gain matrix \( K_t \) is equal to

\[
K_t = \sum_{j=1}^{M} \delta K_j
\]

where the incremental gain matrix \( \delta K_j \) corresponds to \( \delta k_j \). If rank \( E < n \) but rank \( \hat{E} = n \), we apply an arbitrary initial feedback \( K_0 \) and then start the procedure. In this case

\[
K_t = K_0 + \sum_{j=1}^{M} \delta K_j
\]

**Corollary 5.1.** For almost all \( \Lambda_d = \{\lambda_1, \lambda_2, \ldots, \lambda_n\} \), the eigenvalues of the system Eqs. (5.1) and (5.2) can be assigned at \( \Lambda_d \) by constant output feedback if

\[
\text{Rank } E = \text{rank } \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{n-1} \end{bmatrix} = n
\]

(5.26)

where \( e_j \) is a \( 1 \times mr \) vector formed from the rows of the matrix \( CA'B \).

We note that condition Eq. (5.26) is stronger than condition Eq. (5.24). In other words, in some systems pole assignment is possible even if rank \( E < n \). However, Eq. (5.26) implies Eq. (5.24) for almost all \( K \). Also, Eq. (5.26) provides a more direct criterion in terms of the system matrices.

The matrix \( \hat{E} \) is of dimension \( n \times mr \), therefore, rank \( \hat{E} = n \) implies that \( mr \geq n \) for pole assignability, which is an expected result. When the rank condition is satisfied, \( \min(n, mr) \) poles can be assigned by constant output feedback. This is considerably greater than \( m + r - 1 \), the number of assignable poles by the direct method of Sec. 5.2.1.

**Algorithm 5.2**

1. Enter the matrices \( A, B, \) and \( C \) and the desired pole locations \( \lambda_1, \lambda_2, \ldots, \lambda_n \).
2. Compute the matrices \( CB, CAB, \ldots, CA^{n-1}B \) and check the rank of the matrix \( E \). If rank \( E = n \), obtain the open loop characteristic coefficient vector \( p_0 \) and go to step 4. Otherwise, go to step 3.
3. Use a random number generator to generate an arbitrary matrix \( K_0 \). Compute \( A_0 = A + BK_0C; CB, CA_0B, \ldots, CA_0^{n-1}B \) and rank \( E_0 \), where \( E_0 \) is obtained from \( CA_iB, i = 0, 1, \ldots, n - 1 \) as before. If rank \( E_0 = n \), compute the vector \( p_0 \) corresponding to \( A_0 \) and go to step 4. Otherwise, the procedure is terminated and the problem has no solution.
4. Set \( j = 0 \) and compute

\[
\delta p_j = \frac{p_d - p_j}{M \| p_d - p_j \|}
\]

where \( M \) is typically chosen as \( M = 20 \). Compute \( \delta k_j \) from Eq. (5.25) and apply the corresponding feedback matrix \( \delta K_j \) to obtain \( A_{j+1} = A_j + B \delta K_j C \) and \( p_{j+1} \).

5. Increase \( j \) by 1 and repeat step 4 until \( \| p_{id} - p_j \| \leq \alpha, \ i = 1, 2, \ldots, n, \) where \( \alpha \) is typically \( 10^{-6} \).

**Example 5.2**

Consider the system

\[
\begin{bmatrix}
-1 & 0.5 & -0.2 & 0.85 & 0.45 & 0.9 \\
0 & -0.5 & -2 & 0.9 & 0.4 & 0.1 \\
0.15 & 0 & -2 & -0.2 & 0.1 & 0.8 \\
0 & 0.1 & -0.25 & -0.8 & 0 & 0.2 \\
-0.2 & 0.4 & 0 & -0.5 & -2 & 0.1 \\
0.6 & -0.7 & 0 & 0.2 & 0 & -2.5
\end{bmatrix} x + \begin{bmatrix}
1 \\
0 \\
1 \\
0 \\
1 \\
0
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1
\end{bmatrix} x
\]

The open loop characteristic polynomial is

\[
p(s) = s^6 + 8.8s^5 + 29.96s^4 + 48.96s^3 + 39.31s^2 + 14.76s + 2.14
\]

It is desired to assign the poles at \( \Delta_d = (-1, -1, -2, -2, -3, -3) \) with

\[
p_d(s) = (s + 1)^2(s + 2)^2(s + 3)^2
\]

\[
= s^6 + 12s^5 + 58s^4 + 144s^3 + 193s^2 + 132s + 36
\]

**Solution** It can be verified that the matrix \( E \) formed from \( CA^TB \), \( i = 0, 1, \ldots, 5 \) has rank 6 and hence by Theorem 5.3, pole assignment is possible. Algorithm 5.2 was used to obtain the required output feedback matrix. The solution was obtained after 9 iterations. The incremental gain vector \( \delta k_j \) and the resulting characteristic coefficient vector \( p_j \) at each iteration follow, where the elements of \( \delta k_j \) are truncated to four decimal points.

<table>
<thead>
<tr>
<th>( j )</th>
<th>( \delta k_j^T )</th>
<th>( p_j^T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(-.1512 \quad -.1925 \quad -.2088)</td>
<td>(8.9 \quad 30.5 \quad 50.8)</td>
</tr>
<tr>
<td></td>
<td>(-.151 \quad -.0238 \quad .0568)</td>
<td>(42.3 \quad 17.1 \quad 2.8)</td>
</tr>
<tr>
<td>2</td>
<td>(.1838 \quad .4785 \quad .0597)</td>
<td>(9.3 \quad 33.9 \quad 62.2)</td>
</tr>
<tr>
<td></td>
<td>(-.3905 \quad -.1207 \quad -.3119)</td>
<td>(60.8 \quad 31.2 \quad 7.0)</td>
</tr>
<tr>
<td>3</td>
<td>(.1865 \quad .6327 \quad .1822)</td>
<td>(9.6 \quad 37.4 \quad 74.1)</td>
</tr>
<tr>
<td></td>
<td>(-.4026 \quad -.1306 \quad -.4075)</td>
<td>(80.0 \quad 45.9 \quad 11.2)</td>
</tr>
</tbody>
</table>
The final characteristic coefficients are equal to the desired coefficients to seven decimal points. The total feedback gain vector is

$$k^T = \sum_{j=1}^{9} \delta k_j^T = \begin{pmatrix} 1.63339 & 4.61372 & 1.17942 & -3.27644 & -10.14436 & -3.06980 \end{pmatrix}$$

and the corresponding feedback matrix is

$$K_1 = \begin{bmatrix} 1.63339 & 4.61372 & 1.17942 \\ -3.27644 & -10.14436 & -3.06980 \end{bmatrix}$$

It is noted that in this example we have been able to assign $mr = 6$ poles. It is also important to note that the feedback matrix required to obtain a specified set of poles is in general nonunique. In fact, if we apply the arbitrary initial feedback matrix

$$K_0 = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

to obtain a new system ($A + BK_0C, B, C$) and then apply Algorithm 5.2 to this new system, we find the total feedback gain matrix for the original system as

$$K_2 = \begin{bmatrix} -6.02217 & 4.16289 & -3.50770 \\ 3.08261 & -1.19304 & 1.73274 \end{bmatrix}$$

Starting with a different initial gain matrix

$$K_0 = \begin{bmatrix} -1 & 1 & -2 \\ -0.3 & -0.1 & 2 \end{bmatrix}$$

we obtain

$$K_3 = \begin{bmatrix} -38.26036 & 49.43701 & 37.55814 \\ 13.97177 & -17.81245 & -13.25994 \end{bmatrix}$$

It can be verified that

$$|sI - A - BK_1C| = |sI - A - BK_2C| = |sI - A - BK_3C| = (s + 1)^2(s + 2)^3(s + 3)^2$$
5.3 TWO AND THREE TERMS CONTROLLERS

In this section, conventional controllers such as PD, PI, and PID which have demonstrated their merits in single-variable control problems are extended to multivariable systems. This combines the attractive feature of the proportional and derivative terms which provide acceptable transient responses through pole placement, and the integral term in which the controller ensures that in the steady state, the output follows step commands and rejects disturbances of any form with constant final values. The material presented in this section closely follow Seraji (1979, 1980) and Seraji and Tarokh (1977a,b, 1978).

5.3.1 Proportional Plus Derivative Output Feedback

The method to be described in this section is an extension of the single step scheme described at the beginning of Section 5.2.1, where max \((m,r)\) poles were assigned using static output feedback. We now show that it is possible to assign max \((2m,2r)\) with PD output feedback.

Consider the linear time invariant system Eqs. (5.1) and (5.2) with \(D = 0\), then on applying proportional plus derivative output feedback

\[ u = v - Py - Q\dot{y} \]  
(5.27)

The system’s closed-loop block diagram can be seen to have the structure depicted in Fig. 5.2, where once again, \(v\) is the \(m \times 1\) command (reference) vector and \(P\) and \(Q\) are constant \(m \times r\) proportional and derivative feedback gain matrices, respectively. Applying control Eq. (5.27) to the system, we get

\[ \dot{x} = (I + BQC)^{-1}(A - BPC)x + (I + BQC)^{-1}BV \]  
(5.28)

The characteristic polynomial of this closed-loop system can be written as

\[ \hat{p}(s) = \text{det}[sl - (I + BQC)^{-1}(A - BPC)] \]
\[ = \text{det}(I + BQC)^{-1} \cdot \text{det}[(I + BQC)sl - A + BPC] \]  
(5.29)
\[ = \frac{1}{\text{det}(I + BQC)} \cdot \text{det}[sl - A + B(P + sQ)C] \]

This equation relates the feedback matrices directly to the closed-loop characteristic polynomial. Note that in this equation, the term \(K(s) = P + sQ\) represents the \(m \times r\) transfer matrix of the PD controller. The matrices \(P\) and \(Q\) are assumed to have a unity rank structure. The first structure is

\[ P = kp \quad Q = kq \]    
(5.30)

where \(k, p,\) and \(q\) are \(m \times 1, 1 \times r,\) and \(1 \times r\) vectors, respectively. The same vector \(k\) has been used for both \(P\) and \(Q\) so that the matrix \(K(s)\) has unity rank. With this unity rank structure, the closed-loop characteristic polynomial Eq. (5.29) becomes
Figure 5.2 A PD controller for a linear multivariable system.

\[
\hat{p}(s) = \frac{\det(sI - A)}{\det(I + BkC)} \frac{\det[I + (sI - A)^{-1}Bk(p + sq)C]}{
\frac{1}{1 + qCBk}[p(s) + (p + sq)W(s)k]}
\]

(5.31)

using the determinant identity \(\det(I + ab) = 1 + ba\) this expression simplifies to

\[
\hat{p}(s) = \frac{1}{1 + qCBk}[p(s) + (p + sq)W(s)k]
\]

(5.32)

where \(W(s) = C \text{adj}(sI - A)B = M_1s^{r-1} + \cdots + M_r\) is the \(r \times m\) numerator polynomial matrix and \(p(s) = \det(sI - A) = s^n + p_1s^{r-1} + \cdots + p_n\) is the open-loop characteristic polynomial. Substituting these polynomials into Eq. (5.32), we obtain

\[
\hat{p}(s) = s^n + \frac{p_1 + pM_1k + qM_2k}{1 + qM_1k} s^{r-1}
\]

\[+ \cdots + \frac{p_{n-1} + pM_{n-1}k + qM_nK}{1 + qM_1k} s + \frac{p_n + pM_nk}{1 + qM_1k}
\]

(5.33)

The second unity rank structure for the feedback matrices \(P\) and \(Q\) is

\[
P = pk \quad Q = qk
\]

(5.34)

where \(p, q, \) and \(k\) are now \(m \times 1, m \times 1,\) and \(1 \times r\) vectors, respectively. Following a similar procedure, we obtain the closed-loop characteristic polynomial

\[
\hat{p}(s) = s^n + \frac{p_1 + kM_1p + kM_2q}{1 + kM_1q} s^{r-1} + \cdots + \frac{p_n + kM_nq}{1 + kM_1q}
\]

(5.35)

The first unity rank structure Eq. (5.30) contains \(m + 2r\) parameters, while the number of parameters in the second unity rank structure Eq. (5.34) is equal to
2m + r. We, therefore, use the former structure for systems with \( r \geq m \) and the latter structure for \( m > r \).

Given the linear multivariable system \((A, B, C)\), the problem is to design suitable unity rank output feedback matrices \((P, Q)\) such that the closed-loop system has a specified set of poles \( \lambda_1, \lambda_2, \ldots, \lambda_n \).

\[
p_d(s) = (s - \lambda_1)(s - \lambda_2) \cdots (s - \lambda_n)
= s^n + d_1s^{n-1} + \cdots + d_{n-1}s + d_n
\]

(5.36)

Three cases can occur depending on the relative values of \( n, m, \) and \( r \).

\textit{Case 1. Systems with } \( n \leq 2r \). In this case, we use the unity-rank structure of Eq. (5.30) for the feedback matrices \( P \) and \( Q \) and equate the coefficients of like powers for \( s \). Using \( p_d(s) = \hat{p}(s) \), we obtain

\[
(p \mid q) \begin{pmatrix} M_1 \\ M_2 - d_1M_1 \\ \vdots \\ M_n - d_{n-1}M_1 \end{pmatrix} k = d_1 - p_1 \\
\vdots \\
(p \mid q) \begin{pmatrix} M_{n-1} \\ M_n - d_{n-1}M_1 \end{pmatrix} k = d_{n-1} - p_{n-1} \\
(p \mid q) \begin{pmatrix} M_n \\ -d_nM_1 \end{pmatrix} k = d_n - p_n
\]

(5.37)

We now specify the vector \( k \) arbitrarily and obtain \( n \) linear equations in the \( 2r \) elements\(^2\) of \( p \) and \( q \). The solution of these equations gives the feedback matrices \( P = kp \) and \( Q = kq \) required for pole assignment. When \( n = 2r \), the equations have a unique solution, and when \( n < 2r \), then \( 2r - n \) elements of \( p \) and \( q \) are specified and the remaining \( n \) elements are found. In general, it is advisable to choose \( k \) such that if an element of \( Q \) is nonzero, then the corresponding element of \( P \) is also nonzero, as it is more practical to use proportional derivative units than purely derivative units.

For a solution to exist, Eq. (5.37) must be consistent, and this requires the \( 2r \times m \) matrices

\[
\begin{pmatrix} M_i \\ M_{i+1} - d_1M_i \end{pmatrix} \quad i = 1, \ldots, n
\]

to be linearly independent, that is, the equation

\[
\alpha_1 \begin{pmatrix} M_1 \\ M_2 - d_1M_1 \end{pmatrix} + \cdots + \alpha_n \begin{pmatrix} M_n \\ -d_nM_1 \end{pmatrix} = 0
\]

(5.38)

must not hold where \( \alpha_1, \ldots, \alpha_n \) are some constant scalars not all zero. If Eq. (5.38) holds, then on premultiplying it by \( (p \mid q) \) and postmultiplying it by \( k \), we obtain

\(^2\) In Eq. (5.37), the reader should distinguish between \( p_i \) as an element of vector \( p \) and coefficients \( p_i \) of the characteristic polynomial in Eq. (5.15).
\[ \alpha_1(d_1 - p_1) + \cdots + \alpha_n(d_n - p_n) = 0 \]

which means that the coefficients of \( p(s) \), and thus the closed-loop poles, cannot be assigned independently. Thus, for arbitrary pole assignment, the matrices

\[
\left( \begin{array}{c}
M_i \\
M_{i+1} - d_i M_1
\end{array} \right)
\]

must be linearly independent. If for the given system \( (A, B, C) \) this condition is not satisfied, arbitrary proportional and derivative feedback matrices \( \hat{P} \) and \( \hat{Q} \) are applied initially to obtain the new system \( (\hat{A}, \hat{B}, \hat{C}) \). This system has the matrices \( M_1, \ldots, M_n \) different from those of the original system, and in this way the linear dependence can, in general, be removed. The pole assignment is now carried out for the new system to obtain the matrices \( P \) and \( Q \). The feedback matrices for the original system are then \( (P + \hat{P}, Q + \hat{Q}) \). It must be noted that singular cases may exist in which these matrices are always linearly dependent, but these cases are uncommon and cannot easily be characterized.

**Example 5.3**

Given the unstable system

\[
x = \begin{bmatrix}
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & -1 & 0
\end{bmatrix} x + \begin{bmatrix}
0 & 0 \\
1 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} u \\
y = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix} x
\]

find a proportional derivative which places the poles at \(-1, -2, -3, -4\).

**Solution** In this example, we have

\[
W(s) = C \text{adj}(sl - A)B = \begin{bmatrix}
s^2 + 1 \\
0 \\
0 \\
1
\end{bmatrix} s^3 + \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix} s^2 + \begin{bmatrix}
0 & 0 \\
0 & 0
\end{bmatrix} s + \begin{bmatrix}
1 & 0 \\
0 & -1
\end{bmatrix}
\]

\[ p(s) = \text{det}(sl - A) = s^4 - 1 \]

The desired characteristic polynomial is

\[ p_d(s) = (s + 1)(s + 2)(s + 3)(s + 4) = s^4 + 10s^3 + 35s^2 + 50s + 24 \]

Substituting in Eq. (5.37), we obtain

\[
(p \mid q) \begin{pmatrix}
0 & 0 \\
0 & 0 \\
1 & 0 \\
0 & 1
\end{pmatrix} k = 10 \quad (p \mid q) \begin{pmatrix}
1 & 0 \\
0 & 1 \\
0 & 0 \\
0 & 0
\end{pmatrix} k = 35
\]
Sec. 5.3 Two and Three Terms Controllers

\[
(p \, q)
\begin{pmatrix}
0 & 0 \\
0 & 0 \\
1 & 0 \\
0 & -1
\end{pmatrix}
k = 50
\quad
(p \, q)
\begin{pmatrix}
1 & 0 \\
0 & -1 \\
0 & 0 \\
0 & 0
\end{pmatrix}
k = 25
\]

where \( P = kp \) and \( Q = kq \) are the proportional and derivative feedback matrices, respectively. On specifying \( k = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \) arbitrarily, we obtain four linear equations in the four elements of \( p \) and \( q \) as

\[
\begin{bmatrix}
0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 \\
0 & 0 & 1 & -1 \\
1 & -1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
q_1 \\
q_2
\end{bmatrix} =
\begin{bmatrix}
10 \\
35 \\
50 \\
25
\end{bmatrix}
\]

The solution is found to be \( p_1 = 30, p_2 = 5, q_1 = 30 \) and \( q_2 = -20 \). The required feedback matrices are then

\[
P = \begin{bmatrix} 1 \\ 1 \end{bmatrix} (30 \quad 5) = \begin{bmatrix} 30 \\ 30 \end{bmatrix}
\]

\[
Q = \begin{bmatrix} 1 \\ 1 \end{bmatrix} (30 \quad -20) = \begin{bmatrix} 30 \\ 30 \end{bmatrix}
\]

It is noted that this problem has no solution using any full rank proportional output feedback

\[
K = \begin{bmatrix}
k_1 & k_2 \\
k_3 & k_4
\end{bmatrix}
\]

since the resulting closed-loop characteristic polynomial

\[
\hat{\rho}(s) = \det(sI - A + BKC) = s^4 + (k_1 + k_4)s^3 + (-1 + k_1)(1 + k_4) - k_2k_3
\]

is unstable for all values of \( k_i \).

Case 2. Systems with \( n \leq 2m \). In this case, we use the unity rank structure of Eq. (5.34) for the feedback matrices \( P \) and \( Q \) and equate the coefficients of like powers for \( s \). Using \( p_d(s) = \hat{\rho}(s) \), we obtain

\[
k (M_1 \mid M_2 - d_2M_1) \begin{bmatrix} p \\ q \end{bmatrix} = d_1 - p_1
\]

\[
\vdots
\]

\[
k (M_n \mid d_nM_1) \begin{bmatrix} p \\ q \end{bmatrix} = d_n - p_n
\]

On specifying the vector \( k \) arbitrarily, we obtain \( n \) linear equations in the \( 2m \) elements of \( p \) and \( q \). When \( n = 2m \), the equations have a unique solution, and when \( n < 2m \),
then \(2m - n\) elements of \(p\) and \(q\) are specified and the remaining \(n\) elements are found. The required feedback matrices are found as \(P = pk\) and \(Q = qk\).

**Example 5.4**

Given the system

\[
\begin{align*}
\dot{x} &= \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 \\
\end{bmatrix} x + \begin{bmatrix}
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 1 \\
\end{bmatrix} u \\
y &= \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
\end{bmatrix} x
\end{align*}
\]

find a proportional derivative output feedback which places all five poles at \(-1\).

**Solution** In this example, we have

\[
W(s) = \begin{bmatrix}
s^3 & s^2 & 0 \\
0 & s^3 + 1 \\
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 0 & 1 \\
\end{bmatrix} s^3 + \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} s^2 + \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]

\[
p(s) = s^5 + s^2
\]

\[
p_d(s) = (s + 1)^5 = s^5 + 5s^4 + 10s^3 + 10s^2 + 5s + 1
\]

Setting \(k = (1 \ 1)\) arbitrarily and substituting in Eq. (5.39), we obtain five linear equations in the six elements of \(p\) and \(q\) as

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
\end{bmatrix} \begin{bmatrix}
p_1 \\
p_2 \\
p_3 \\
q_1 \\
q_2 \\
q_3 \\
\end{bmatrix} = \begin{bmatrix}
5 \\
10 \\
9 \\
5 \\
5 \\
1 \\
\end{bmatrix}
\]

Specifying \(p_1 = 1\) arbitrarily, we find

\[
P = \begin{bmatrix}
1 \\
9 \\
1 \\
\end{bmatrix} (1 \ 1) = \begin{bmatrix}
1 & 1 \\
9 & 9 \\
1 & 1 \\
\end{bmatrix} \quad Q = \begin{bmatrix}
0 \\
8 \\
5 \\
\end{bmatrix} (1 \ 1)
\]

\[
= \begin{bmatrix}
0 & 0 \\
8 & 8 \\
5 & 5 \\
\end{bmatrix}
\]

It is interesting to note that using purely proportional output feedback, this problem does not have any solution even with a full-rank feedback matrix.
We conclude that for systems with \( n \leq \max(2r, 2m) \), all \( n \) closed-loop poles can be assigned arbitrarily using unity rank proportional and derivative feedback matrices and the required matrices are obtained from simple linear equations.

**Case 3. Systems with \( n > \max(2r, 2m) \).** In this case, one can choose between two approaches. The first approach is to specify only \( l = \max(2r, 2m) \) poles of the closed-loop system and set \( \ddot{p}(s) \) equal to zero at these poles. On specifying the vector \( k \), we obtain \( l \) linear equations in the two unknown vectors \( p \) and \( q \). The feedback matrices which assign \( l \) poles are then found. The remaining \( n - l \) poles of the closed-loop system move to unspecified locations. It is noted that for \( l = 2r \), we use the unity-rank structure of Eq. (5.30) and \( \ddot{p}(s) \) from Eq. (5.33) and for \( l = 2m \) we use Eqs. (5.34) and (5.35).

The second approach is to extend the two-step method of Section 5.2.1 to PD output feedback. It is easy to show that using the two-step method, it is possible to assign \( \max(2m + r - 1, m + 2r - 1) \) closed-loop poles with PD output feedback. This is done by assigning \( r - 1 \) and \( 2m \) poles in the first and second steps, respectively, to obtain \( 2m + r - 1 \) desired closed-loop poles. Alternatively, if \( m - 1 \) and \( 2r \) poles are placed in the first and second steps, respectively, we obtain a total of \( m + 2r - 1 \) desired closed-loop poles. Details are left as an exercise.

**Example 5.5**

Given the system

\[
\dot{x} = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix} x + \begin{bmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0
\end{bmatrix} x
\]

find a proportional derivative output feedback which places all five poles at \(-1\).

**Solution** In this example, we have

\[
W(s) = \begin{bmatrix}
s^2 & 0 \\
0 & s^3 + 1
\end{bmatrix}
\]

\[
p(s) = s^5 + s^2
\]

\[
\ddot{p}(s) = (s + 1)^5 = s^5 + 5s^4 + 10s^3 + 10s^2 + 5s + 1
\]

Using the unity rank feedback matrices \( P = kp \) and \( Q = kq \), Eq. (5.37) becomes

\[
(p \ q) \begin{bmatrix}
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} k = 5
\]

\[
(p \ q) \begin{bmatrix}
0 & 0 \\
0 & 1 \\
0 & 0 \\
0 & 0
\end{bmatrix} k = 10
\]
An exact solution for these equations is found to be

\[(p \quad q) = (9 \quad 1 \quad 9 \quad 5) \quad k = (1 \quad 1)^T\]

The required unity rank feedback matrices are then

\[P = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad Q = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \]

The five closed-loop poles are found to be at \(-1\) as specified.

**CAD Example 5.2**

Let us use the primitive "PRDR" of CONTROL.lab to solve an example.

```
<>
HELP PRDR
PRDR  Find the feedback matrices Kp=P & Kd=Q for the observable and
controllable MIMO linear time-invariant system:

\[\dot{x} = Ax + Bu; \quad y = Cx\]
\[u = -Kp*y - Kd*dy/dt\]

PRDR(A,B,C,POL,q1)

A   (n\times n)
B   (n\times m)
C   (r\times n)
POL  ((2r + m - 1)\times 2)  \quad m<1 \quad OR \quad (2r + m - 1)\times 2 \quad m\geq 1
q1  (m\times 1) \quad m\geq 1
k1  (1\times r) \quad m<1

Where A and B are the system matrices, POL contains the 2m + r - 1 \leq n, if m<r
or 2r + m - 1 \leq n for m \geq 1, desired nonrepeated poles; s.t. the REAL and
IMAGINARY parts are in the first and second column, respectively. Vectors
q1 and k1 are arbitrarily chosen by the user. If \( m \geq 1 \) then input column

vector q1, otherwise input row vector k1.

\[
\begin{align*}
\text{<> LOAD(\"PDR\") <> A} \\
A & = \\
& \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\text{<> B} \\
B & = \\
& \begin{bmatrix}
0 & 1 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
1 & 0
\end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\text{<> C} \\
C & = \\
& \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0
\end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\text{<> // 2r + m - 1 = 4 + 2 - 1 = 5; n = 5} \\
\text{<> POL} \\
\text{POL} & = \\
& \begin{bmatrix}
-1 & 0 \\
-2 & 0 \\
-3 & 0 \\
-1 & 1 \\
-1 & -1
\end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\text{<> Q1} \\
\text{Q1} & = \\
& \begin{bmatrix}
2 \\
0
\end{bmatrix}
\end{align*}
\]
\[ \begin{pmatrix} 0.9500 & 1.2500 \\ -0.9500 & -1.2500 \end{pmatrix} \]

\[ \begin{pmatrix} 1.6000 & 0.7500 \\ -1.6000 & -0.7500 \end{pmatrix} \]

// LET US VERIFY THIS.

\[
\begin{matrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1
\end{matrix}
\]

\[
\begin{matrix}
-8.0000 & 20.0000 & 15.0000 & 25.0000 & 0.0000 \\
0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 \\
7.0000 & -19.0000 & -15.0000 & -25.0000 & 0.0000
\end{matrix}
\]

\[
\begin{matrix}
-3.0000 & -0.0000i \\
-1.0000 & +1.0000i \\
-2.0000 & +0.0000i \\
-1.0000 & +0.0000i \\
-1.0000 & -1.0000i
\end{matrix}
\]

// QED
Figure 5.3 shows the stabilization effect of PD controller on the unstable plant. We used capabilities of MATRIXx to obtain the step response of open-loop and closed-loop systems.

5.3.2 Proportional Plus Integral Output Feedback

In order to improve the steady-state behavior of the system resulting in a satisfactory regulation, one would need to use an integrator element in the control law. In this subsection, a proportional plus integrator controller will be investigated for a linear system.

\[
\dot{x} = Ax + Bu + Ed \\
y = Cx + Fd
\]  \hspace{1cm} (5.40) \hspace{1cm} (5.41)

where all vector and matrices are appropriately dimensioned and \( d \) is the \( r \times 1 \) disturbance input vector. The system matrix \( A \) is cyclic\(^3\) or is made cyclic by the initial application of an arbitrary constant output feedback matrix.

It is desired to design a simple controller acting directly on the outputs, such that the resulting closed-loop system meets the following three specifications.

\(^3\) For a definition of cyclic matrices, see Section 4.3.3.
1. When the system is excited by the \( r \times 1 \) step command (reference or set-point) vector \( v(t) = V \), the output vector \( y(t) \) becomes equal to \( V \) in the steady state.

2. When the system is subjected to the unmeasurable arbitrary disturbance vector \( d(t) \) with constant final value, the output vector \( y \) is not affected in the steady state.

3. The poles of the closed-loop system are at specified locations in the complex plane.

Specification 1 ensures that in the closed-loop system, a step change in any command will cause an equal change only in the corresponding output and the remaining outputs will not be altered in the steady state. Thus, the closed-loop system is steady-state decoupled for step commands. Specification 2 implies that for unmeasurable disturbances of arbitrary time functions with constant final values, the outputs exhibit transient responses but will not be affected in the steady state. Specification 3 is relevant to the degree of stability and the characteristics of transient responses of the closed-loop system. Thus, acceptable transient responses and stability margin may be obtained by proper selection of the closed-loop pole positions.

In order to meet the specifications 1 to 3, a proportional-plus-integral (PI) controller is designed to act directly on the available outputs. The integral term \( Q \) in the controller is in the forward loop and acts on the error \( e = v - y \) and the proportional term \( P \) is in the feedback loop and acts on the output \( y \), as shown in Fig. 5.4. Because of the integral term, the order of the system is increased by \( r \) and the \( r \) additional state variables are elements of the \( r \times 1 \) vector \( z = \int (v - y) \, dt \). Rewriting Eqs. (5.40) and (5.41) to include \( z \), we obtain the \((n + r)\)th order state model of the open-loop system augmented by the \( r \) integrators as

\[
\begin{bmatrix}
\dot{x} \\
\dot{z}
\end{bmatrix} =
\begin{bmatrix}
A & 0 \\
-C & 0
\end{bmatrix}
\begin{bmatrix}
x \\
z
\end{bmatrix} +
\begin{bmatrix}
B \\
0
\end{bmatrix}
\begin{bmatrix}
u \\
1
\end{bmatrix} +
\begin{bmatrix}
E \\
-F
\end{bmatrix} d
\tag{5.42}
\]

![Figure 5.4](image)

Figure 5.4 A PI controller for a multivariable system.
\[
\begin{bmatrix}
y \\
z 
\end{bmatrix} = \begin{bmatrix}
C & 0 \\
0 & I 
\end{bmatrix} \begin{bmatrix}
x \\
z 
\end{bmatrix} + \begin{bmatrix}
F \\
0 
\end{bmatrix} d 
\] (5.43)

and the PI control law is given by

\[
u = -Py + Q \int (v - y) \, dt = -[P - Q] \begin{bmatrix}
y \\
z 
\end{bmatrix} 
\] (5.44)

It is seen that the control law for the augmented system \((A^*, B^*, C^*)\) with

\[
A^* = \begin{bmatrix}
A & 0 \\
-C & 0 
\end{bmatrix} \quad B^* = \begin{bmatrix}
B \\
0 
\end{bmatrix} \quad C^* = \begin{bmatrix}
C & 0 \\
0 & I 
\end{bmatrix} 
\] (5.45)

is of the constant output feedback type. Necessary conditions for arbitrary pole placement using output feedback are that the augmented system \((A^*, B^*, C^*)\) be both controllable and observable. Furthermore, to have a simple and exploitable relationship between the closed-loop poles and the controller matrices \(P\) and \(Q\), these matrices are initially assumed to have unity rank. This requires the augmented system matrix \(A^*\) to be cyclic for pole placement to be possible. Seraji (1975) established the conditions for controllability, observability and cyclicity of \((A^*, B^*, C^*)\). These are briefly given here.

1. Controllability: The augmented system \((A^*, B^*)\) is controllable, if and only if (a) the original system \((A, B)\) is controllable, that is, rank \([B, AB, \ldots, A^{n-1}B]\) = \(n\); and (b) the matrix \(\begin{bmatrix} A & B \\ C & 0 \end{bmatrix}\) is of full rank \(n + r\). Note that from condition (b), we need \(m \geq r\) and that the controllability of \((A, B)\) is necessary but not a sufficient condition for the controllability of \((A^*, B^*)\).

2. Observability: It can be readily shown that the necessary and sufficient condition for observability of the augmented system \((A^*, C^*)\) is that the original system \((A, C)\) be observable, that is, rank \([C^T, (CA)^T, \ldots, (CA^{n-1})^T]^T = n\).

3. Cyclicity: The necessary and sufficient condition for cyclicity of the augmented system matrix \(A^* = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix}\) is that the rational matrix \((sl - A^*)^{-1}\) be irreducible. It can be verified that

\[
(sI - A^*)^{-1} = \frac{1}{s' \det(sI - A)} \begin{bmatrix}
s' \adj(sI - A) & 0 \\
-s'^{-1}C \adj(sI - A)s'^{-1} \det(sI - A) I 
\end{bmatrix} 
\]

For multioutput systems \((r > 1)\), the factor \(s'^{-1}\) is cancelled in \((sl - A^*)^{-1}\) and hence \(A^*\) is not cyclic. For single-output systems \((r = 1)\), a sufficient condition for
cyclicity of $A^\ast$ is that the original system $(A, C)$ be observable, since in this case the numerator terms $C \text{adj} (sl - A)$ and $\det (sl - A)$ do not have any common factors and $(sl - A^\ast)^{-1}$ is irreducible. The PI controller is designed such that $2m + r - 1$ poles of the $(n + r)$th-order closed-loop system are placed at specified locations $\lambda_1, \ldots, \lambda_{2m+r-1}$ in the $s$-plane.

**Design Method.** The design process is summarized by the following three steps.

**Step 1.** The augmented system $(A^\ast, B^\ast, C^\ast)$ is made cyclic by applying the feedback law $u = \hat{Q}z + \hat{u}_c$, where $\hat{Q}$ is an arbitrary matrix of full rank $r$ and $\hat{u}_c$ is the $m \times 1$ control vector. The state model of the resulting system is

$$
\begin{aligned}
\begin{bmatrix}
\dot{x} \\
\dot{z}
\end{bmatrix} &=
\begin{bmatrix}
A & B\hat{Q} \\
-C & 0
\end{bmatrix}
\begin{bmatrix}
x \\
0
\end{bmatrix} +
\begin{bmatrix}
0 \\
I
\end{bmatrix} \hat{u}_c +
\begin{bmatrix}
0 \\
-F
\end{bmatrix} v +
\begin{bmatrix}
E \\
0
\end{bmatrix} d \\
\begin{bmatrix}
y \\
z
\end{bmatrix} &=
\begin{bmatrix}
C & 0 \\
0 & I
\end{bmatrix}
\begin{bmatrix}
x \\
z
\end{bmatrix} +
\begin{bmatrix}
0 \\
F
\end{bmatrix} d
\end{aligned}
$$

(5.46) (5.47)

The new system matrix $A_1^\ast = \begin{bmatrix} A & B\hat{Q} \\ -C & 0 \end{bmatrix}$ has distinct eigenvalues and is therefore cyclic. The transfer function model of the system is

$$
\begin{bmatrix} Y(s) \\ Z(s) \end{bmatrix} = \frac{1}{\det(sl - A^\ast)} C^\ast \text{adj}(sl - A_1^\ast) B^\ast \hat{U}_c(s)
$$

(5.48)

$$
= \frac{1}{p_1(s)} \begin{bmatrix} W_1(s) \\ -W_1(s)s \end{bmatrix} \hat{U}_c(s)
$$

where $W_1(s) = (C \ 0)\text{adj}(sl - A_1^\ast)B^\ast$, $p_1(s) = \det(sl - A_1^\ast)$ and noting that, in the absence of $V$, $Z(s) = -Y(s)/s = [-W_1(s)/p_1(s)]\hat{U}_c(s)$.

**Step 2.** In this step, $r - 1$ poles of the system $(A_1^\ast, B^\ast, C^\ast)$ are placed at the distinct specified locations $\lambda_1, \ldots, \lambda_{r-1}$ by applying the unity-rank feedback $\hat{u}_c = \hat{q}kz + u_c$, where $\hat{q}$ and $k$ are $m \times 1$ and $1 \times r$ vectors, respectively and $u_c$ is the $m \times 1$ control vector. Here, the closed-loop poles must be distinct for preservation of poles to be possible in step 3. The resulting closed-loop characteristic polynomial can be expressed as

$$
p_2(s) = p_1(s) + \frac{1}{s} k W_1(s) \hat{q}
$$

(5.49)

In order to place $r - 1$ poles at $\lambda_1, \ldots, \lambda_{r-1}$ the vector $\hat{q}$ is specified arbitrarily such that $(A_1^\ast, B^*\hat{q})$ is controllable and the vector $k$ is found by solving the set of $r - 1$ linear equations.
\[ p_i(\lambda_i) + \frac{1}{\lambda_i} \hat{k} W_i(\lambda_i) \hat{q} = 0 \quad i = 1, \ldots, r - 1 \]  

(5.50)

then

\[
\begin{bmatrix}
\dot{x} \\
\dot{z}
\end{bmatrix} = \begin{bmatrix}
A & B (\hat{Q} + \hat{q} \hat{k}) \\
-C & 0
\end{bmatrix} \begin{bmatrix}
x \\
z
\end{bmatrix} + \begin{bmatrix}
B \\
0
\end{bmatrix} u_c + \begin{bmatrix}
0 \\
I
\end{bmatrix} v + \begin{bmatrix}
E \\
-F
\end{bmatrix} d
\]  

(5.51)

\[
\begin{bmatrix}
y \\
z
\end{bmatrix} = \begin{bmatrix}
C & 0 \\
0 & I
\end{bmatrix} \begin{bmatrix}
x \\
z
\end{bmatrix} + \begin{bmatrix}
F \\
0
\end{bmatrix} d
\]  

(5.52)

and \( r - 1 \) poles of this system are at \( \lambda_1, \ldots, \lambda_{r-1} \). The transfer-function model of the system is

\[
\begin{bmatrix}
Y(s) \\
Z(s)
\end{bmatrix} = \frac{1}{\det(sI-A_2^*)} C^* \ \text{adj}(sI-A_2^*) B^* \ \hat{U}_c(s)
\]

(5.53)

\[
= \frac{1}{p_2(s)} \begin{bmatrix}
W_2(s) \\
s W_2(s)
\end{bmatrix} \hat{U}_c(s)
\]

where \( W_2(s) = (C \ 0) \ \text{adj}(sI-A_2^*)B^* \), \( p_2(s) = \det(sI-A_2^*) \) and

\[ A_2^* = \begin{bmatrix}
A & B (\hat{Q} + \hat{q} \hat{k}) \\
-C & 0
\end{bmatrix} \]

Step 3. In this step, the \( r - 1 \) poles of the system \((A_2^*, B^*, C^*)\) at \( \lambda_1, \ldots, \lambda_{r-1} \) are preserved and \( 2m \) additional poles are placed at the specified locations \( \lambda_1, \ldots, \lambda_{2m+r-1} \) by applying the unity-rank feedback law \( u_c = -pky + qkz \), where \( k, p \) and \( q \) are \( 1 \times r \), \( m \times 1 \), and \( m \times 1 \) vectors, respectively. The resulting closed-loop characteristic polynomial can be expressed as

\[ p_3(s) = p_2(s) + k W_2(s) p + \frac{1}{s} k W_2(s) q \]  

(5.54)

In order to preserve the \( r - 1 \) poles at \( \lambda_1, \ldots, \lambda_{r-1} \) in the closed-loop system irrespective of \( p \) and \( q \), from Eq. (5.54), the vector \( k \) must satisfy the equations

\[ k W_2(\lambda_i) = 0 \quad i = 1, \ldots, r - 1 \]  

(5.55)

Since the matrices \( \text{adj}(\lambda_i I - A_2^*) \), \( i = 1, \ldots, r - 1 \), have unity rank, each of the matrices \( W_2(\lambda_i) \) contains only one independent column denoted by \( w_i \). Thus, the vector \( k \) is found from the \( r - 1 \) linear equations

\[ k w_i = 0 \quad i = 1, \ldots, r - 1 \]  

(5.56)
Once $k$ is determined, the $m \times 1$ vectors $p$ and $q$ which place $2m$ additional poles at $\lambda_1, \ldots, \lambda_{2m+1}$ are found by solving the set of $2m$ linear equations

$$p_2(\lambda_i) + kW_2(\lambda_i)p + \frac{1}{\lambda_i} kW_2(\lambda_i)q = 0 \quad i = 1, \ldots, 2m + r - 1 \quad (5.57)$$

The total PI control law is $u = -Py + Q \int (v - y) \, dt$, where $P = pk$ is the unity rank $m \times 1$ proportional feedback matrix and $Q = \hat{Q} + \hat{q}k + qk$ is the $m \times r$ integral feedback matrix of full rank $r$. The state model of the final $(n + r)$th-order closed-loop system is

$$\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} A - BPC & BQ \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} v + \begin{bmatrix} E - BPF \\ -F \end{bmatrix} d \quad (5.58)$$

$$y = (C \quad 0) \begin{bmatrix} x \\ z \end{bmatrix} + Fd \quad (5.59)$$

$2m + r - 1$ closed-loop poles are at the specified locations $\lambda_1, \ldots, \lambda_{2m+r-1}$.

In step 3, the choice of $K$ for preservation of poles make the single-output systems $[A_f^*, B^*, k(C \quad 0)]$ and $[A_f^*, B^*, k(0 \quad I)]$ partially unobservable by producing pole-zero cancellations at $s = \lambda_1, \ldots, \lambda_{r-1}$ in the transfer function vectors $kW_2(s)/p_2(s)$ and $-kW_2(s)/sp_2(s)$. Subsequently, the unobservable poles at $\lambda_1, \ldots, \lambda_{r-1}$ remain invariant under the output feedback vectors $p$ and $q$. The number of poles that can be placed arbitrarily by $p$ and $q$ is given by $\rho = \text{rank} [R_0^T (B^* \ A_f^* \ B^*)]$, where $R_0$ is the observability matrix of $[A_f^*, B^*, k(0 \quad I)]$. Thus the total number of poles that can be placed by the PI controller is equal to $\rho + r - 1$. However, for a general system $(A, B, C)$, $\rho = 2m$ and $2m + r - 1$ poles can be placed arbitrarily.

For systems with $n < 2m$, all $n + r$ closed-loop poles can be placed arbitrarily using the PI controller. When $n \geq 2m$, $2m + r - 1$ poles of the closed-loop system can be placed arbitrarily but the remaining $n - 2m + 1$ poles move to unspecified locations. If these locations are undesirable, the matrix $\hat{Q}$ or the vector $\hat{q}$ are altered and a new controller is calculated. Since for many practical multivariable systems the number of unspecified poles $(n - 2m + 1)$ is small and the design method is computationally fast, a suitable controller may be obtained by repeating the design procedure a number of times. In practice, the transient responses of the system are largely dependent on the locations of the dominant poles and placement of all closed-loop poles is not essential for satisfactory transient responses. Typically, the system contains actuators whose poles are sufficiently far in the complex plane and these poles need not be placed by the controller.
Before the design method is applied to a numerical example, we briefly discuss the steady-state characteristics of PI controller.

**Steady-state characteristics.** Let us investigate the steady-state behavior of the closed-loop system when subjected to step commands or disturbances with constant final values.

Consider the closed-loop system shown in Fig. 5.4 with the PI controller $K(s)$ designed based on this method. We first take the case where the disturbance $d$ is not acting on the system and the system is subjected to the command $v$. The control input is given by

$$U(s) = K(s) [V(s) - Y(s)]$$  \hspace{1cm} (5.60)

The output is related to the control input by $Y(s) = G(s)U(s)$, where $G(s) = C(sI - A)^{-1}B$. Hence we obtain

$$Y(s) = [I + G(s)K(s)]^{-1}G(s)K(s)V(s)$$  \hspace{1cm} (5.61)

Substituting for $K(s)$ and $V(s) = V/s$, where $V$ is the constant command vector, we have

$$Y(s) = \left[ I + G(s)\left( P + \frac{Q}{s} \right) \right]^{-1}G(s)\left( P + \frac{Q}{s} \right)\frac{V}{s}$$

$$= [sI + G(s)(Ps + Q)]^{-1}G(s)(Ps + Q)\frac{V}{s}$$  \hspace{1cm} (5.62)

As the closed-loop system is stable through pole placement, the output reaches a constant steady-state value given by

$$y_{ss} = \lim_{s \to 0} sY(s) = [G(0)Q]^{-1}G(0)QV = V$$  \hspace{1cm} (5.63)

Thus, for any desired constant command vector $V$, the output vector $y$ becomes equal to $V$ in the steady state. This means that in the steady state the outputs follow step commands.

We now take the case where the commands are equal to zero and the system is subjected to the disturbance $d(t)$ with the constant final value $\lim_{t \to \infty} d(t) = D$. The output is related to the disturbance by $Y(s) = G(s)D(s)$, where $G(s) = C(sI - A)^{-1}E$. In the absence of $v$, the output of the closed-loop system is given by

$$Y(s) = G(s)U(s) + G(s)D(s)$$

$$= -G(s)K(s)Y(s) + G(s)D(s)$$

or

$$Y(s) = [I + G(s)K(s)]^{-1}G(s)D(s)$$  \hspace{1cm} (5.64)

Substituting for $K(s)$, we obtain

$$Y(s) = [sI + G(s)(Ps + Q)]^{-1}G(s)[sD(s)]$$  \hspace{1cm} (5.65)
The steady-state value of the output is

\[ y_{ss} = \lim_{s \to 0} sY(s) = [G(0)Q]^{-1}G(0)[\lim_{s \to 0} sD(s)] = 0 \] (5.66)

since \( \lim_{s \to 0} sD(s) = d(\infty) = D \). Thus, when the system is subjected to disturbances of any time functions with constant final values, the outputs exhibit transient responses but will not be affected in the steady state. When the system is subjected to both constant command \( V \) and disturbances with constant final values, then by superposition principle, the output vector \( y \) becomes equal to \( V \) in the steady state.

**CAD Example 5.3**

\begin{verbatim}
< >
HELP PRPI

PRPI   Find the feedback matrices kp=P and ki=Q for an observable and controllable MIMO linear time-invariant system:

\[
\dot{x} = Ax + Bu; \quad y = Cx \\
u = xp*y - ki*Int(e); \quad e = v - y
\]

PRPI(A,B,C,POL,Qhat,q1 or k1)

A (n x n) 
B (n x m) 
C (r x n) 
POL ((2m+r-1) x 2) 
Qhat (m x 1) 
q1 (m x 1) m>1 
k1 (1 x r) m<1

Where A and B are the system matrices, POL contains the \( 2m+r-1 \leq n+1 \) desired nonrepeated poles; s.t. the REAL and IMAGINARY parts are in the first and second column, respectively. Qhat, q1, and k1 are arbitrarily chosen by the user. If \( m>1 \) then input column vector q1, otherwise input row vector k1.

< >
LOAD('PRPI')

< >
A

A =

1 3 2
0 1 2
0 0 1
\end{verbatim}
\[
B = \\
\begin{bmatrix}
1 & 0 \\
2 & 0 \\
1 & 1 \\
\end{bmatrix}
\]

\[
C = \\
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
\end{bmatrix}
\]

\[
2m + r - 1 = 4 + 2 - 1 = 5; \ n + r = 3 + 2 = 5
\]

\[
POL = \\
\begin{bmatrix}
-1 & 0 \\
-2 & 0 \\
-3 & 0 \\
-4 & 0 \\
-5 & 0 \\
\end{bmatrix}
\]

\[
QHAT = \\
\begin{bmatrix}
-1 & 0 \\
0 & -1 \\
\end{bmatrix}
\]

\[
Q1 = \\
\begin{bmatrix}
0 \\
1 \\
\end{bmatrix}
\]

\[
<KP,KI> = PRPI(A,B,C,POL,QHAT,Q1)
\]
KI

\[\begin{bmatrix}
-12.6327 & -3.8776 \\
91.5711 & 32.8570
\end{bmatrix}\]

KP

\[\begin{bmatrix}
10.8000 & 3.60000 \\
16.7954 & 5.5985
\end{bmatrix}\]

// LET US VERIFY THIS RESULT

A1 = A - B*KP*C

\[\begin{bmatrix}
-9.8000 & -0.6000 & 2.0000 \\
-21.6000 & -6.2000 & 2.0000 \\
-27.5954 & -9.1985 & 1.0000
\end{bmatrix}\]

A2 = B*KI

\[\begin{bmatrix}
-12.6327 & -3.8776 \\
-25.2654 & -7.7551 \\
78.9384 & 28.9795
\end{bmatrix}\]

A3 = -C

\[\begin{bmatrix}
-1 & 0 & 0 \\
0 & -1 & 0
\end{bmatrix}\]

A4 = \langle 0 \ 0 \ ; \ 0 \ 0 \rangle

\[\begin{bmatrix}
0 & 0 \\
0 & 0
\end{bmatrix}\]
\[ AHAT = \begin{bmatrix} A1 & A2; A3 & A4 \end{bmatrix} \]

\[
AHAT = \\
\begin{bmatrix}
-9.8000 & -0.6000 & 2.0000 & -12.6327 & -3.8776 \\
-1.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & -1.0000 & 0.0000 & 0.0000 & 0.0000
\end{bmatrix}
\]

\[ EIG(AHAT) \]

\[
ANS = \\
\begin{bmatrix}
-5.0000 & -0.0000i \\
-4.0000 & +0.0000i \\
-3.0000 & -0.0000i \\
-2.0000 & -0.0000i \\
-1.0000 & +0.0000i
\end{bmatrix}
\]

\[ POL \]

\[
POL = \\
\begin{bmatrix}
-1 & 0 \\
-2 & 0 \\
-3 & 0 \\
-4 & 0 \\
-5 & 0
\end{bmatrix}
\]

// QED

Figure 5.5 shows the steady-state characteristic of PI controller. The step response of the system is obtained by using MATRIXs primitives.

### 5.3.3 Proportional Plus Integral Plus Derivative Output Feedback

The design of Proportional-Integral-Derivative (PID) controllers for linear multivariable systems with the controllers using only the available system outputs combines the attractive features of PI and PD controllers discussed so far.

In order to meet the steady state specification of asymptotic tracking and disturbance rejection, an integral term is provided in the controller. However, this term
introduces poles at the origin and tends to make the closed-loop system less stable, and therefore, stabilizing terms are needed in the controller. The proportional and derivative terms have the required stabilizing effect and when included in the controller can provide acceptable stability margin and transient responses by placing the poles of the closed-loop system at suitable locations in the complex plane.

Thus, this argument suggests that a PID controller should be employed in control systems in which improvement in both the transient response and the steady-state response are required. Similar to PD and PI, a PID routine is available in CONTROL.lab which facilitates the design of the controller (see Prob. 5.9). We conclude our discussion by pointing out that several related design methods which also include the robustness of PID-type controllers for multivariable systems can be found in Pang and MacFarlane (1987) and Lunze (1989).

5.4 DYNAMIC COMPENSATOR

For SISO systems, there are several design methods available based on transfer function approach. To develop the MIMO analogous results, the concept of matrix fraction description should be introduced. However, as we pointed out in the introduction chapter of this text, we shall not discuss the possibility of direct transfer
function design, instead, we refer the interested readers to Kailath (1980), Chen (1984), and Vidyasagar (1985). These methods allow one to obtain results exactly equivalent to those obtained by use of an observer based controller but also suggest more general compensation schemes. The design methods of proportional, two- and three-term controllers considered in Sect. 5.2 and 5.3 are based on both state space and transfer function without requiring the access of state variables. Here we shall continue our discussion along the same line, however, we consider both aspects of feedback compensation namely state and output feedback. Subsequently, we show that the dynamic feedback compensator of multivariable systems may be reduced to an equivalent static output feedback design problem considered previously.

5.4.1 State-Space Design for Stabilization, Tracking, and Disturbance Rejection

If a dynamical equation is controllable, then all the eigenvalues can be arbitrarily assigned by any one of the methods discussed so far. Generally speaking, for a stabilizable system in which the uncontrollable part of the system has stable eigenvalues, we can shift the unstable eigenvalues to the desired location. It turns out that the design of stabilizing compensators can be very broad, and therefore, one is seeking first to obtain all compensators that stabilize a given plant, then to select one that meets certain performance specifications. One of the most important goals in control system design is to require the output of the plant to track the reference signal and at the same time to reject the disturbance acting on the plant or compensator. Because of practical limitations, this can be only achieved asymptotically and consequently, it is termed asymptotic tracking. It is also important to have an idea of the nature and extent of the uncertainties in the plant and/or compensator that can be permitted without destroying the stability of the compensated system. Thus, given a nominal plant together with a description of the plant uncertainty, the objective is to design a compensator that stabilizes all plants lying within the specified band of uncertainty. This is referred to as robust stabilization. Similarly, the objective of the robust tracking problem is to design a compensator such that the plant output tracks the reference signal and continues to do so even as the plant is slightly perturbed. The details of these interrelationships between stability robustness and performance robustness are outside the scope of this text, however. Refer to Sain (1981), Vidyasagar (1985), and Dorato and Yedavalli (1990) Francis (1987), Bhattacharyya (1987), Dorato (1987), Zafiriou and Morari (1989) if interested. Only a few important issues in the design of control systems are emphasized here.

In Chap. 1, we briefly discussed the SISO feedback control system characteristics and explored the role of compensator which enhance such characteristics. It is not difficult to extend the results to the more complex case of MIMO systems and discuss the associated characteristics and trade-offs. The tracking and disturbance rejection of MIMO systems can be analyzed via the configuration of Fig. 5.6. The output $C(s)$ in terms of reference input $R(s)$ and disturbance input $D(s)$ can be expressed as
\[
C(s) = G_p(s)G_c(s)[I + G_p(s)G_c(s)]^{-1}R(s) + [I + G_p(s)G_c(s)]^{-1}D(s) \\
= T(s)R(s) + S(s)D(s)
\]

Where the sensitivity function \( S(s) \) and the transfer function of the closed-loop system in the absence of disturbance, also known as complementary sensitivity function, \( T(s) \) satisfy the identity

\[
S(j\omega) + T(j\omega) = I
\]

This establishes the trade-off in the multivariable case. Note that if the measurement noise \( N(s) \) is included in the feedback path of Fig. 5.6, then we have an extra term—\( T(s)N(s) \) in the output equation. As in the SISO case, the sensitivity reduction and disturbance rejection are quantified by \( S \) and sensor noise rejection and robustness to high frequency uncertainty are quantified by \( T \).

There is only one important difference when comparing the two cases. Let us define open-loop transfer function and return difference matrices by \( L(s) \triangleq G_p(s)G_c(s) \) and \( F(s) = I + L(s) \), respectively. Then, good tracking and disturbance rejection in the specified bandwidth means

\[\begin{array}{ll}
\text{SISO} & \text{MIMO} \\
|L(j\omega)| > 1 & \sigma[L(j\omega)] > 1 \\
|F(j\omega)| > 1 & \sigma[F(j\omega)] > 1 \\
|S(j\omega)| < 1 & \sigma[S(j\omega)] < 1 \\
|T(j\omega)| = 1 & \sigma[T(j\omega)] = 1, \text{ or } T(j\omega) \approx I
\end{array}\]

The difference is identified by minimum and maximum singular values \( \sigma \) and \( \overline{\sigma} \) introduced for the MIMO case. Recall that the largest singular value of a matrix is defined by the Euclidean norm of that matrix and the smallest singular value becomes one over the norm difference of the inverse matrix. On the other hand, reducing sensor noise response and improved robustness at high frequency means

\[\begin{array}{ll}
\text{SISO} & \text{MIMO} \\
|L(j\omega)| < 1 & \sigma[L(j\omega)] < 1 \\
|F(j\omega)| \approx 1 & \sigma[F(j\omega)] \approx 1 \\
|S(j\omega)| \approx 1 & \sigma[S(j\omega)] \approx 1 \text{ or } S(j\omega) \approx I \\
|T(j\omega)| < 1 & \sigma[T(j\omega)] < 1
\end{array}\]
Therefore, keep in mind these requirements when designing control systems to achieve tracking and disturbance rejection. Since state space and frequency domain methods enhance and complement each other, it is interesting to see how the state-space technique can be used to fulfill the aforementioned design requirements. Thus, the main goal of this subsection is to provide a tracking system design which uses a combination of state feedback and output feedback. Before proceeding, let's discuss some aspects of tracking system design.

Let a system with equal number of inputs and outputs be represented by
\[
\dot{x} = Ax + Bu \quad x(t_o) = x_o \tag{5.67}
\]
\[
y = Cx \tag{5.68}
\]
Then we apply the control law
\[
u = u_F + u_K \tag{5.69}
\]
with
\[
u_F = Fv \tag{5.70}
\]
\[
u_K = Kx \tag{5.71}
\]
so that the closed-loop system of Fig. 5.7 is constructed. It is clear that there are two inputs acting on the system: the initial state \(x_o\) and the reference variable \(v\). Suppose the nonzero initial condition \(x_o\) arises from some disturbance and feedback \(u_K = Kx\) is used to restore the state to zero at a rate determined by the eigenvalues of \(A + BK\). Here, the state feedback is used for the so-called regulation problem rather than mere stabilization. Rapid decay can be obtained by moving the eigenvalues far from the \(j\omega\)-axis which requires large \(K\) and consequently, large control energy, increased bandwidth, and high sensitivity to noise, and so on. Thus, there should be a trade-off between these conflicting goals and one specific way of doing this will be discussed in Chap. 6 as previously mentioned.

In the regulator problem, the goal is to return the state to zero, and therefore, the external input \(v(t)\) is taken to be zero. Now, suppose that in addition to regulation, we wish the output \(y(t)\) to track the constant reference signal \(v(t) = V\). This can be

---

**Figure 5.7** Regulation and tracking.
achieved by using the other portion of control law \( u_F = Fv \) as follows. We assume \( x_0 = 0 \) and obtain the transfer function from \( u_F \) to \( y \) as

\[
Y(s) = G_K(s) \ U_F(s)
\]  

(5.72)

where

\[
G_K(s) = C [sI - A - BK]^{-1} B
\]

We also have

\[
U_F(s) = F \ V(s)
\]  

(5.73)

and using the final value theorem of Laplace transform the steady state form of the Eqs. (5.72) and (5.73) uniquely determine \( F \) as

\[
F = G_K^{-1}(0)
\]  

(5.74)

provided \( G_K(0) = -C(A + BK)^{-1}B \) is invertible. Finally, the overall control law and the responses of the system can be obtained by using superposition. There are several different ways by which one can arrive to the same result (see Prob. 5.11). Note that if the reference signal is not constant, then one should consider the generalization of Eq. (5.73), namely \( U_F(s) = F(s)V(s) \), where \( F(s) \) denotes the transfer function of the filter. Upon substitution of \( U_F(s) \) in Eq. (5.72), we obtain \( Y(s) = G_K(s)F(s)V(s) \) and in order \( Y(s) \) to follow \( V(s) \), the product \( G_K(s) \ F(s) \) must be set equal to identity which reveals that \( F(s) = G_K^{-1}(s) \). Thus, the problem reduces to realization of transfer function \( G_K^{-1}(s) \), which requires minimum phase assumption of the system.

So far we did not mention anything about disturbances. If we are dealing with short time impulsive disturbances, then we can interpret them as initial conditions and apply the previous methodology. Whenever long-term disturbances are present, we have to take a different approach. Suppose that in our model a constant, but unknown, disturbance term is included, that is,

\[
\dot{x}(t) = A \ x(t) + B \ u(t) + E \ d(t) \quad x(0) = x_0
\]  

(5.75)

\[
y(t) = C \ x(t)
\]  

(5.76)

where \( d(t) \) is an \( r \times 1 \) disturbance vector. If we use state feedback \( u = Kx \) to stabilize the system, then the presence of \( d \) yields a nonzero steady-state value. Although this can be reduced by increasing \( K \), one has a certain limitation because of saturation and noise effect.

One approach to overcome the noise effect is to estimate the unknown \( d \) and use this estimate to cancel out the disturbance (see Prob. 5.13). Here, we may note that the effects of constant disturbance vectors can often be eliminated by using the integral error feedback as we used it successfully in conjunction with pole placement by dynamic (two and three terms) output feedback controller. Thus, we introduce an additional state variable
and use the feedback

\[ u = Kx + Gz \]  \hspace{1cm} (5.78)

The augmented closed-loop system becomes

\[
\begin{bmatrix}
\dot{x} \\
\dot{z}
\end{bmatrix} =
\begin{bmatrix}
A + BK & BG \\
C & 0
\end{bmatrix}
\begin{bmatrix}
x \\
z
\end{bmatrix}
+ \begin{bmatrix}
E \\
0
\end{bmatrix} d
\]  \hspace{1cm} (5.79)

and the unknown matrices \( K \) and \( G \) are chosen to make the system stable. Then the steady-state value of \( y \) will be zero, since the lower half of Eq. (5.79) gives \( 0 = Cx(\infty) = y(\infty) \). It is also worth noting that by using a command input in addition to integral feedback, we can obtain a desired nonzero set point.

In the above discussion, we assumed that the unknown disturbance to be constant. This assumption was important to set up the problem and provide a solution for it. Generally speaking, one needs at least some information about the unknown disturbance before carrying out any type of design for the system. Let us consider another situation, more often encountered in practice, where the disturbance vector \( d \) can be described as the output of a system excited by new initial conditions. This is equivalent to the assumption that

\[ \dot{z} = Lz \quad z(0) = z_0 \]  \hspace{1cm} (5.80)

\[ d = Mz \]  \hspace{1cm} (5.81)

Combining Eqs. (5.80) and (5.81) with Eqs. (5.75) and (5.76), we obtain

\[
\begin{bmatrix}
\dot{x} \\
\dot{z}
\end{bmatrix} =
\begin{bmatrix}
A & EM \\
0 & L
\end{bmatrix}
\begin{bmatrix}
x \\
z
\end{bmatrix}
+ \begin{bmatrix}
B \\
0
\end{bmatrix} u
\]  \hspace{1cm} (5.82)

\[ y = [C \quad 0]
\begin{bmatrix}
x \\
z
\end{bmatrix}
\]  \hspace{1cm} (5.83)

Now Eqs. (5.82) and (5.83) can be treated as a new system with the same input and output and apply the design appropriately. Again, a reasonable approach might be to attempt to estimate the unknown \( d \). An observer for Eqs. (5.82) and (5.83) provides an estimate for \([\hat{x} \quad \hat{z}]^T\) which in turn gives the estimate \( \hat{d} = \hat{M}\hat{z} \), then by assuming that \( B = EM \), we can set \( u = -\hat{d} \) to cancel out the disturbance. It is interesting to investigate the equivalence between this method and the integral feedback (see Prob. 5.13).

Let us continue our discussion with an alternative problem in which \( d(t) \) in Eq. (5.75) is defined as a constant vector whose elements are composed of input signals and disturbances. The objective of the design is to find feedback control law from state variables as well as output variables such that certain states are driven to any desired state asymptotically while the overall system is asymptotically stable. For this problem, it is convenient to define the output equation as
instead of Eq. (5.76) with $y = \dot{z}$ as defined by Eq. (5.77). The Eq. (5.84) means that, for example, to drive the state $x_1$ to the set point $v$, is equivalent to driving $y = d_1 - x_1$ to zero as $s$ approaches infinity, where $d_1 = v = v_d =$ constant. Thus, the design problem may be regarded as output regulation in this stage. The overall control law may be written as

$$u = Kx + G \int y \, dt \quad (5.85)$$

and one should obtain the unknown gain matrices $K$ and $G$. Example 5.6 illustrates application of the design with state feedback and output integral control.

Example 5.6

Consider the second order system

$$\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} =
\begin{bmatrix}
-2 & 1 & 0 \\
0 & -1 & 1 \\
-k_2 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} +
\begin{bmatrix}
0 \\
1 \\
0
\end{bmatrix} u +
\begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
d_1 \\
d_2
\end{bmatrix}$$

where $d_2$ and $d_3$ are unknown constant disturbance signals. It is desired that the state $x_1$ follows a reference input $d_1 =$ constant. Design a control with state and dynamic feedback such that all the roots of the characteristic equation of the closed-loop system are at $-3$, and $\lim x_1(t) = d_1$. Sketch $x_1(t)$ for $t \geq 0$ for $w_1 = 1$, $d_2 = 1$, $d_3 = 1$ and with zero initial conditions.

Solution Let $u = k_1x_1 + k_2x_2 + g \int (w_1 - x_1) \, dt$, and with $\dot{z} = y = w_1 - x_1$ define for convenience $\dot{x}_3 = g\dot{z}$. Then the state equation of the closed-loop system can be written as

$$\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} =
\begin{bmatrix}
-2 & 1 & 0 \\
0 & k_1 & k_2 - 1 \\
-g & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} +
\begin{bmatrix}
0 & 0 & 1 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
w_1 \\
w_2 \\
w_3
\end{bmatrix}$$

The characteristic equation is

$$s^3 + (3 - k_2)s^2 + (2 - 2k_2 - k_1)s + g = 0$$

and the desired characteristic equation is

$$s^3 + 9s^2 + 27s + 27 = 0$$

Thus, $k_1 = -13$, $k_2 = -6$, $g = 27$. The output transform is

$$X_1(s) = \frac{gD_1(s) + sD_2(s) + s(s - k_2 + 1)D_3(s)}{s^3 + 9s^2 + 27s + 27}$$

and for $D_1(s) = D_2(s) = D_3(s) = 1/s$, it is easy to see that

$$\lim_{t \to \infty} x_1(t) = \lim_{s \to \infty} sX_1(s) = D_1$$

Figure 5.8 shows the step response of the system.

We close this subsection by presenting a tracking system design in a general setting (Porter and Bradshaw (1974,1978) and Bradshaw et al. (1978)). Consider the controllable system
\[
\dot{x} = Ax + Bu \\
y = Cx = \begin{bmatrix} E \\ F \end{bmatrix} x
\]

where \( y_p = E x \) is a \( p \times 1 \) vector representing the outputs which are required to follow a \( p \times 1 \) dimensional vector \( v \) asymptotically. The design consists of augmenting the equation

\[
\dot{z} = v - y_p = v - Ex
\]

with Eqs. (5.86) and (5.87) as

\[
\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -A & 0 \\ -E & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ I \end{bmatrix} v
\]

\[
y = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}
\]

and applying the control law

\[
u = K_1 x + K_2 z = [K_1 \ K_2] \begin{bmatrix} x \\ z \end{bmatrix}
\]
The control law assigns the desired closed-loop eigenvalue spectrum if and only if the augmented system is controllable or the following equivalent condition is satisfied

\[ \rho \begin{bmatrix} B & A \\ 0 & -E \end{bmatrix} = n + p \]  \hspace{1cm} (5.92)

The feedback gain \( K = [K_1, K_2] \) can be obtained by using any one of the pole placement methods described in the previous chapter, and the closed-loop system becomes

\[ \begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} A + BK_1 & BK_2 \\ -E & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} v \]  \hspace{1cm} (5.93)

Then since the inputs are constants, the steady-state values of the states \( x \) and \( z \) are also constants. Therefore, \( \dot{z} = 0 \) in the steady state and Eq. (5.88) indicates that

\[ \lim_{t \to \infty} y_p(t) = v(t) \]  \hspace{1cm} (5.94)

Thus the outputs \( y_p(t) \) track the piecewise constant command vector \( v(t) \) in the steady state.

In this tracking system design, the outputs \( y(t) \) effectively follow constant input commands \( v(t) \) in the steady state. The outputs are also decoupled and noninteracting so that each output follows only its respective input. The design method requires, however, that all the states be fed back as part of the procedure. This is shown in Fig. 5.9. If it is not feasible to physically include sensors in order to measure all of the state variables, then one should use an observer to estimate the state. The direct use of the output measurements to produce the input signal to the controller avoids the requirement for measuring or reconstructing the entire state vector. This was used in the pole placement design of previous sections and can certainly be used for tracking system design as well. In fact, based on a high gain approach (Porter and Bradshaw (1979, 1981)) the closed-loop system exhibits a distinctive asymptotic structure in which there are slow and fast modes. The slow modes are asymptotically uncontrollable or unobservable and thus the output response is dominated by the fast modes. This

Figure 5.9  Tracking System design.
leads to very fast tracking of the command input by the output. (See also D’Azzo and Houpis 1988.)

5.4.2 Robust Control Design for Tracking and Disturbance Rejection

In this subsection, we shall discuss the design of robust control systems to achieve asymptotic tracking and disturbance rejection by using state variable techniques. We assume that the system is controllable and observable. Before providing the design method in general terms, let’s discuss the nature of the reference and disturbance signals \( r(t) \) and \( d(t) \) and provide preliminary results to justify the robustness of the design. Consider the scalar version of Fig. 5.6 and assume the reference and disturbance signals be represented by \( R(s) = \mathcal{L} [r(t)] = N_r(s)/D_r(s) \) and \( D(s) = \mathcal{L} [d(t)] = N_d(s)/D_d(s) \). This is equivalent to the assumption that \( r(t) \) and \( d(t) \) are generated by

\[
\dot{x}_r = A_r x_r \quad (5.95a)
\]

\[
r(t) = c_r x_r \quad (5.95b)
\]

and

\[
\dot{x}_d = A_d x_d \quad (5.96a)
\]

\[
d(t) = c_d x_d \quad (5.96b)
\]

with unknown initial states \( x_r(0) \) and \( x_d(0) \).

The parts of \( r(t) \) and \( d(t) \) which go to zero as \( t \to \infty \) have no effect on \( y(t) \) as \( t \to \infty \); hence, it is reasonable to assume that some roots of \( D_r(s) \) and \( D_d(s) \) have zero or positive real parts. Let the least common denominator of the unstable poles of \( R(s) \) and \( D(s) \) be denoted by a polynomial \( Q(s) \) of degree \( q \). Then it can be shown that there exists a compensator which asymptotically stabilizes the feedback system and achieves asymptotic tracking and disturbance rejection, provided that no root of \( Q(s) \) is a zero of \( G_p(s) \). Under this condition, the cascade connection of the system with transfer function \( 1/Q(s) \) followed by \( G_p(s) = N_p(s)/D_p(s) \) is controllable and observable. Consequently, the polynomials \( N_p(s) \) and \( D_p(s)Q(s) \) do not have common roots and it is not difficult to verify that the compensator \( G_c(s) = N_c(s)/D_c(s)Q(s) \) of degree \( l = n + q - 1 \) will achieve the goals.

In order to see how the compensator is determined, we write the transfer function of the feedback system, assuming for a moment \( D(s) = 0 \), that is,

\[
\frac{N_f(s)}{D_f(s)} = \frac{N_c(s)N_p(s)}{D_c(s)D_p(s)Q(s) + N_c(s)N_p(s)} \quad (5.97)
\]

Now, for any desired \( D_f(s) \) of degree \( n + l \) there exists a solution to the polynomial equation

\[
D_f(s) = D_c(s)D_p(s)Q(s) + N_c(s)N_p(s) \quad (5.98)
\]
if and only if \( l \geq n - 1 \). Consequently, this polynomial equation, also known as Diophantine equation, can be solved for \( N_c(s) \) and \( D_c(s) \) via coefficient matching.

Let \( y_r(t) \) denote the output excited by \( r(t) \) and \( d(t) = 0 \). Then asymptotic tracking can be checked by

\[
R(s) - Y_r(s) = \left( 1 - \frac{N_c(s)N_p(s)}{D_c(s)D_p(s)Q(s)} \right) R(s) \frac{D_c(s)D_p(s)N_r(s)}{D_c(s)D_p(s)Q(s) + N_c(s)N_p(s)} . Q(s) \]

(5.99)

Since all unstable roots of \( D_r(s) \) are cancelled by \( Q(s) \), all the poles of \( R(s) - Y_r(s) \) have negative real parts, and we have \( r(t) - y_r(t) \to 0 \) as \( t \to \infty \).

Similarly, the output \( y_d(t) \), excited by \( d(t) \) and \( r(t) = 0 \), is given by

\[
Y_d(s) = \frac{D_c(s)D_p(s)Q(s)}{D_c(s)D_p(s)Q(s) + D_c(s)D_p(s)} \cdot D(s) \]

(5.100)

and using similar reasoning we have \( y_d(t) = -e_d(t) \to 0 \) as \( t \to \infty \). Thus, the disturbance rejection is also guaranteed and we have \( y(t) = y_r(t) + y_d(t) \to 0 \) as \( t \to \infty \).

This design philosophy is based on the duplication of the dynamic or model, \( 1/Q(s) \), inside the loop, which is often referred to as the internal model principle. Any design based on this principle is deemed robust. It is well known that in classical feedback design, in order to have a zero steady-state error for a step reference input, the plant transfer function must be of type 1; that is, \( G_p(s) \) has one pole at the origin. In this case, \( Q(s) = s \) and \( [1/Q(s)]G_p(s) \) is of type 1 transfer function. Although one can generalize these results to the multivariable case using transfer function approach (Desoer and Wang (1980)), we shall provide its equivalent state-space technique. (See also Davison (1977) and Chen (1984) for a more detailed discussion.)

Consider a controllable and observable system described by

\[
\dot{x} = Ax + Bu + Ed(t) \quad (5.101) \\
y = Cx + Du + Hd(t) \quad (5.102)
\]

where the dimensions of variables are defined as before. It is assumed that the disturbance vector \( d(t) \) and the reference vector \( r(t) \) are generated by

\[
\dot{x}_d = A_d x_d \quad (5.103a) \\
d(t) = C_d x_d \quad (5.103b)
\]
and

\[ \dot{x}_r = A_r x_r \]  \hspace{1cm} (5.104a)

\[ r(t) = C_r x_r \]  \hspace{1cm} (5.104b)

with unknown initial states \( x_d(0) \) and \( x_r(0) \). Let the minimal polynomials of \( A_d \) and \( A_r \) be \( Q_d(s) \) and \( Q_r(s) \), respectively, and let

\[ Q(s) = s^q + \alpha_{q-1} s^{q-1} + \alpha_{q-2} s^{q-2} + \cdots + \alpha_0 \] \hspace{1cm} (5.105)

be the least common multiple of the closed right-half \( s \)-plane roots of \( Q_d(s) \) and \( Q_r(s) \). The internal model \( Q^{-1}(s) y_r \) can be realized as

\[ \dot{x}_c = A_c x_c + B_c e(t) \] \hspace{1cm} (5.106)

\[ y_c = x_c \] \hspace{1cm} (5.107)

where

\[ A_c = \text{block diag} \{ \bar{A}, \bar{A}, \ldots, \bar{A} \} \]

\[ B_c = \text{block diag} \{ \bar{b}, \bar{b}, \ldots, \bar{b} \} \]

consist of \( r \) blocks each with

\[
\bar{A} = \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-\alpha_0 & -\alpha_1 & -\alpha_2 & \cdots & -\alpha_{q-1}
\end{bmatrix} \quad \bar{b} = \begin{bmatrix}
0 \\
0 \\
\vdots \\
0 \\
1
\end{bmatrix}
\]

and \( e(t) = r(t) - y(t) \) denotes the tracking error.

The tandem connection of the system and the internal model compensator can be represented by

\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_c
\end{bmatrix} = \begin{bmatrix}
A & 0 \\
-B_c C & A_c
\end{bmatrix} \begin{bmatrix}
x \\
x_c
\end{bmatrix} + \begin{bmatrix}
B \\
-B_c D
\end{bmatrix} u \\
+ \begin{bmatrix}
E \\
-B_c H
\end{bmatrix} d(t) + \begin{bmatrix}
0 \\
0
\end{bmatrix} r(t)
\] \hspace{1cm} (5.108)

The composite system Eq. (5.108) is controllable and observable, if and only if \( m \geq r \) and no root of \( Q(s) \) is a transmission zero of the system. Equivalently, Eq. (5.108) is controllable if

\[ \rho \begin{bmatrix}
\lambda I - A & B \\
-C & D
\end{bmatrix} = n + r \quad \text{for every root } \lambda \text{ of } Q(s) \] \hspace{1cm} (5.109)

Under the controllability assumption of Eq. (5.109), the eigenvalues of the composite system can be arbitrarily assigned by the state feedback
\[ u = [K \ K_c] \begin{bmatrix} x \\ x_c \end{bmatrix} = Kx + K_c x_c \] (5.110)

If the state of the system is not available, we can design an observer to estimate the state and then implement the feedback law. This design method is robust and achieves asymptotic tracking and disturbance rejection.

**Example 5.7**

Consider the system Eqs. (5.101) and (5.102) with

\[ A = \begin{bmatrix} 0 & 1 \\ 2 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad E = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad D = 0, H = 0 \]

It is desired to design a robust controller.

**Solution** Let the disturbance and reference vectors be constant. Then \( Q(s) = s^2 \) and

\[ A_c = \tilde{A} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, B_c = \tilde{b} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \]

The composite system Eq. (5.108) becomes

\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_c
\end{bmatrix} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
2 & -2 & 0 & 0 \\
0 & 0 & 0 & 1 \\
-1 & 0 & 0 & 0
\end{bmatrix} \begin{bmatrix}
x \\
x_c
\end{bmatrix} + \begin{bmatrix}
0 \\
1 \\
0 \\
0
\end{bmatrix} u + \begin{bmatrix}
0 \\
1 \\
0 \\
0
\end{bmatrix} d + \begin{bmatrix}
0 \\
0 \\
0 \\
1
\end{bmatrix} r
\]

\[ y = [1 \ 0 \ 0 \ 0] \begin{bmatrix} x \\ x_c \end{bmatrix} \]

and its eigenvalues can be assigned to \(-1, -2, -3, -4\) by

\[ u = [k \ k_c] \begin{bmatrix} x \\ x_c \end{bmatrix} = \begin{bmatrix} -37 & -8 & 24 & 50 \end{bmatrix} \begin{bmatrix} x \\ x_c \end{bmatrix} \]

The step response of the system is obtained using MATRIXx and is shown in Fig. 5.10. This completes the robust design to achieve asymptotic tracking and disturbance rejection. In this design, the parameter perturbation even large perturbation of plant is permitted as long as the composite system remains stable.

**5.4.3 Dynamic Compensator and Static Output Feedback**

Let the system be represented again by Eqs. (5.1) and (5.2) with \( D = 0 \). Then we define a dynamic compensator described by

\[
\begin{align*}
\dot{z}(t) &= Fz(t) + Gy(t) \\
s(t) &= Mz(t) + Ny(t)
\end{align*}
\] (5.111) (5.112)
where \( z(t) \) is the compensator state vector of order \( l \) and \( s(t) \) is the compensator output vector of order \( m \). The matrices \( F, G, M, \) and \( N \) are real constant with dimensions \( l \times l, l \times m, m \times l, \) and \( m \times m \), respectively. The feedback control law for the system is

\[
u(t) = v(t) + s(t)
\]  

(5.113)

and the state equation of the compensated system (see Fig. 5.11) becomes

\[
\dot{x}_c(t) = A_c x_c(t) + B_c v(t)
\]

(5.114)

where \( x_c(t) = [x^T(t) \ z^T(t)]^T \) and

\[
A_c = \begin{bmatrix} A + BNC & BM \\ GC & F \end{bmatrix}, \quad B_c = \begin{bmatrix} B \\ 0 \end{bmatrix}
\]

(5.115)

The output equation of the compensated system remains the same as the original uncompensated system that is, \( y(i) = C x(i) \). The overall system is of order \( n + l \), where the dimension of the compensator \( l \) is preferred to be as small as possible. The closed-loop system matrix \( A_c \) may be decomposed as follows

\[
A_c = \begin{bmatrix} A + BNC & BM \\ GC & F \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} B & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} N & M \\ G & F \end{bmatrix} \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}
\]

\[
A_c = \hat{A} + \hat{B} + \hat{K} + \hat{C}
\]

(5.116)
which has a similar structure of an auxiliary system (\(\hat{A}, \hat{B}, \hat{C}\)) with a static output feedback matrix \(\hat{K}\). Now, we can apply one of the methods described in Sec. 5.2, to obtain \(\hat{K}\). The submatrices of \(\hat{K}\) specify the required compensator matrices \(F, G, M,\) and \(N\). In particular, let us consider the iterative method of Sec. 5.2, namely Algorithm 5.2, which can be used in this connection. Suppose the rank condition Eq. (5.24) for the original system is not satisfied. Then by assuming that the original system is controllable and observable, we can use a dynamic compensator of the form Eqs. (5.111) and (5.112) for pole assignment. The equivalent static output feedback Eq. (5.116) can be set up and the condition Eq. (5.24) should be applied to the modified matrices \(\hat{A}, \hat{B},\) and \(\hat{C}\). As a by-product of the iterative method, an interesting result is deduced which shows that a necessary condition for pole assignment in this case is \((m + l)(r + l) \geq (n + l)\). Thus the order of the required compensator is obtained from

\[
l > \frac{1}{2} [\sqrt{(m + r - 1)^2 + 4(n - mr)} - (m + r - 1)]
\]  

(5.117)

The order of the dynamic compensator obtained from Eq. (5.117) is in general less than the order required by other existing methods.

**PROBLEMS**

5.1 Can the observability of a minimal system be affected by output feedback \(u = v + Ky\)? Investigate this for single-input single-output system and compare it with the state feedback case.
5.2 Consider the system

\[
\begin{bmatrix}
0 & 1 & 0 \\
-2 & 3 & 0 \\
5 & 1 & 3
\end{bmatrix} x + \begin{bmatrix}
0 & 0 \\
1 & 3 \\
0 & 1
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
0 & 0 & 7 \\
7 & 9 & 0
\end{bmatrix} x
\]

Is it possible to obtain an output feedback control law to shift the eigenvalues to \(-3, -3, -4\) through the extension of state feedback? If yes, find it; if no, why not?

5.3 Consider the stick balancing Prob. 4.23. Can the system be stabilized by proportional output feedback?

5.4 Use Algorithm 5.1 to solve Prob. 5.2. Check your result with a command like "PROP" available in CONTROL.lab.

5.5 A controllable and observable system is given by

\[
\begin{bmatrix}
0 & 1 & 0 & 0 \\
-2 & 3 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 12 & -1
\end{bmatrix} x + \begin{bmatrix}
0 & 0 \\
1 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix} x
\]

Show that it is only possible to assign three out of four eigenvalues to the desired locations, say \(-1, -2, -3\), by using proportional output feedback. Obtain the required output feedback gain \(K\).

5.6 Consider the system of Ex. 5.1 with deleted last column of the matrix \(B\), as

\[
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-1 & 0 & 0 & 0
\end{bmatrix} x + \begin{bmatrix}
1 & 0 \\
0 & 0 \\
0 & 1 \\
0 & 0
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0
\end{bmatrix} x
\]

It is required to assign all eigenvalues of this system with the desired location given by \(-1, -2, -3, -4\). Show that the iterative Algorithm 5.2 can achieve this goal and obtain the output feedback gain \(K\).

5.7 CAD Problem. Let the dynamical equation of a system be represented by

\[
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & -1 & 1 & 0 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0
\end{bmatrix} x + \begin{bmatrix}
0 & 0 \\
1 & 0 \\
0 & 0 \\
0 & 1
\end{bmatrix} u
\]

\[
y = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\]
Use proportional plus derivative output feedback controller such that the closed-loop system poles are at $-1$, $-1$, $-2$, $-2$, $-3$. You may use the primitive "PRDR" of CONTROL.lab or other CAD programs to solve this problem.

5.8 Consider the system

$$
\dot{x} = \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
10 & 2 & -9 & -2 \\
\end{bmatrix}
x + \begin{bmatrix}
0 & 1 \\
0 & 0 \\
0 & 2 \\
1 & 0 \\
\end{bmatrix}
u + \begin{bmatrix}
-1 \\
0 \\
0 \\
0 \\
\end{bmatrix}d
$$

$$
y = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
\end{bmatrix}x
$$

a. Check the controllability, observability, and cyclicity of the system.

b. If possible, design a proportional plus integral output feedback controller such that the closed-loop poles of the system are at $-1 \pm j$, $-3$, $-4$, $-4$, $-5$.

c. Repeat as in part b, but use a PID controller. Compare the transient and steady-state characteristics of the system with the one obtained in part b.

5.9 CAD Problem. Consider the turbojet gasturbine system represented by the following linearized perturbation model.

$$
\dot{x} = \begin{bmatrix}
-1.268 & -0.04528 & 1.498 & 951.5 \\
1.002 & -1.957 & 8.52 & 1240 \\
0 & 0 & -10 & 0 \\
0 & 0 & 0 & -100 \\
\end{bmatrix}
x + \begin{bmatrix}
0 & 0 \\
0 & 0 \\
10 & 0 \\
0 & 100 \\
\end{bmatrix}u + \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
\end{bmatrix}d
$$

$$
y = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
\end{bmatrix}x
$$

where the state variables $x_1$, $x_2$ are high and low pressure spool speed, $x_3$ is the jet pipe nozzle area, and $x_4$ is the fuel flow rate. The control inputs $u_1$, $u_2$ are demanded jet pipe nozzle area and fuel flow rate, respectively. Furthermore, the disturbance $d$ represents the change of altitude.

a. Design a PID controller such that in the closed loop system for a unit step change at $v_1$ (or $v_2$) the corresponding output $y_1$ (or $y_2$) exhibits a fast transient response lasting less than 2 seconds and changes by unity in the steady state and the steady state value of $y_2$ (or $y_1$) remains unaltered. Furthermore, a step change at $d$ does not affect the steady-state values of $y_1$ and $y_2$. On this basis of the specified transient duration, the desired pole positions are chosen to be $-4$, $-4.2$, $-5$, $-5.2$, $-40$, $-60$.

b. Use the primitive "PID" available in CONTROL.lab to obtain the three term parameters. Also plot various step responses before and after the design.

5.10 The dynamical equation description of a system is given by

$$
\dot{x} = \begin{bmatrix}
0 & 1 & 0 & 0 \\
2.125 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 \\
0 & -1 & -1 & 0 \\
\end{bmatrix}x + \begin{bmatrix}
0 \\
0 \\
0 \\
1 \\
\end{bmatrix}u
$$

$$
y = \begin{bmatrix}
0 & 0 & 1 & 0 \\
\end{bmatrix}x
$$
a. Referring to the discussion in Sec. 5.4.1, design a feedback controller such that the output is reached at the desired value \( y_d \) in steady state by a proper nonzero set point \( y_d \) and at the same time the closed-loop system poles are placed at \(-1.5, -2, \) and \(-1.5 \pm 1.5j\).

b. Replace the output equation by \( y = x_1 \) and repeat as in part a, if possible.

5.11 Consider a controllable and observable SISO system \( \{A, b, c, d\} \) with the proper transfer function \( g(s), g(0) \neq 0 \).

a. Show that the state feedback control law \( u = f + kx \) with \( f = 1/g_d(0) \), where \( g_d(0) \) is the closed-loop transfer function evaluated at \( s = 0 \), and a proper choice of feedback gain vector \( k \) allows the asymptotic tracking of \( v(t) = V \).

b. Prove that if the input \( u(t) \) is of the form \( e^{\lambda t} \), where \( \lambda \) is not a pole of \( g(s) \), then the output due to the initial state \( x(0) = - (A - \lambda I)^{-1} b \) and the specified input is equal to \( y(t) = g(\lambda)e^{\lambda t} \) for \( t \geq 0 \).

c. Investigate the generalization of parts a and b for MIMO systems.

5.12 Consider the helicopter Prob. 4.4 with a constant (but unknown) headwind \( d \), so that the equations of motion have an added term on the right hand side \([-0.02, -0.01]d \). Design a velocity-command controller that incorporates integral error feedback and results in a stable closed-loop system with eigenvalues at \(-1, -2, -1 \pm j\).

5.13 For a SISO system driven by a constant unknown disturbance term \( bd \), that is \( \dot{x} = Ax + Bu + bd \), \( y = Cx \), design an observer to estimate \( d \), and use this to compensate for the disturbances. Investigate the equivalence between this method and the integral feedback.

5.14 The dynamical equation description of a dc motor is given by

\[
\dot{x} = Ax + Bu + Ed
\]

\[
y = Cx + Hd
\]

where

\[
A = \begin{bmatrix} -K_v & K_m \\ -K_b & -R_a \\ J & L_a \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 \\ 1 \\ L_a \end{bmatrix}, \quad H = \begin{bmatrix} 0 \\ 1 \\ L_a \end{bmatrix}, \quad E = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}
\]

with the state variables \( x = [x_1, x_2]^T = [\omega, i_a]^T \) defined as motor angular velocity and armature current, the control variable \( u = v_a \) as armature, applied voltage, the vector \( d = [d_1, d_2]^T = [T_i, v]^T \) consisting of constant load torque and set point. Furthermore, the parameters \( K_v, K_m, K_b, R_a, L_a, \) and \( J \) are viscous friction coefficient, motor torque constant, back emf constant, armature resistance, armature inductance, and moment of inertia, respectively. Design the control law \( u = k_1x_1 + k_2x_2 + k_3 \int y(t) \, dt \) such that
lim \( x_1 = r \), \( \lim_{t \to \infty} \dot{x}_1 = 0 \), \( \lim_{t \to \infty} \dot{x}_2 = 0 \), and at the same time the closed-loop system poles be placed at \(-300, -10 \pm j\) to guarantee stability. Plot the state response of the system for \( T_i = 1 \), \( v = 1 \) and \( T_f = 0 \), \( v = 1 \).

5.15 Consider the first-order system

\[
\dot{x} = -x + u + d_2
\]

where \( u(t) \) is the control and \( d_2(t) \) is an unknown constant disturbance. It is desired that the state follows a reference input \( d_1 \) = constant. Design a control law with state and dynamic feedback such that the roots of the characteristics equation of the closed-loop system are at \(-1\), and

\[
\lim_{t \to \infty} x(t) = d_1
\]

Sketch \( x(t) \) for \( t \geq 0 \) for \( d_1 = 1 \), \( d_2 = -1 \) and with zero initial conditions.

5.16 Consider the general controller structure of Fig. P5.16 which summarizes our discussion on stabilization, tracking, and disturbance rejection of Subsec. 5.4.1.

a. Define all the input-output variables and examine the purpose of the three transmission paths of the controller structure.

b. Assume that a measurable disturbance \( d \) influences the plant through a known matrix \( E \) as

\[
\dot{x} = Ax + Bu + Ed
\]

Let the control variable \( u \) consist of two parts \( u_f \) and \( u_d \), representing feedback and disturbance control signals, respectively. Show that a suitable \( u_d \) which counteracts the disturbance is given by

\[
u_d = -(B^TB)^{-1}B^TEd\]

![Figure P5.16 System of Problem 5.16.](image-url)
5.17 Consider the system

\[ \dot{x} = Ax + Bu + Ed(t) \]

\[ y = Cx \]

and let the models of the disturbance and reference signals be

\[ \dot{x}_d = 0, \quad d(t) = x_d \]

\[ \dot{x}_r = 0, \quad r(t) = x_r \]

a. Show that the control law which stabilizes the system and makes \( y \) to follow \( r \) in the presence of the constant disturbance \( d \) is given by

\[ u(t) = K_1x(t) + K_2 \int (y - r) \, dt + \text{constant} \]

b. How should the constant term of the control \( u(t) \) be chosen if the integral is taken from \( \tau = 0 \)? Assume \( x(0) = 0 \) and \( u(0) \) as the steady-state value.

c. How should the control law be modified, when the estimate of the state \( \hat{x} \) from an observer is used instead of the true state? Incorporate any necessary changes in constant disturbance signal.

d. Assuming

\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad E = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad x(0) = 0
\]

\[ c = [1 \quad 0] \]

verify that the control law \( u = -[2 \quad 2] x + \int (y - r) \, dt \) stabilizes the system and tracks the reference signal in approximately 10 sec. Perform your simulation with and without observer and compare the results.

5.18 Consider the system

\[ \dot{x} = Ax + Bu + Ex_d \]

It is desired that the state \( x \) tracks a reference state \( x_r \), and reject the disturbance \( x_d \). Let \( x_r \) and \( x_d \) satisfy the known differential equations \( \dot{x}_r = A_r x_r \), \( \dot{x}_d = A_d x_d \) and let the error \( e = x - x_r \) be measurable through the observation matrix \( C \). Since the steady-state condition is characterized by \( \dot{e} = 0 \), \( e \) can not be zero, instead, we require \( y = C e = 0 \).

a. Show that the required control law is given by

\[ u = -ke - k_r x_r - k_d x_d \]

where \( k \) is stabilizable gain and

\[(k_r, \quad k_d) = \left[C(A - Bk)^{-1}B\right]^{-1} C(A - Bk)^{-1} F\]

with \( F = [A - A_r \quad E] \).

b. Assuming \( A_r = 0, A_d = 0 \), the desired state \( x_r = [1 \quad 1]^T \) and

\[
A = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad E = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad C = [1 \quad 2] \]
Obtain the control law such that the closed-loop system has the desired poles at $-2$, $-3$ and tracking and disturbance rejection are achieved.

5.19 Consider the unstable system

$$
\dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 4 & 4 & -1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u \\

y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} x
$$

Use the tracking system design discussed in Subsec. 5.4.1 to track the command $r(t) = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$ by the output $y(t)$ and guarantee the stability of the closed-loop system by assigning the eigenvalues at $-2$, $-3$, $-4$, $-5$, $-6$.

5.20 Repeat robust tracking design of Example 5.7 when $d(t) = e'$.

5.21 Consider the following systems

a. A controllable and observable system.

$$
\dot{x} = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 2 & -1 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u \\
y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x
$$


$$
\dot{x} = \begin{bmatrix} -2.6 & 0.25 & -38 & 0 \\ -0.075 & -0.27 & 4.4 & 0 \\ 0.078 & -0.99 & -0.23 & 0.052 \\ 1 & 0.078 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 17 & 7 \\ 0.82 & -3.2 \\ 0 & 0.046 \\ 0 & 0 \end{bmatrix} u \\
y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x
$$

c. The symmetric vibration model of the standard Draper/RPL Satellite (Bhattacharyya 1987).

$$
\dot{x} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 14.8732 & 32.8086 & 0 & 0 & 0 \\ 0 & -146.702 & -7476.64 & 0 & 0 & 0 \\ 0 & -41.8468 & -2699.36 & 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -0.04168 & 0.23623 \\ 10.38611 & -25.647 \\ 3.725120 & -9.1629 \end{bmatrix} u \\
y = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} x
$$
which one of these systems can not be stabilized by static output feedback? Identify the system and design a dynamic compensator by using the method of Subsec. 5.4.3 such that the closed-loop system is stable (choose your own desired set of eigenvalues).

5.22 The following system can not be stabilized by static output feedback

\[
\dot{x} = \begin{bmatrix}
1 & 2 & 3 & 4 \\
-1 & -3 & 2 & 6 \\
0.1 & 0.2 & 3 & 6 \\
0.3 & 0.4 & 1 & 2 \\
\end{bmatrix} x + \begin{bmatrix}
1 \\
2 \\
3 \\
-1 \\
\end{bmatrix} u \\
y = \begin{bmatrix}
1 & 2 & -1 & 1 \\
-2 & 1 & 0 & -1 \\
\end{bmatrix} x
\]

Show that a first-order dynamic compensator can stabilize the system. Design the dynamic compensator based on its equivalent static output feedback problem such that the closed-loop system poles are places at \(-1, -2, -3, -4, -5\).
6 Optimal Control Design

6.1 INTRODUCTION

Optimal control problems have received a great deal of attention since the early 1960s. The demand for high performance on the one hand, and the advent of digital processors on the other hand, have made these problems somewhat attractive. Optimal control problems often comprise of finding a control (decision) rule, subject to certain constraints, which would minimize some measure of deviation from ideal behavior. Such a measure is commonly known as a performance criterion or performance index which is formulated by the control engineer. Performance index of an optimal control system is a measure or an indicator of the cumulative deviation of the system from a desired or ideal state. When the performance criterion involves the economic consequence of a given control, the index is called the cost function. It is also possible to define other measures of performance for the system. These can be "energy" for a power system, "fuel" for an aircraft, or "time" for a spacecraft to move in outer space. These are not the only possible measures; there are more possibilities in which the designer could use a combination of various measures and prioritize them in some fashion. It is imperative to note that the optimization process should provide not only control policies and system configurations that are optimal, but also a measure of degradation of the performance when the corresponding index departs from its desired minimum (or maximum) value. The latter may be caused as a result of a nonoptimal control application.
In any optimal control problem, there is a natural conflict between analytical feasibility and practical utility in the selection of the performance index. It would be ideal that practical considerations would determine the choice of performance index. However, a compromise is often made between a meaningful determination of system performance and a tractable mathematical problem. The exact definition of a performance index is a very difficult task, specially for complex systems. In general, the more experience and knowledge one may have of a system, the better the chance of defining a meaningful performance criterion. For example, in military or space applications, criteria such as minimum fuel expenditure, minimum target miss, and minimum time may be appropriate. In other applications, the prime considerations may be cost and economics.

In forming an optimal control problem, the quantities appearing in the optimization process are state variables, control variables, and system parameters. As an example, consider a vehicle traveling in space. The vehicle’s three position and velocity coordinates, and its instantaneous mass can be considered as state variables. The control variables of this system may be the thrust magnitudes and directional angles. The system parameters may be vehicle’s exhaust velocity of the propulsion and the size of the on-born power plant.

In summary, an optimal control problem can be formulated using the following information: (a) state and output equations, (b) control vector, (c) problem’s constraints, (d) performance index, and (e) system parameters. Therefore, an optimal control problem would be to find an optimal control vector within the class of allowable control vectors. This vector commonly depends on (a) initial state or output, (b) desired state or output, (c) nature of constraints, and (d) nature of the performance index. Except for a few special cases, the optimal control problem may be so complicated that the only viable solution is a numerical one. In sequel, a mathematical definition of an optimal control problem for both continuous- and discrete-time system will be given.

Consider a continuous-time system defined on a fixed interval $t_o \leq t \leq t_f$

$$\dot{x}(t) = f [x(t), u(t), t]$$  \hspace{1cm} (6.1)

with a fixed initial condition,

$$x(t_o) = x_o$$  \hspace{1cm} (6.2)

a set of allowable controls,

$$u(t) \in U$$  \hspace{1cm} (6.3)

where $\in$ denotes containment of member of a set; and a performance index (or objective function),

$$J_o[x(t_o), u(\cdot), t_o] = F [x(t_f)] + \int_{t_o}^{t_f} L[x(t), u(t)] \, dt.$$  \hspace{1cm} (6.4)

An optimal control $u^*(t)$ is sought which would satisfy the system dynamic constraints, that is, state Eq. (6.1), static constraints, that is, the initial condition Eq. (6.2) and
the feasible control set Eq. (6.3) that minimizes the cost function Eq. (6.4). In Eq. (6.4), $F(\cdot)$ represents a terminal cost or penalty term for not achieving a given possible set of target conditions. The integral is a measure of dynamic performance of the system during the process time. The exact expression and value of $F(\cdot)$ and $L(\cdot, \cdot)$ depends on the specific goal and nature of the optimization process. For example, for a so-called free-end point problem or no-penalty situation, $F(\cdot) = 0$; for a time-optimal control problem, $L(\cdot, \cdot) = 1$ and for a fuel-optimal system $L(\cdot, \cdot)$ is a function of $|u|$. Therefore, it should be noted that, this problem statement is just one of several possible such problems. This is evident by noting that the dynamic constraint-differential Eq. (6.1) can be linear, differential delayed, partial differential, and so on. The initial state $x_0$ in Eq. (6.1) or final state $x_f = x(t_f)$ may be both fixed, both free or a mixture of the two, and the feasibility constraint Eq. (6.3) can include the state as well as control. Finally, as it was indicated, the cost function, Eq. (6.4) can be quadratic in $x$ and $u$, special functions of $x$ and $u$ (e.g., optimal time, energy or fuel control situations) or be, in fact, described as an expected value of a random function. The problem statement mentioned is for continuous-time systems. Below, an equivalent discrete-time optimal control problem is defined.

Now, let us formulate a similar discrete-time optimal control problem. Consider a discrete-time system described by
\[ x(k + 1) = f[x(k), u(k), k] \]  
(6.5)
over a fixed interval $k_o \leq k \leq k_f$ and an initial state
\[ x(k_o) = x_0 \]  
(6.6)
an allowable set,
\[ u(k_o) \in U \]  
(6.7)
and a performance index,
\[ J = F[x(k_f), k_f] + \sum_{k=k_o}^{k_f} L[x(k), u(k), k]. \]  
(6.8)

The optimal control problem is to find a control sequence $u^*(k_o), u^*(k_o + 1), \ldots, u(k_f - 1)$ which satisfies Eqs. (6.5) to (6.7), while minimizing (or maximizing) Eq. (6.8). Note, once again, that the finite sum in Eq. (6.8) represents a cumulative deviation of the system's performance from a desired condition. If $L(\cdot)$ is, for example, a quadratic function of system errors, the optimal control problem is effectively a least-square optimization criterion. The latter criterion is very common in such related areas as estimation, identification, and filtering.

Here, one can similarly consider stochastic cases of these problems. It is obvious that considering even some of the possible combinations of system dynamics, boundary conditions, feasibility constraints and the objective function, a huge class of optimal control problems could be created which can be potentially subjects of several books.

It is our objective here to cover two basic classes of optimal control design
problems: linear continuous-time quadratic cost case (Sec. 6.3, and 6.4) and linear
discrete-time quadratic cost case (Sec. 6.5). The reason for choosing these two
problems is their convenient analytical solutions and implementation in CAD envi-
ronment.

The solution of an optimal control problem can be obtained through two basic
approaches. The first is by the application of Hamilton-Jacobi equation which con-
stitutes a sufficiency condition. The other approach is the Minimum Principle of
Pontryagin which constitutes necessary conditions for optimality.

To illustrate this problem, consider an initial example.

Example 6.1
Consider an inverted pendulum or a so-called "broom" balancing system shown in Fig. 6.1
(see also Cannon, 1967; Elgerd, 1967; Kwakernaak and Sivan, 1972). The carriage can be
moved in the horizontal direction through exertion of proper force \( f(t) \). The carriage is driven
by a small motor which is assumed to have a limited capacity in terms of its output torque.
The motor exerts the force \( f(t) \) at time \( t \) which is constrained by a maximum value of \( f_{\text{max}} \).
This displacement of the pivot at time \( t \) is \( p(t) \), and the angular position of the pendulum
with the vertical axis is represented by \( \theta(t) \). The pendulum's mass is \( m \), its length \( l \), its
moment of inertia is \( J \). The mass of the carriage is represented by \( M \). Derive the dynamic
model and formulate the optimal control problem.

Solution In an attempt to develop a dynamic model for this system, it is noted that
the forces acting on the pendulum are gravity \( mg \), \( (g \) is the gravitational acceleration
constant) horizontal reaction force \( F_h(t) \) and a vertical reaction force \( F_v(t) \) in the pivot.
The following applications of Newton's laws can now be stated:

\[
m \frac{d^2}{dt^2} [p(t) + l \sin \theta(t)] = F_h(t) \tag{6.9}
\]

\[
m \frac{d^2}{dt^2} [l \cos \theta(t)] = F_v(t) - mg \tag{6.10}
\]

\[
J \ddot{\theta}(t) = l F_v(t) \sin \theta(t) - l F_h(t) \cos \theta(t) \tag{6.11}
\]

\[
M \ddot{p}(t) = f(t) - F_h(t) - B \dot{p}(t) \tag{6.12}
\]

It is noted that no friction has been assumed at the pivot and the carriage has
a friction coefficient \( B \). After differentiating the terms in Eqs. (6.9) and (6.10), one
obtains

---

![Figure 6.1 An inverted pendulum.](image-url)
\[ m \ddot{p}(t) + m l \dddot{\theta}(t) \cos \theta(t) - m l \dot{\theta}^2(t) \sin \theta(t) = F_h(t) \]  
(6.13)

\[ - m l \ddot{\theta}(t) \sin \theta(t) - m l \dot{\theta}^2(t) \cos \theta(t) = F_v(t) - mg \]  
(6.14)

\[ J \ddot{\theta}(t) = l F_v(t) \sin \theta(t) - l F_h(t) \cos \theta(t) \]  
(6.15)

\[ M \ddot{p}(t) = f(t) - F_h(t) - B \dot{p}(t) \]  
(6.16)

Assuming that \( m \) is small as compared with \( M \) and thus neglecting the horizontal reaction force \( F_h(t) \), Eq. (6.16) would reduce to

\[ M \ddot{p}(t) = f(t) - B \dot{p}(t) \]  
(6.17)

After forces \( F_h(t) \) and \( F_v(t) \) have been eliminated in Eqs. (6.13), (6.14), and (6.15) one obtains

\[ (J + ml^2) \ddot{\theta}(t) - mgl \sin \theta(t) + ml \ddot{p}(t) \cos \theta(t) = 0 \]  
(6.18)

Dividing Eq. (6.18) by \((J + ml^2)\) and denoting \((J + ml^2)/ml\) by “effective pendulum length” \( l_e \), Eq. (6.18) reduces to

\[ \ddot{\theta}(t) - \frac{g}{l_e} \sin \theta(t) + \frac{1}{l_e} \ddot{p}(t) \cos \theta(t) = 0 \]  
(6.19)

Note that a pendulum with a mathematical length \( l_e \) would have a dynamic relationship as in Eq. (6.19).

Now, assuming the following choice for the state and control variables:

\[ x_1 = p(t), x_2 = \dot{p}(t) \]  
(6.20)

\[ x_3 = p(t) + l_e \theta(t), x_4 = \dot{p}(t) + l_e \dot{\theta}(t), u = f(t) \]

then, the state equation of the nonlinear inverted pendulum, after some manipulation, can be represented by

\[ \dot{x}_1 = x_2 \]

\[ \dot{x}_2 = \frac{u}{M} - \frac{B}{M} x_2 \]  
(6.21)

\[ \dot{x}_3 = x_4 \]

\[ \dot{x}_4 = g \sin \left( \frac{x_3 - x_1}{l_e} \right) - \frac{1}{M} \cos \left( \frac{x_3 - x_1}{l_e} \right) + \frac{u}{M} - \frac{B}{M} x_2 \]

which can be summarized by

\[ \dot{x} = f(x, u) \]  
(6.22)

where \( x = (x_1, x_2, x_3, x_4)^T \) is the state vector and control \( u \) is constrained by

\[ u \leq u_{max} \]  
(6.23)
A reasonable performance criterion for the inverted pendulum system is

$$J = \int_0^T (x^T Q x + u^2) \, dt$$

(6.24)

where $Q$ is a $4 \times 4$ positive—semidefinite matrix whose elements reflect relative weighting factors among the four states. Thus, the optimal control problem can be stated as follows: Find a control variable [force $f(t)$] which minimizes the cost function Eq. (6.24), while satisfying dynamic state Eq. (6.22) and constraint Eq. (6.23). This problem would be further treated in the current chapter.

### 6.2 SOLUTION OF THE OPTIMAL CONTROL PROBLEM

In this section, the optimal control problem for a continuous-time system, defined by Eqs. (6.1) to (6.4), will be solved. Both sufficient conditions (the Hamilton-Jacobi equation) and necessary conditions (the Minimum Principle) will be used to obtain the solutions. Because of the fact that the derivation of the Minimum Principle is, in general, lengthy, we provide a complete derivation of Hamilton-Jacobi equation first and then simply state the Minimum Principle. The special linear time-invariant case will then be considered.

#### 6.2.1 The Hamilton-Jacobi Equation

Consider the system Eq. (6.1) and (6.2), it is desired to find the optimal control $u^*(t)$, $t_o \leq t \leq t_f$, which minimizes the cost function Eq. (6.4). Assume that $f(\cdot)$, $L(\cdot)$, and $F(\cdot)$ are smooth functions of their arguments. Other than this restriction, $f(\cdot)$ can be arbitrary and $L(\cdot)$ and $F(\cdot)$ are nonnegative to reflect their physical meaning in the optimal control formulation. As indicated before, the optimal control $u^*(t)$ can be constrained to belong to a class or set [see Eq. (6.3)]. An example is for $U$ to represent the set of piecewise continuous functions, square-integrable functions bounded by unity, and so on.

Let $u_{[c,d]}$ denote a control function $u(\cdot)$ restricted to time interval $[c,d]$. Let us also provide the definition

$$J^* [x(t), t] = \min J [x(t), u(\cdot), t]$$

(6.25)

where $J^*(\cdot)$ represents the minimum value of $J(\cdot)$ which we are required to find. It is noted that $J^* [x(t), t]$ is independent of $u(\cdot)$, because the knowledge of $x(t_o)$ and $t$ intuitively determine $u(t)$, by the requirement that the control minimize $J [x(t), u(t), t]$. In order to find the optimal control which minimizes $J(\cdot)$, we can evaluate Eq. (6.25) for all $t$ and $x(t)$. Assume that an optimum $J^*$ exists in terms of $x(t)$ and $t$, together with the optimal control, we solve the optimization problem defined by Eqs. (6.1) and (6.4) by setting $t_o = t$.

For an arbitrary value of $t_o \leq t \leq t_f$ and $a t \leq t_1 \leq t_f$ one can describe the optimal value $J^*$ in Eq. (6.25) by
\[ J^*[x(t), t] = \min_{u(t,t_1)} \left[ \int_t^{t_1} L[x(\tau), u(\tau), \tau] \, d\tau + F[x(t_1)] \right] \]

\[ = \min_{u(t,t_1)} \left\{ \min_{u(t,t_2)} \left[ \int_t^{t_2} L[x(\tau), u(\tau), \tau] \, d\tau + \int_{t_2}^{t_1} L[x(\tau), u(\tau), \tau] \, d\tau + F[x(t_1)] \right] \right\} \]

\[ = \min_{u(t,t_1)} \left\{ \int_t^{t_1} L[x(\tau), u(\tau), \tau] \, d\tau + \min_{u(t,t_2)} \left[ \int_{t_2}^{t_1} L[x(\tau), u(\tau), \tau] \, d\tau + F[x(t_1)] \right] \right\} \]

or

\[ J^*[x(t), t] = \min_{u(t,t_1)} \left[ \int_t^{t_1} L[x(\tau), u(\tau), \tau] \, d\tau + J^*[x(t_1), t_1] \right] \quad (6.26) \]

If we let \( t_1 = t + \Delta t \) in Eq. (6.26), for small \( \Delta t \) and expand its right-hand side by a Taylor’s series expansion (noting that this is possible through the smoothness assumption on \( J^* \)), one obtains

\[ J^*[x(t), t] = \min_{u(t,t_1 + \Delta t)} \left\{ \Delta t L\left[ x(t + a \Delta t), u(t + a \Delta t), t + a \Delta t \right] + J^*[x(t), t] \right. \]

\[ + \left[ \frac{\partial J^*}{\partial x} [x(t), t] \right] \frac{dx(t)}{dt} \Delta t + \frac{\partial J^*}{\partial t} [x(t), t] \Delta t + O(\Delta t^2) \] \]

where \( 0 \leq a \leq 1 \) is a constant. In view of Eq. (6.1) and solving for the last part derivative in the right-hand side of this relation,

\[ \frac{\partial J^*}{\partial t} [x(t), t] = - \min_{u(t,t_1 + \Delta t)} \left\{ L\left[ x(t + a \Delta t), u(t + a \Delta t), t + a \Delta t \right] \right. \]

\[ + \left[ \frac{\partial J^*}{\partial x} [x(t), t] \right]^T f [x(t), u(t), t] + O(\Delta t) \left\} \]

If we now let \( \Delta t \) approach zero, this equation would become

\[ \frac{\partial J^*}{\partial t} [x(t), t] = - \min_{u(t)} \left\{ L\left[ x(t), u(t), t \right] + \left[ \frac{\partial J^*}{\partial x} [x(t), t] \right]^T f [x(t), u(t), t] \right\} \quad (6.27) \]

Here, \( J^* \) is the unknown quantity, while \( f \) and \( L \) are known functions. This is one form of the Hamilton-Jacobi equation. In this format, it is a mixture of partial and functional differential equation, but not a pure partial differential equation.

The value of \( u(t) \) minimizing the right-hand side of Eq. (6.27) depends on \( x(t) \), \( \partial J^*/\partial x \) and \( t \). Let this minimizing control be denoted by \( \tilde{u}[x(t), \partial J^*/\partial x, t] \). With this definition of a minimizing control, Eq. (6.27) becomes

\[ \frac{\partial J^*}{\partial t} = -L \left[ x(t), \tilde{u} \left[ x(t), \frac{\partial J^*}{\partial x}, t \right] \right] + \frac{\partial J^*}{\partial x} f \left[ x(t), \tilde{u} \left[ x(t), \frac{\partial J^*}{\partial x}, t \right], t \right] \quad (6.28) \]
This equation is a first-order partial differential equation with \( J^* \) as the dependent variable and \( x(t) \) and \( t \) as independent variables since \( L(\cdot), f(\cdot), \) and \( \tilde{u}(\cdot) \) are known functions of their arguments. To derive a boundary condition for Eq. (6.28), note that the performance index Eq. (6.4) implies \( J[x(t_f), u(\cdot), t_f] = F[x(t_f)] \) for all \( u(\cdot) \), and accordingly, the minimum value of this performance index with respect to \( u(\cdot) \) is also \( F[x(t_f)] \), that is,

\[
J^*[x(t_f), t_f] = F[x(t_f)]
\]  

(6.29)

The pair of Eqs. (6.28) and (6.29) constitute the Hamilton-Jacobi partial differential equation.

With the Hamilton-Jacobi equation defined, we now seek the optimal control for the problem defined by Eqs. (6.1) to (6.4). Assume that a solution to Eqs. (6.28) and (6.29) exists such that \( J^*[x(t), t] \) is a known function of its arguments. Let this solution define a control \( \hat{u}[x(t), t] \), that is,

\[
\hat{u}[x(t), t] = \hat{u} \left[ x(t), \frac{\partial J^*}{\partial x} (x(t), t), t \right]
\]

This new control function has two important attributes. First, \( \hat{u}[x(t), t] \) represents the control which minimizes

\[
J[x(t), u(\cdot), t] = F[x(t_f)] + \int_t^{t_f} L[x(\tau), u(\tau), \tau] \, d\tau
\]  

(6.30)

This indicates that to obtain the optimal performance index \( J^*[x(t), t] \), we begin with control \( \hat{u}[x(t), t] \). This point is implicit in the arguments leading to Eq. (6.27), as previously shown.

The second attribute or property of \( \hat{u}(\cdot) \) is that the optimal control \( u^*(\cdot) \) for the original minimization problem defined by Eqs. (6.1) through (6.4), with \( t_o \) as the initial time and \( t \) as an intermediate value of time, is related to \( \hat{u}(\cdot, \cdot) \) simply by

\[
u^*(t) = \hat{u} \left[ x(t), t \right]
\]  

(6.31)

where \( x(t) \) is the state at time \( t \) arising from application of \( u^*(\cdot) \) over \([t_o, t]\).

Although this argument may be clear to some, we can demonstrate it through a variation of arguments leading to Eq. (6.26). By definition,

\[
J^* [x(t_o), t_o] = \min \left[ \int_{t_o}^{t_f} L[x(\tau), u(\tau), \tau] \, d\tau + F[x(t_f)] \right]
\]

and the minimum is obtained by \( u^*(\cdot) \). With \( u^*(\cdot) \) regarded as the sequential use of \( u^*(t_o, t) \) and \( u^*(t, t) \); it is evident that through the assumption that \( u(t_o, t) \) is applied until time \( t \), one has

\[
J^*[x(t_o), t_o] = \min_{u(t_o, t)} \left[ \int_{t_o}^{t_f} L[x(\tau), u(\tau), \tau] \, d\tau + F[x(t_f)] \right]
\]  

(6.32)
\begin{equation}
= \int_{t_0}^{t} L \left[ x(\tau), u(\tau), \tau \right] d\tau \\
+ \min_{u(t, \tau)} \left[ \int_{t}^{\tau} L \left[ x(\tau), u(\tau), \tau \right] d\tau + F(x(t_f)) \right]
\end{equation}

Thus, the minimization in Eq. (6.32), and hence in Eq. (6.33), is achieved by \( u^*(t, t_f) \). On the other hand, \( u^*(t, t_f) \) is the optimal control for the system in Eq. (6.1) with performance index

\begin{equation}
J(\cdot) = \int_{t}^{\tau} L \left[ x(\tau), u(\tau), \tau \right] d\tau + F(x(t_f))
\end{equation}

with initial state \( x(t) \), where \( x(t) \) is obtained by steering Eq. (6.1) with state \( x(t_0) \) at time \( t_0 \) when \( u(t_0, t) \) is applied. In fact, this is one possible statement of Dynamic Programming's Principle of Optimality, which in effect, states that a control policy which is optimal over an interval \((t_0, t_f)\) is optimal over all subintervals \((t, t_f)\). However, \( \hat{u} [x(t), \tau] \) is the optimal control at time \( t \) for the performance index Eq. (6.34), and so

\begin{equation}
\hat{u} [x(t), \tau] = u^*(t, t_f)(t) = u^*(t)
\end{equation}

A few points need to be discussed. One is that \( \hat{u}[x(t), \tau] \) is independent of \( t_0 \) which implies that the optimal control at an arbitrary time \( t \) for the minimization of

\begin{equation}
J[x(\sigma), u(\cdot), \sigma] = \int_{\sigma}^{t} L \left[ x(\tau), u(\tau), \tau \right] d\tau + F(x(t_f))
\end{equation}

is also \( \hat{u}[x(t), \tau] \). In other words, the control \( \hat{u} [x(\cdot), \cdot] \) is the optimal control for the whole class of problems Eq. (6.36), with variable \( \sigma \) and \( x(\sigma) \).

The other point is that the optimal control at time \( t \) is expressed in terms of the state \( x(t) \) at time \( t \), even though its functional dependence on the state may not be constant, it is generally a time-varying function of the state. Theoretically, the optimal control is implemented with a feedback law. It should be noted that other approaches such as the Minimum Principle (Sec. 6.2.2) and Euler-Lagrange equations for computing the optimal control do not necessarily have this desirable property. In the latter schemes, the optimal control is normally given as a certain function of time.

The final point is that the remarks leading to the Hamilton-Jacobi equation can be reversed, that is, if a suitable solution to the equation is known, this solution has to be the optimal performance index \( J^*[x(t), \tau] \). To illustrate the derivation of the Hamilton-Jacobi equation, let us consider a simple example.

**Example 6.2**

Consider a simple first-order system described by

\[ \dot{x} = 2u \]

with performance index
\[ J(x(0), u(\cdot), 0) = \int_0^t \left( u^2 + x^2 + \frac{1}{4} x^4 \right) dt \]

It is desired to develop the Hamilton-Jacobi equation.

**Solution** Using Eq. (6.27), one has

\[ \frac{\partial J^*}{\partial t} = -\min_{u(t)} \left\{ u^2 + x^2 + \frac{1}{4} x^4 + \frac{\partial J^*}{\partial x} \frac{\partial J^*}{\partial u} 2u \right\} \]

The minimizing \( u(t) \) is given by

\[ \hat{u} = -\frac{\partial J^*}{\partial x} \]

and one has

\[ \frac{\partial J^*}{\partial t} = \left( \frac{\partial J^*}{\partial x} \right)^2 - x^2 - \frac{1}{4} x^4 \]

as the Hamilton-Jacobi equation for the problem with boundary condition \( J^*[x(t_f), t_f] = 0 \). The solution of the equation is quite complex and in fact, it is rarely possible to solve the Hamilton-Jacobi equation in its general form. In the following section a special solution to this equation is obtained for a linear time-varying system.

**Solution of linear regulator problem.** In this section, a special optimal control problem, Linear Regulator Problem, will be addressed.

Consider a linear time-varying system

\[ \dot{x}(t) = A(t) x(t) + B(t) u(t), \quad x(t_0) \quad (6.37) \]

is given where \( A(t) \) and \( B(t) \) are assumed to be continuous. Let us define a *performance index* (cost function)

\[ J[x(t_0), u(\cdot), t_o] = \int_{t_o}^{t_f} [u^T R(t) u + x^T Q(t) x] dt + x^T(t_f) F x(t_f) \quad (6.38) \]

where matrices \( Q(t) \) and \( R(t) \) are continuous, symmetric, and nonnegative and positive definite, respectively. Let \( F \) be a nonnegative definite matrix. The optimal control problem is to find a control function \( u^*(t) \), \( t_0 \leq t \leq t_f \), which satisfies the system dynamics Eq. (6.37), while minimizing cost functional Eq. (6.38).

Let us, for the time being, assume that \( t_f \) is finite and the quadratic cost functional \( J^*[x(t), t] \) in Eq. (6.26) be of the form,

\[ J^*[x(t), t] = x^T(t) K(t) x(t) \quad (6.39) \]

where \( K(t) \) is, without any loss of generality, a symmetric matrix. If \( K(t) \) is not symmetric, it may be replaced by the symmetric matrix \( 1/2 [K(t) + K^T(t)] \) without altering Eq. (6.39).

The Hamilton-Jacobi equation will be used to derive the optimal control. Consider the first form of the Hamilton-Jacobi equation Eq. (6.27) repeated here
\[
\frac{\partial J^*}{\partial t} [x(t), t] = - \min_{u(t)} \left\{ L [x(t), u(t), t] + \left[ \frac{\partial J^*}{\partial x} [x(t), t] \right]^T f [x(t), u(t), t] \right\}
\]  \hspace{2cm} (6.40)

Now, replacing for \( f (\cdot), L (\cdot) \), and \( J^* (\cdot) \) from Eq. (6.37), (6.38) and (6.39), respectively, Eq. (6.40) becomes
\[
x^T K x = - \min (u^T R u + x^T Q x + 2x^T K A x + 2 x^T K B u) \hspace{2cm} (6.41)
\]

In order to find the minimum of the expression on the right-hand side of Eq. (6.41), one can note the following identity which is obtained by completing the square
\[
u^T R u + x^T Q x + 2x^T K A x + 2 x^T K B u = x^T (Q - K B R^{-1} B^T K + K A + A^T K) x \]

Since it is assumed that \( R (t) \) is positive definite, it follows that Eq. (6.41) is minimized by setting
\[
\hat{u}(t) = - R^{-1} (t) B^T (t) K (t) x(t)
\]  \hspace{2cm} (6.42)

where one obtains
\[
x^T K x = - x^T (Q - K B R^{-1} B^T K + K A + A^T K) x
\]
or by equating the middle coefficients, and using the fact that both sides are symmetric, one has
\[
- \dot{K}(t) = K(t) A(t) + A^T(t) K(t) - K(t) B(t) R^{-1}(t) B^T(t) K(t) + Q(t)
\]  \hspace{2cm} (6.43)

Equation (6.43) is known as the matrix Riccati equation which constitutes a nonlinear differential matrix equation. Note from Eq. (6.43), that the Riccati matrix is a symmetric matrix. To obtain a boundary condition for the Riccati equation, recall that \( J^* [x(t_f), (t_f)] = F [x(t_f)] \) which in view of Eqs. (6.38) and (6.39), it is clear that
\[
K(t_f) = F
\]  \hspace{2cm} (6.44)

To summarize the solution of the linear state regulator by the Hamilton-Jacobi equation, it is noted that

1. the optimal control is given by
\[
u^* (t) = - R^{-1} (t) B^T (t) K(t) x(t)
\]  \hspace{2cm} (6.45)

where \( K(t) \) is the solution of the differential matrix Riccati equation Eqs. (6.43) and (6.44).

2. the optimal value of the performance index or cost function is given by
\[
J^* [x(t_0), (t_0)] = x^T (t_0) K(t_0) x(t_0)
\]  \hspace{2cm} (6.46)

3. \( K(t) \) is a symmetric positive definite matrix.

Example 6.3
Consider a linear system,
\[ \dot{x} = \frac{1}{2}x + u \quad x(0) = 1 \]

with a performance index,

\[ J = \frac{1}{2} \int_0^1 \left( e^{-t}x^2 + 4e^{-t}u^2 \right) dt \]

It is desired to find the optimal control for this system.

**Solution**  The Riccati equation for this problem is given by

\[ \dot{K} = -K + \frac{1}{2}e^tK^2 - \frac{1}{2}e^{-t} \quad K(1) = 0 \]

The solution of this equation can be verified to be (Anderson and Moore, 1971)

\[ K(t) = (1 - e^t e^{-1})(e^t + e^{2t} e^{-1})^{-1} \]

The optimal control would be given by,

\[ u^*(t) = \frac{-1}{2} (1 - e^t e^{-1})(1 + e^t e^{-1})^{-1} x(t) \]

and the optimal performance index is \( J^* = 1 \).

### 6.2.2 The Minimum Principle

The Hamilton-Jacobi equation, Eq. (6.27) provides a sufficient condition for the solution of the optimal control problem. If the set of admissible controls is not restricted, one may use the calculus of variations (Gelfand and Fomin, 1963) to derive a set of necessary conditions for optimization. When the set of admissible control is bounded in some manner, unrestricted variations of \( u(t) \) are not allowed. This point can be illustrated by noting a function of a single variable shown in Fig. 6.2. If the minimum occurs at the boundary, then it is no longer true that the first variation, that is, the slope, vanishes at that point. The minimum principle (Pontryagin, et al., 1962) is an extension of the calculus of variations to the case of bounded control variables.

Consider an autonomous system,

\[ \dot{x} = f(x, u) \quad (6.47) \]

with initial state,

\[ x(t_0) = x_0 \quad (6.48) \]

a class of admissible controls

\[ u(t) \in U \quad (6.49) \]

and performance index,
\[ J(x_0, u) = F[x(t_f)] + \int_{t_0}^{t_f} L[x(t), u(t)] dt \]  \hspace{1cm} (6.50)

where \( L(\cdot) \) is continuously differentiable in \( \mathbb{R}^n \times U \). Let us define a scalar function first.

**Definition 6.1.** Let \( H(x, u, p) \) denote the real-valued scalar function of the \( n \)-dimensional vector \( x \), the \( m \)-dimensional vector \( u \), and the \( n \)-dimensional vector \( p \), given by

\[ H(x, u, p) = L(x, u) + p^T f(x, u) \]  \hspace{1cm} (6.51)

where \( p \in \mathbb{R}^n \) is called the costate vector. The function \( H(\cdot) \) is called the Hamiltonian function. Now, the minimum principle can be stated by the following theorem.

**Theorem 6.1.** (Pontryagin et al. 1962) Suppose that \( u^* \) is an optimal control for this problem and \( x^* \) is the corresponding state trajectory. Then there exists a nonzero vector \( p(t) \) such that

\[ \dot{x} = \frac{\partial H}{\partial p} = f(x, u) \]  \hspace{1cm} (6.52)

\[ \dot{p} = -\frac{\partial H}{\partial x} = -\frac{\partial L}{\partial x} - \left( \frac{\partial f}{\partial x} \right)^T p \]  \hspace{1cm} (6.53)

\[ H(x^*, u^*, p^*) = \min H(x^*, u, p^*) \]  \hspace{1cm} (6.54)

\[ u \in U \]

Furthermore, \( H(x^*, u^*, p^*) \) is constant for \( 0 \leq t \leq t_f \). The costate vector differential Eq. (6.53) will be accommodated with a final condition which is known as the
transversality condition. This condition depends on the final condition \( x(t) \) and final time \( t_f \) (Athans and Falb, 1966) which is detailed below.

**Transversality condition.** The notion of transversality conditions is a very important aspect of the optimal control problem and deserves a detailed discussion. To begin this discussion, consider the following definitions to define the concept of "smoothness" for subsets of \( \mathbb{R}^n \).

**Definition 6.2.** Let \( f(x) \) be a continuous real-valued scalar function of vector \( x \in \mathbb{R}^n \). Then a subset of \( \mathbb{R}^n \), denoted by \( S(f) \) which has the property

\[
S(f) = \{ x : f(x) = 0 \}
\]

is called a continuous hypersurface in \( \mathbb{R}^n \) determined by \( f \).

**Definition 6.3.** Let \( x_0 \in S(f) \), an element of the hypersurface in \( \mathbb{R}^n \) determined by \( f(x) \), that is, \( f(x_0) = 0 \). Then, if the following gradient exists and satisfies \( \partial f/\partial x_0 \neq 0 \), \( x_0 \) is said to be a regular point of \( S(f) \). Moreover, if \( x_0 \) is a regular point of \( S(f) \), then the set \( L(x) \) defined by

\[
L(x_0) = \{ x : \langle \frac{\partial f}{\partial x} (x_0), x - x_0 \rangle = 0 \}
\]

is called a hyperplane which passes through the point \( x_0 \). Here \( \langle \cdot, \cdot \rangle \) denotes the usual scalar product.

**Definition 6.4.** If \( x_0 \) is a regular point of the hypersurface \( S(f) \), then \( L(x_0) \) is said to be the target hyperplane of \( S(f) \) at \( x_0 \). Moreover, the vector \( n(x_0) \) satisfying

\[
n(x_0) = d \frac{\partial f(x_0)}{\partial x}, \ d \neq 0
\]

is called normal to \( S(f) \) at \( x_0 \). Furthermore, if \( f(x) \) is differentiable, that is, \( \partial f/\partial x \) exists; and the gradient \( \partial f/\partial x \) is continuous from \( \mathbb{R}^n \) to \( \mathbb{R}^n \), then \( f \) is said to be a smooth function.

**Definition 6.5.** If \( f \) is a smooth function and if every point \( x_0 \in S(f) \) is regular, then \( S(f) \) is said to be a smooth hypersurface.

To appreciate some of these definitions, consider the following example.

**Example 6.4**

Let a real-valued function be defined by \( f(x) = x_1^2 + x_2^2 - 4 \), when \( x \in \mathbb{R}^2 \). It is desired to illustrate the above definitions.

**Solution** The set \( S(f) \) is a circle with radius 2. It is clear that \( f \) is a smooth function. The set \( S(f) \) has an infinite number of tangent hyperplanes. One such hyperplane, \( L([- \sqrt{2}, \sqrt{2}]) \), is the line \( x_1 - x_2 + 2\sqrt{2} = 0 \) which is shown in Fig. 6.3.
Now, let us consider a number of distinct smooth functions $f_1, f_2, \ldots, f_{n-k}$ on $\mathbb{R}^n$. Then the set $S(f_1, f_2, \ldots, f_{n-k})$ is the intersection of $n-k$ hypersurfaces $S(f_1), S(f_2), \ldots, S(f_{n-k})$ that is,

$$S(f_1, f_2, \ldots, f_{n-k}) = S(f_1) \cap S(f_2) \cap \ldots \cap S(f_{n-k}) \quad (6.58)$$

In other words, $S(f_1, f_2, \ldots, f_{n-k})$ represents all the points (its elements) at which all functions $f_i, i = 1, \ldots, n-k$ are zero, that is,

$$S(f_1, f_2, \ldots, f_{n-k}) = \{ x : f_i(x) = 0, \quad i = 1, 2, \ldots, n-k \} \quad (6.59)$$

Now, we finally have the following two definitions.

**Definition 6.6.** A set $S(f_1, f_2, \ldots, f_{n-k})$ is said to be a smooth $k$-fold in $\mathbb{R}^n$ if, for every point $x_0 \in S(\cdot)$, then the $n-k$ vectors $(\partial f_1 / \partial x)(x_0), \ldots, (\partial f_{n-k} / \partial x)(x_0)$ are linearly independent (see Appendix A).

It is noted that if $S(f_1, \ldots, f_{n-k})$ is a smooth $k$-fold, then each of the hypersurfaces $S(f_i)$ is smooth. If $x_0 \in S(f_1, \ldots, f_{n-k})$, then the tangent hyperplane $T_i(x_0)$ of $S(f_i)$ at $x_0$ is defined for $i = 1, 2, \ldots, n-k$. Now, let

$$M(x_0) = \bigcap_{i=1}^{n-k} T_i(x_0) = \left\{ x : \left< \frac{\partial f_i}{\partial x}(x_0), x - x_0 \right> = 0 \right\}$$

for $i = 1, 2, \ldots, n-k \quad (6.60)$
which is called the tangent plane of $S(f_1, \ldots, f_{n-k})$ at $x_0$. Therefore, a smooth $k$-fold has a tangent plane at every point.

Example 6.5

Let $f_1$ and $f_2$ be two real-valued functions on $\mathbb{R}^3$ given by

$$f_1(x) = x_1 - \sin x_3 \quad f_2(x) = x_2 - \cos x_2$$

where $x = (x_1, x_2, x_3)^T$. It is desired to define its 1-fold surface.

Solution The functions $f_1$ and $f_2$ are smooth, and the 1-fold $S(f_1, f_2)$ in $\mathbb{R}^2$ is the helix illustrated by Fig. 6.4. There are an infinite number of tangent lines for the 1-fold $S(\cdot)$. One such line is $M([0 \ 1 \ 0]^T)$, at point $[0 \ 1 \ 0]^T$, the intersection of the planes $x_1 - x_3 = 0$ and $x_2 - 1 = 0$.

Now, consider one final definition.

Definition 6.7. Let $S(f_1, \ldots, f_{n-k})$ be a smooth $k$-fold in $\mathbb{R}^n$ and let $x_0$ be an element of $S(f_1, f_2, \ldots, f_{n-k})$. Then a vector $p$ is said to be transversal to $S(\cdot)$ at $x_0$ if

$$< p, x - x_0 > = 0$$

(6.61)

for all $x \in M(x_0)$. It is noted here that $p$ is a linear combination (see Appendix A) of the $n - k$ vectors $(\partial f_i/\partial x)(x_0)$, $i = 1, \ldots, n - k$. Furthermore, a vector $p$ will be transversal to $S(\cdot)$ at $x_0$ if and only if $p$ satisfies the $k$ relations

$$< p, x^i - x_0 > = 0, \quad i = 1, \ldots, k$$

(6.62)

where $x^i$ are elements of $M(x_0)$ such that the vectors $x^i - x_0$ are linearly independent. After these preliminary mathematics, we can now give a detailed account of the transversality condition of the optimal control.

With respect to Pontryagin's necessary conditions for optimality, (Theorem
there exist a nonzero vector \( p^*(t) \), state and control vectors \( x^*(t) \) and \( u^*(t) \) which satisfy

\[
\dot{x}^* = \frac{\partial H}{\partial p} (x^*, p^*, u^*) \quad (6.63)
\]

\[
\dot{p}^* = - \frac{\partial H}{\partial x} (x^*, p^*, u^*) \quad (6.64)
\]

\[H (x^*, u^*, p^*) = \min H (x^*, u, p^*) \quad (6.65)\]

\[u \in U\]

with boundary condition

\[x^*(t_0) = x_0, \quad x^*(t_f) \in S_1 \quad (6.66)\]

where \( S_1 \) is a smooth \( k \)-fold in \( R^n \). Then, the vector \( p^*(t) \) is transversal to \( S_1 \) at \( x^*(t_f) \), that is,

\[< p^*(t), x - x^*(t_f) > = 0 \quad (6.67)\]

for all \( x \in M \{ x^*(t_f) \} \), where \( M(\cdot) \) is the tangent plane of \( S_1(\cdot) \) at \( x^*(t_f) \), see Eq. (6.60). The conditions Eq. (6.67) which are the same as Eq. (6.61) are called the transversality conditions of optimal control. The exact expression for these conditions depends on the target set \( S_1 \) for the final optimal state of the system. For example, for a nonlinear time-invariant system with \( F(x) \) as its penalty function and target set \( S_1 \) \( k \)-fold in \( R^n \), that is, \( \gamma_i(x) = 0, \ i = 1, \ldots, n - k \), then the transversality condition Eq. (6.67) reduces to

\[p^*(t_f) = \sum_{i=1}^{n-k} \alpha_i \frac{\partial \gamma_i}{\partial x} \bigg|_{x(x_f)} + \frac{\partial F}{\partial x} \bigg|_{x(x_f)} \quad (6.68)\]

Table 6.1, depicted from Athans and Falb (1966) provides a good summary of five optimal control problems' necessary conditions of optimality, including the transversality.

For example, as shown in Table 6.1, if \( x(t_f) = x_f \), a fixed point, then regardless of \( t_f \) these would be no conditions on \( p(t) \). For a free-end point situation, that is, \( x(t_f) \in R^n \) and point situation, that is, \( x(t_f) \in R^n \) and fixed or free \( t_f \), this condition would be

\[p(t_f) = \frac{\partial F[x(t_f)]}{\partial x(t_f)} \quad (6.69)\]

The relation Eq. (6.54) constitutes a set of \( m \) static equations, that is, \( m \) inequalities for a constrained control \( u(t) \in U \) or a set of \( m \) algebraic equations,

\[0 = \frac{\partial H}{\partial u} = \frac{\partial L}{\partial x} + \left( \frac{\partial f}{\partial x} \right)^T p \quad (6.70)\]
for unconstrained \( u(t) \). It is noted that if \( u(t) \), in Eq. (6.54) or Eq. (6.65), can be eliminated in favor of \( x(t) \) and \( p(t) \) in Eqs. (6.52) and (6.53) together with initial and final conditions on \( x(t) \) and \( p(t) \), the following would result:

\[
\dot{x}(t) = f[x(t), p(t), t], \quad x(t_0) = x_0 \quad (6.71)
\]

\[
\dot{p}(t) = h[x(t), p(t), t], \quad p(t_f) = \frac{\partial F[x(t_f)]}{\partial x(t_f)} \quad (6.72)
\]

This constitutes \( 2n \)-dimensional two-point boundary-value (TPBV) problem. Here, a TPBV problem constitutes a set of differential equations in which the initial conditions of some and final conditions of the remaining equations are known. In other words, neither a complete set of initial nor final conditions for all equations is available. Clearly, the key to the solution of the optimal control problem Eqs. (6.52) to (6.54) and Eqs. (6.66) and (6.68) is the solution of its corresponding TPBV problem—one similar to Eqs. (6.71) and (6.72).

**Example 6.6**

Consider a scalar optimal control problem representing a simplified model of a nuclear reactor.

\[
\dot{x} = ux, \quad x(0) = x_0
\]

\[
J = \frac{1}{2} \int_0^1 (x^2 + u^2) \, dt
\]

It is desired to set up its optimal control problem.

**Solution** The optimal solution of this problem is demonstrated by the minimum principle. For this example, the Hamiltonian is,

\[
H = \frac{1}{2} (x^2 + u^2) + pux
\]

and the necessary conditions of optimality, following Eqs. (6.52) to (6.54) and Eqs. (6.66) and (6.68) are given by

\[
\dot{x} = \frac{\partial H}{\partial p} = ux \quad (6.73)
\]

\[
x(0) = x_0 \quad (6.74)
\]

\[
\dot{p} = \frac{-\partial H}{\partial x} = -x - pu \quad (6.75)
\]

\[
p(1) = 0 \quad (6.76)
\]

\[
0 = \frac{\partial H}{\partial u} = u + px \quad (6.77)
\]

which would result in the following TPBV problem after \( u \) has been eliminated using Eq. (6.77)
TABLE 6.1 Necessary Conditions for a Few Typical Optimal Control Problems

<table>
<thead>
<tr>
<th>System</th>
<th>Cost</th>
<th>Time</th>
<th>Target set</th>
<th>Hamiltonian</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$L = L(x, u), F = 0$</td>
<td>$t_f$ fixed end point</td>
<td>$x_f$ fixed end point</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$\dot{x} = f(x, u)$</td>
<td>$L = L(x, u), F = F(x)$</td>
<td>$S_f$ $k$-fold in $R^n$</td>
<td>$g(x) = 0$ $i = 1, 2, \ldots, n - k$</td>
</tr>
<tr>
<td>3</td>
<td>$\dot{x} = f(x, u)$</td>
<td>$L = L(x, u), F = F(x)$</td>
<td>$R^n$ free end point</td>
<td>$H = H(x, p, u)$</td>
</tr>
<tr>
<td>4</td>
<td>$L = L(x, u), F = 0$</td>
<td>$t_f$ fixed end point</td>
<td>$x_f$ fixed end point</td>
<td>$= L(x, u) + \langle p, f(x, u) \rangle$</td>
</tr>
<tr>
<td>5</td>
<td>$L = L(x, u), K = K(x)$</td>
<td>$g(x) = 0$ $i = 1, 2, \ldots, n - k$</td>
<td>$S_f$ $k$-fold in $R^n$</td>
<td></td>
</tr>
<tr>
<td>Table 6.1 (continued)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Necessary Conditions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1)</td>
<td>(2)</td>
<td>(3)</td>
<td>(4)</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
</tbody>
</table>
| $\dot{x}^* = \frac{\partial H}{\partial \rho} | _*$. | $H^* = H(x^*, p^*, u^*)$  
$= \min_{u \in U} H(x^*, p^*, u)$  
or  
$H(x^*, p^*, u^*) \leq H(x^*, p^*, u)$  
for all $u$ in $U$ | $H^*(t) = H^*(t_f) = 0$ | No condition on $p^*(t_f)$ |
| $\dot{p}^* = -\frac{\partial H}{\partial x} | _*$. | $p^*(t_f) = \sum_{i=1}^{n-k} \alpha_i \frac{\partial g_i}{\partial x} |_{t^*_{u^h}}$ | $p^*(t_f) = \frac{\partial F}{\partial x} |_{t^*_{u^h}}$ |
| | | | $p^*(t_f)$ normal to $S_f$ at $x^*(t_f)$ |
| | | | $p^*(t_f) = \sum_{i=1}^{n-k} \alpha_i \frac{\partial g_i}{\partial x} |_{t^*_{u^h}}$  
$+ \frac{\partial F}{\partial x} |_{t^*_{u^h}}$ |
| | | | No condition on $p^*(t_f)$ |

\[ \dot{x} = x^2 p, \quad x(0) = x_0 \]
\[ \dot{p} = -x + xp^2, \quad p(1) = 0 \]

Once this TPBV problem is solved, the optimal open-loop optimal control is obtained from,
\[ u(t) = -p(t)x(t) \]

The solution of TPBV problem is beyond the scope of the present text. Instead, we concentrate on the optimal control of linear systems.

**State regulator problem.** In this section the state regulator problem, defined in the previous section, will be discussed using the minimum principle.

Consider a linear controllable time-varying system,
\[ \dot{x}(t) = A(t)x(t) + B(t)u(t), \quad x(t_0) = x_0 \quad (6.78) \]
and a quadratic cost function,
\[ J = \frac{1}{2} x^T(t_f)Fx(t_f) + \frac{1}{2} \int_{t_0}^{t_f} [x^T(t)Q(t)x(t) + u^T(t)R(t)u(t)] \, dt \quad (6.79) \]

The basic assumptions are that \(Q(t)\) and \(R(t)\) are positive semidefinite and positive definite, respectively for all \(t\). Matrix \(F\) is assumed to be positive semidefinite. The problem is to find an optimal control \(u^*(t)\) which satisfies the state Eq. (6.78) while minimizing \(J\) in Eq. (6.79). The Hamiltonian, Eq. (6.51) in this case, is given by
\[ H(\cdot) = L + p^T f = \frac{1}{2} x^T(t)[Q(t)x(t) + \frac{1}{2} u^T(t)R(t)u(t)] + p^T(t)[A(t)x(t) + B(t)u(t)] \]
with the costate and control condition equations given by,
\[ \dot{\lambda}(t) = \frac{-\partial H(\cdot)}{\partial x(t)} = -Q(t)x(t) - A^T(t)p(t) \quad (6.80) \]
\[ 0 = \frac{\partial H(\cdot)}{\partial u(t)} = R(t)u(t) + B^T(t)p(t) \quad (6.81) \]

Since \(R(t)\) is nonsingular, \(u(t)\) can be eliminated from Eq. (6.81) to result,
\[ u(t) = -R^{-1}(t)B^T(t)p(t) \quad (6.82) \]
and when substituted in the state Eq. (6.78) and costate Eq. (6.80), the following linear TPBV problem would result
\[ \dot{x}(t) = A(t)x(t) + B(t)R^{-1}(t)B^T(t)p(t) \quad (6.83) \]
\[ \dot{\lambda}(t) = -Q(t)x(t) - A^T(t)p(t) \quad (6.84) \]
with boundary conditions,
\[ x(t_0) = x_0 \quad (6.85) \]
\[ \frac{\partial}{\partial x(t_f)} \left[ \frac{1}{2} x^T(t_f) F x(t_f) \right] = F x(t_f) \quad (6.86) \]

In matrix form, Eqs. (6.83) and (6.84) can be written as
\[
\begin{bmatrix}
    \dot{x}(t) \\
    \dot{p}(t)
\end{bmatrix}
= \begin{bmatrix}
    A(t) & -S(t) \\
    -Q(t) & -A^T(t)
\end{bmatrix}
\begin{bmatrix}
    x(t) \\
    p(t)
\end{bmatrix}
= H(t)
\begin{bmatrix}
    x(t) \\
    p(t)
\end{bmatrix}
\quad (6.87)
\]

where \( S(t) = B(t)R^{-1}(t)B^T(t) \) and the \( 2n \times 2n \) dimensional matrix \( H(t) \) is known as the Hamiltonian matrix.

Let \( \Phi(t; t_0) \) be the \( 2n \times 2n \) fundamental matrix of system (6.87), then
\[
\begin{bmatrix}
    x(t) \\
    p(t)
\end{bmatrix}
= \Phi(t; t_0)
\begin{bmatrix}
    x(t_0) \\
    p(t_0)
\end{bmatrix}
= \begin{bmatrix}
    \phi_{11}(t; t_0) & \phi_{12}(t; t_0) \\
    \phi_{21}(t; t_0) & \phi_{22}(t; t_0)
\end{bmatrix}
\begin{bmatrix}
    x(t_0) \\
    p(t_0)
\end{bmatrix}
\]

Using the transition properties of \( \phi(\cdot, \cdot) \), one can find,
\[
\begin{bmatrix}
    x(t_f) \\
    p(t_f)
\end{bmatrix}
= \begin{bmatrix}
    \phi_{11}(t_f; t) & \phi_{12}(t_f; t) \\
    \phi_{21}(t_f; t) & \phi_{22}(t_f; t)
\end{bmatrix}
\begin{bmatrix}
    x(t) \\
    p(t)
\end{bmatrix}
\]
or
\[
x(t_f) = \phi_{11}(t_f; t)x(t) + \phi_{12}(t_f; t)p(t) \\
p(t_f) = \phi_{21}(t_f; t)x(t) + \phi_{22}(t_f; t)p(t) \\
= F x(t_f) \\
= F \phi_{11}(t_f; t)x(t) + F \phi_{12}(t_f; t)p(t)
\]

Solving for \( p(t) \) in favor of \( x(t) \), in the last two relations, would lead to
\[
p(t) = \left[ F \phi_{22}(t_f; t) - F \phi_{12}(t_f; t) \right]^{-1} \left[ F \phi_{11}(t_f; t) - \phi_{21}(t_f; t) \right] x(t)
\]
\[
\Delta \equiv K(t)x(t)
\]

where \( K(t) \) is the \( n \times n \) time-varying matrix which depends on \( F \) and \( t_f \) and not on the initial state \( x(t_0) \). It can be shown that \( \left[ F \phi_{22}(t_f; t) - F \phi_{12}(t_f; t) \right] \) is nonsingular (see Prob. 6.6). In particular, \( K(t_f) \) is
\[
K(t_f) = \left[ F \phi_{22}(t_f; t) - F \phi_{12}(t_f; t) \right]^{-1} \left[ F \phi_{11}(t_f; t) - \phi_{21}(t_f; t) \right] \\
= (I_n - F0)^{-1} (FI_n - 0) = F.
\]

Alternatively we have
which also results that \( K(t) = F \).

Thus, the assumption \( p(t) = K(t)x(t) \) is justified and now can be used to reduce the linear TPBV problem Eq. (6.87) to two separate problems. Taking derivative of \( p(t) = K(t)x(t) \) and using Eq. (6.87) results,

\[
\dot{p}(t) = \dot{K}(t)x(t) + K(t)\dot{x}(t) \\
= \dot{K}x + K(Ax - Sp) = \dot{K}x + KAx - KS(Kx) \\
= -Qx - A^T p = -Qx - A^T Kx 
\]  

Comparing the coefficients of the \( x \) term in Eq. (6.88) leads to

\[
\dot{K}(t) = K(t)A(t) + A^T(t)K(t) - K(t)S(t)K(t) + Q(t) \tag{6.89} \\
K(t_0) = F \tag{6.90}
\]

This equation is the differential matrix Riccati equation (DMRE) which is identical to Eq. (6.43). The solution of DMRE Eqs. (6.89) and (6.90) can be proved to be nonnegative definite and symmetric. At first glance, DMRE represents \( n^2 \) simultaneous nonlinear differential equations which needs to be solved backward in time since only \( K(t) \) is known. However, since \( K(t) \) is symmetric, only \( n(n+1)/2 \) equations of the \( n^2 \) in Eq. (6.89) are needed, that is unknown elements are

\[
\begin{bmatrix}
k_{11}(t) & k_{12}(t) & \cdots & k_{1n}(t) \\
\vdots & k_{22}(t) & \ddots & \vdots \\
\vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \cdots & k_{nn}(t)
\end{bmatrix} \tag{6.91a}
\]

Once, \( K(t) \) is known, the optimal control law, by noting Eq. (6.82) and the new expression for \( p(t) \), is given by

\[
u^*(t) = -R^{-1}(t)B^T(t)K(t)x(t) = -G(t)x(t) \tag{6.91b}
\]

while the closed-loop optimal system is given by

\[
\dot{x}(t) = [A(t) - B(t)R^{-1}(t)B^T(t)K(t)]x(t) \\
= [A(t) - S(t)K(t)]x(t) \tag{6.91c} \\
= [A(t) - B(t)G(t)]x(t)
\]

The value of the optimal cost \( J \) in Eq. (6.79) can, once again, be shown (see Prob. 6.12) to be

\[
J^*[x(t_o), t_o] = \frac{1}{2} x_o^T K(t_o)x_o \tag{6.92}
\]

The optimal closed-loop system in Eq. (6.91c) is asymptotically stable since the system Eq. (6.78) is assumed to be controllable.

**Example 6.7**

Consider a second-order system, known as the double-integral plant.
\[ x_1 = x_2 \]
\[ x_2 = u \]

with cost function,
\[ J = \frac{1}{2} \left[ x_1^2 (3) + 2 x_2^2 (3) \right] + \frac{1}{2} \int_0^3 \left[ 2 x_1^2 (t) + 4 x_2^2 (t) + 2 x_1(t)x_2(t) + \frac{1}{2} u^2 (t) \right] dt \]

It is desired to solve the optimal control problem.

**Solution** In this problem,
\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad F = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}
\]

\[ Q = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix} \]

and \[ R = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \]

The corresponding Riccati equation is given by,
\[
\begin{bmatrix} \dot{k}_{11}(t) \\ \dot{k}_{12}(t) \\ \dot{k}_{21}(t) \\ \dot{k}_{22}(t) \end{bmatrix} = - \begin{bmatrix} k_{11}(t) & k_{12}(t) \\ k_{12}(t) & k_{22}(t) \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k_{11}(t) & k_{12}(t) \\ k_{12}(t) & k_{22}(t) \end{bmatrix} + \begin{bmatrix} k_{11}(t) & k_{12}(t) \\ k_{12}(t) & k_{22}(t) \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 4 \end{bmatrix}
\]

with boundary condition
\[ K(t_f) = K(3) = F = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \]

The three nonlinear coupled Riccati differential equations are given by
\[ \dot{k}_{12}(t) = 2k_{22}(t) - 2, \quad k_{11}(3) = 1 \]
\[ \dot{k}_{12}(t) = -k_{11}(t) + 2k_{12}(t)k_{22}(t) - 1, \quad k_{12}(3) = 0 \]
\[ \dot{k}_{22}(t) = -2k_{12}(t) + 2k_{22}(t) - 4, \quad k_{22}(3) = 2 \]

Clearly, these equations can not be easily solved analytically and a computer solution would be called for. In Sec. 6.2.3 numerical solutions of the DMRE and the algebraic matrix Riccati equation (AMRE) will be discussed and CAD examples will be given. Once the Riccati matrix \( K(t) \) is known, the overall system block diagram would be as shown in Fig. 6.5. The double lines in Fig. 6.5 represent multivariable or vector quantities.

**Example 6.8**

Consider an RC network shown in Fig. 6.6. It is desired to find a supply voltage \( u(t) = v(t) \) such that the capacitor is charging from \( t = 0, \quad v_c(0) = 10 \) volts to \( v_c(1) = 20 \) volts at \( t = 1 \) second while the energy lost in the resistor \( R \) is minimized.
Figure 6.5  Block diagram for the linear state regular problem.

Solution  Let the only state variable (energy-storing element capacitor) be \( x = v_c \). The state equation is

\[
 c v_c = i = \frac{u - v_c}{R}
\]

The cost function is

\[
 J = \int_0^1 Ri^2 \, dt = \int_0^1 \frac{(u - x)^2}{R} \, dt = \int_0^1 (u - x)^2 \, dt
\]

The Hamiltonian function and the corresponding necessary conditions for optimality are given by

\[
 H(x, p, u) = (u - x)^2 + p(u - x)
\]

\[
 0 = \frac{\partial H}{\partial u} = 2(u - x) + p, \text{ or } u = x - \frac{p}{2}
\]

Figure 6.6  An RC network for Example 6.8.
\[ \dot{x} = \frac{\partial H}{\partial p} = u - x = x - \frac{p}{2} - x = -\frac{p}{2} \]
\[ \dot{p} = -\frac{\partial H}{\partial x} = p + 2u - 2x = p + 2x - p - 2x = 0 \]

Therefore, the TPBV problem for this example reduces to
\[ \dot{x} = -\frac{p}{2}, \quad x(0) = 10 \]
\[ \dot{p} = 0, \quad p(1) = 20 \]

which is already decoupled, hence no Riccati transformation is necessary. The solution is given by
\[ p(t) = a = \text{constant} \]
\[ x(t) = x_0 - \frac{at}{2} \]

Using the boundary conditions on \( x(t) \) and \( p(t) \), one has,
\[ p(t) = 20 \]
\[ x(t) = 10 - 10t \]

and the optimal control is
\[ u^*(t) = x - \frac{p}{2} = -10 - 10t \]

For linear time-invariant case, consider
\[ \dot{x}(t) = Ax(t) + Bu(t) \quad (6.93) \]
\[ J = \frac{1}{2} \int_0^\infty [x^T(t)Qx(t) + u^T(t)Ru(t)] \, dt \]

where matrices \( A, B, Q, \) and \( R \) are constant and \( Q \) is assumed to be symmetric positive semidefinite, while \( R \) is symmetric positive definite.

The optimal solution of this problem is identical to Eq. (6.91b) except that the Riccati matrix is now constant
\[ u^*(t) = -R^{-1}B^TKx(t) \quad (6.94) \]

where
\[ 0 = -KA - ATK + KSK - Q \quad (6.95) \]

This equation is known as the algebraic matrix Riccati equation (AMRE). The closed-loop system's state equation and the optimal cost functional are given by,
\[ \dot{x}(t) = (A - SK)x(t), \quad x(0) = x_0 \]
\[ J^* = \frac{1}{2}x_0^TKx_0 \]
Example 6.9

Let us reconsider the second-order system of Ex. 6.7 with \( t_f \to \infty \), \( F = 0 \), \( x_0 = [1 \quad 2]^T \). It is desired to find \( u^* \), \( p \), \( x \), and \( J^* \).

**Solution** The \( A \), \( B \) matrices and controllability matrix are

\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad R_c = [B \quad AB] = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}
\]

which indicates that the system is completely controllable, hence an optimal solution exists. The algebraic matrix Riccati equation (AMRE), Eq. (6.89) is simply the one in Eq. (6.89) by letting all the time derivatives of \( k_{ij} \) equal to zero

\[
2k_{12}^2 - 2 = 0 \\
-k_{11} + 2k_{12}k_{22} - 1 = 0 \\
2k_{22}^2 - 2k_{12} - 4 = 0
\]

The positive-definite solution of this equation is given by, \( k_{12} = 1 \), \( k_{22} = (3)^{1/2} \) and \( k_{11} = 2(3)^{1/2} - 1 \), or

\[
K = \begin{bmatrix} 2(3)^{1/2} - 1 & 1 \\ 1 & (3)^{1/2} \end{bmatrix}
\]

The optimal control is given by

\[
u^* = -R^{-1}B^T Kx = -2x_1 - 2(3)^{1/2}x_2
\]

and the closed-loop system equation is,

\[
\dot{x} = (A - SK)x = \begin{bmatrix} 0 & 1 \\ -2 & -2(3)^{1/2} \end{bmatrix} x, \quad x_0 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}
\]

The costate vector is given by

\[
p = Kx = \begin{bmatrix} [2(3)^{1/2} - 1]x_1 + x_2 \\ x_1 + (3)^{1/2}x_2 \end{bmatrix}
\]

The optimal value of the cost is,

\[
J^* = \frac{1}{2} x_0^T K x_0 = \frac{1}{2} [6(3)^{1/2} + 3] = 6.7
\]

6.2.3 Numerical Solutions of Matrix Riccati Equations

In this section, two techniques for the numerical solution of the differential and algebraic matrix Riccati equations are briefly discussed. The first method is the standard numerical integration of the differential matrix Riccati equation by a fifth order variable step Runge-Kutta method. Numerical integration is still the most reasonable technique for the solution of DMRE. The second method is the Schur vector transformation
method which is the most reliable scheme to handle AMREs for both continuous-time and discrete-time cases.

Consider, once again, the time-varying DMRE,

\[ K(t) = -K(t)A(t) - A^T(t)K(t) + K(t)S(t)K(t) - Q(t) \]  \hspace{1cm} (6.96a)

\[ K(t_f) = F \]  \hspace{1cm} (6.96b)

where \( S(t) = B(t)R^{-1}(t)B^T(t) \). One of the earliest methods of solving DMRE Eq. (6.96) has been straight integration by classical numerical techniques such as the fourth-order Runge-Kutta. Program RICRKUT of LSSPAK/PC and primitives DMRE and AMRE of CONTROL/ab, and LQR of PC_MATLAB as well as RICCATI of MATRIXx all help solve the algebraic or differential matrix Riccati equations. The numerical integration scheme takes advantage of the symmetry of the Riccati matrix and solves Eq. (6.96) using a Runge-Kutta (or similar) algorithm. The program RICRKUT takes care of both time-varying and time-invariant cases. It is noted that for the time-invariant case one can let the solution matrix reach its steady state value

\[ K_{ss} = \lim_{t \to \infty} K(t) \]

Recall the AMRE Eq. (6.95) which is rewritten here again,

\[ A^T K + KA - KSK + Q = 0 \]  \hspace{1cm} (6.97)

We begin by defining the Hamiltonian matrix

\[ H = \begin{pmatrix} A & -S \\ -Q & -A^T \end{pmatrix} \]

Then a well-known property of \( H \) is that all the eigenvalues of \( H \) are symmetric on both sides of the \( \omega \)-axis, that is, \( \lambda \in \lambda(H) \) implies that \(-\lambda \in \lambda(H)\), where \( \lambda(H) \) denotes the spectrum (eigenvalues) of \( H \). Moreover, under the stabilizability and detectability conditions of the pairs \((A, B)\) and \((C, A)\), respectively, \( H \) has no pure imaginary eigenvalues (Wonham, 1968). More on these notions will be given.

Numerical solutions of the AMRE has been a very active area for the past decade. The literature is extensive and it is not our purpose here to conduct a survey on some eight classes of solution methods. Refer to a survey by Jamshidi (1980) in which over 280 references on the solutions of the Riccati and Lyapunov equations are listed.

One of the earlier favorites among direct solution methods of AMRE was through the complete determination of the \( 2n \) eigenvectors of the Hamiltonian matrix \( H \) (Potter, 1966). This method soon turned out to be unattractive numerically and the need for a computationally reliable method for solving AMRE became evident. The answer to this need was a so-called Schur vector decomposition scheme which is, in fact, a variant of the eigenvector method, proposed by Laub (1979). The method provides a basis for a certain subspace of the problem. The method is much more efficient than all the direct methods (e.g., eigenvector), iterative methods (e.g., Newton), and
of course steady-state solution of DMRE through integration. The method has been
coded in almost all CACSD environments regardless of whether the package is based
on MATLAB or not (see also Appendix B). In sequel, a Schur vector transformation
method (Laub, 1979) for the solution of continuous-time AMRE will be briefly
discussed.

Before the solution of Eq. (6.97) via Schur vector transformation method is
presented, consider the following theorem from a classical similarity theory which
is a fundamental issue in modern numerical linear algebra.

**Theorem 6.2.** Let $H$ be a $2n \times 2n$ Hamiltonian (or any general) matrix.
Then there exists an orthogonal similarity transformation matrix $V$ such that $V^T HV$
is quasitriangular. Furthermore, $V$ can be chosen such that the 11 and 22
diagonal blocks appear in any desired order.

In this theorem, let the transformed matrix be represented by

$$V^T HV = U = \begin{pmatrix}
U_{11} & U_{12} \\
0 & U_{22}
\end{pmatrix}$$

where $U_{11} \in \mathbb{R}^{n \times n}$. We shall refer to first $n$ vector of $V$ as the **Schur vector** corresponding
to $\lambda(U_{11}) \subseteq \lambda(H)$. Once again, $\lambda(\cdot)$ represents the set of all eigenvalues. It is possible
to arrange that the real parts of the spectrum of $U_{11}$ are negative, while the real parts
of the spectrum of $U_{22}$ are positive. How this may be accomplished, will be discussed
later. Let a two $n \times n$ partition of $V$ be represented by

$$V = \begin{pmatrix}
V_{11} & V_{12} \\
V_{21} & V_{22}
\end{pmatrix}$$

then we present the main theorem leading to the solution of AMRE Eq. (6.97).

Toward the existence of a nonnegative definite solution of AMRE Eq. (6.70),
note a number of assumptions. First, it is well known that the pairs $(A, G)$ can be
**stabilizable** which is somewhat weaker than the controllability of the same pair, but
often is more difficult to check. Here, $GG^T = S$ and $G$ is said to be a full-rank
factorization of $S$. Moreover, the pair $(F, A)$ can be readily **detectable** in place of
observable, but once again it can not be easily checked as observability (Wonham,
1968). Again, we have $F^TF = Q$ and $F$ is a full-rank factorization of $Q$.

**Theorem 6.3.** Noting these assumptions and conditions of Theorem 6.2 for
the Hamiltonian matrix, the following are valid:

1. $V_{11}$ is nonsingular and $K = V_{21}V_{11}^{-1}$ solves the AMRE Eq. (6.97) with $K =
K^T \geq 0$,
2. The "closed-loop" system spectrum is equal to the spectrum of $V_{11}$, that is,
$\lambda(A - SK) = \lambda(V_{11})$. 
A proof of this theorem can be found in Laub (1979).

From a numerical point of view, two steps are involved in solving AMRE Eq. (6.97) using Schur's method. The first step is to reduce the $2n \times 2n$ Hamiltonian matrix to an ordered real Schur form; and the second is the solution of an $n$th order linear matrix equation. We will briefly discuss the numerical modification of these two steps. To handle the ordering of the Schur form, it is required to calculate and order the real Schur form (RSF) of a real upper Hessenberg matrix in descending order. However, for our purpose, since we need RSF in ascending order, a small modification of the computer code would be necessary. This has been done here. For the second step of computation which involves the solution of $K V_{11} = V_{21}$ to find $K = V_{21} V_{11}^{-1}$, one may use reliable codes for linear equations solver of LINPACK (1979). Since $K$ is symmetric, it is commonly more convenient to solve $V_{11}^T K = V_{21}^T$ to find $K = V_{11}^{-T} V_{21}^T = V_{21} V_{11}^{-1}$. The solution of AMRE is available within CONTROL.lab under the primitive name RKSR, MATRIXx name RICCATI. Some CAD examples for the solutions of matrix Riccati equations follow.

**CAD Example 6.1**

In this example, a time-varying linear MIMO system will be considered. The system is represented by,

$$
\dot{x} = \begin{bmatrix}
0.25t & -t & -e^{-t} \\
0 & te^{-t} & 0 \\
1 & 0.5 & -0.05t^2
\end{bmatrix} x + \begin{bmatrix} t & 0.5 \\
-0.25t & 0 \\
0.1t & 1
\end{bmatrix} u
$$

with output equation

$$
y(t) = \begin{bmatrix} 1 & 2 & \sin(0.25t) \\
-1 & 0.5 & 1
\end{bmatrix} x
$$

and cost functional,

$$
J = \frac{1}{2} \int_0^4 (x^T x + u^T u) \, dt
$$

which indicates that $Q = I_3$ and $R = I_2$. The following solution of the corresponding Riccati equation is obtained using RICRKUT of LSSPAK/PC. Note that the entire trajectories of the resulting Riccati matrix and their polynomial approximation coefficients have been saved on two data files called RIC_6X21.DAT and RIC_6X3.DAT. One of these files will be used later.

<<RICRKUT>> Solve Differential Matrix Riccati Equation with both constant and time-varying matrix coefficients via Runge-Kutta integration. The program allows for saving of the Riccati matrix elements trajectories and a polynomial approximation for the Riccati matrix.

For an equation of the form:

$$
- \frac{dK(t)}{dt} = A(t)K(t) + K(t)A(t) - K(t)B(t)R(t)^{-1}B^*(t)K(t) + Q(t)
$$

with final condition $K(t_f)$, determine $K(t)$, to $\leq t \leq t_f$
Order of Matrix A = n = 3  
Order of Matrix B = m = 2  
Initial Time t0 = 0  
Final Time tf = 4  
Step Size dt = 2.

K(tf):
1  0.0E + 0
2  0.0E + 0
3  0.0E + 0
4  0.0E + 0
5  0.0E + 0
6  0.0E + 0

At Time t = 0 Riccati Matrix K(t):
3.907E-01  1.833E-01  2.257-02
1.833-01  1.578E-00  6.505E-02
2.257E-02  6.505E-02  6.912E-02

At Time t = .2 Riccati Matrix K(t):
4.215E-01  2.232E-01  2.382E-02
2.232E-01  1.765E-00  7.024E-02
2.382E-02  7.024E-02  7.118E-02

At Time t = 3.8 Riccati Matrix K(t):
1.983E-01  -3.247E-03  -1.736E-02
-3.247E-03  2.047E-01  3.759E-04
-1.736E-02  3.759E-04  2.079E-03

At Time t = 4.0 Riccati Matrix K(t):
0.000E +00  0.000E +00  0.000E +00
0.000E +00  0.000E +00  0.000E +00
0.000E +00  0.000E +00  0.000E +00

Would you like to SAVE the resulting Riccati matrix solution to disk?  
(Y/N) y

Enter Riccati matrix Storage Code (1 . . . Time-varying  
trajectory) or (2 . . . Steady - state value) (1/2):1
Save Time-varying Trajectory . . . . . . . . . K(t), to <= t <= tf . . .
Please NOTE that the integration parameters are:
Initial Time t0 = 0 Final Time tf = 4 Step Size dt = .2
Please ENTER DISK Drive Identifier (A,B,C,D) b
DATA DISK SHOULD BE IN DRIVE B
CORRECT (Y/N)? y
Sec. 6.2 Solution of the Optimal Control Problem

Please enter the matrix's disk file name in the form
NNNNNNNN.SSS (No Quotes)

RIC_6x21.DAT
Matrix Riccati Matrix Trajectory K(t) saved to disk file
B:RICI_6x21.DAT
Would you like to compute a polynomial approximation for the
RICCATI matrix coefficients (Y/N)? Y
Input polynomial order (order < 6)? 2

THE POLYNOMIAL COEFFICIENT A(0) + A(1)*T + ... + A(2)*T**2.
FOR THE RICCATI MATRIX ELEMENTS ARE
K(1,1) 0.2750 0.5050 -0.1362
K(1,2) 0.4103 -0.2386 0.0198
K(1,3) 0.0882 -0.1364 0.0245
K(2,2) 1.6521 1.7482 -0.5449
K(2,3) 0.0389 0.1165 -0.0299
k(3,3) 0.0552 0.0530 -0.0158

Please enter the matrix's disk file name in the form
NNNNNNNN.SSS (No Quotes)
RIC_C6x3.DAT
Matrix Riccati Matrix Trajectory K(t) saved to disk file
B:RIC_C6x3.DAT

Below is a listing of Subroutine "ABQRMS" which must be defined by the user to be
chain merged with RICRKUT:

30000 'SUB ABQRMS (A(*),B(*),Q(*),R(*),S(*),T,N,M)
30010 FOR I1 = 1 TO N: FOR J1 = 1 TO N: A(I1,J1) = 0: NEXT: NEXT
30020 FOR I1 = 1 TO N: FOR J1 = 1 TO N: B(I1,J1) = 0: NEXT: NEXT
30030 FOR I1 = 1 TO N: FOR J1 = 1 TO N: Q(I1,J1) = 0: NEXT: NEXT
30040 A(1,1) = .25*X: A(1,2) = X: A(2,2) = .5: A(3,3) = -.05*X**2
30050 A(1,3) = -EXP(-X): A(2,2) = X*EXP(-X): A(3,1) = 1
30060 B(1,1) = X: B(1,2) = .5: B(3,2) = 1: B(3,1) = .1: X = B(2,1) = -.25*X
30080 R(1,1) = 1: R(2,2) = 1
30090 C(1,1) = 1: C(1,2) = 2: C(1,3) = SIN (.25*X)
30100 C(2,1) = -1: C(2,2) = .5: C(2,3) = 1
30110 RETURN

CAD Example 6.2
In this example, a second-order linear time-invariant system's algebraic Riccati equation
is solved by four different CAD programs in LSSPAK/PC, CONTROLlab and MATRIXx
utilizing standard integration and Schur vector transformation methods. The system was
already considered in Ex. 6.6 where the solution was obtained analytically.
Solution 1. RICRKUT on LSSPAK/PC

<<RICRKUT>> Solve Differential matrix Riccati Equation via Runge-Kutta integration including a polynomial approximation for the Riccati matrix. For an equation of the form:

dK(t)/dt = A*K + KA - KBR - 1B'K + C, with Final condition K(tf)

Order of Matrix A - n = 2
Order of Matrix B - m = 1
Initial Time to = 0
Final time tf = 2
Step size dt = .1

Matrix A

0.000E+00  0.100E+01
0.000E+00  0.000E+00

Matrix B

0.000E+00
0.100E+01

Matrix Q

0.200E+01  0.100E+01
0.100E+01  0.400E+01

Matrix R

0.500E+00

K(tf):

1  0.000E+00
2  0.000E+00
3  0.000E+00

At Time t = 0 Riccati Matrix K(t):

2.217E+0  9.105E-01
9.105E-0  1.700E+00

At Time t = .2 Riccati Matrix K(t):

2.139E+0  8.829E-01
8.829-01  1.690E+00

K(tf):

1  2.217E+00
2  9.105E-01
3  1.700E+00
At Time $t = 0$ Riccati Matrix $K(t)$:

\[
egin{align*}
2.450E+00 & \quad 9.949E-01 \\
9.949E-01 & \quad 1.730E+00
\end{align*}
\]

At Time $t = .1$ Riccati Matrix $K(t)$:

\[
egin{align*}
2.448E+00 & \quad 9.941E-01 \\
9.941E-01 & \quad 1.730E+00
\end{align*}
\]

At Time $t = .2$ Riccati Matrix $K(t)$:

\[
egin{align*}
2.445E+00 & \quad 9.932E-01 \\
9.932E-01 & \quad 1.730E+00
\end{align*}
\]

Note that this use of RICRKUT represents two successive numerical integrations of the DMRE of Ex. 6.9. The answer corresponds closely with what was obtained for Ex. 6.7.

**Solution 2. AMRE on CONTROL.lab**

\[
\begin{align*}
\diamond a & = <0 \ 1 ; \ 0 \ 0 > ; \ b = <0 ; \ 1 > \ q = <2 \ 1 ; \ 1 \ 4 > ; \ r = 1/2 ; \\
\diamond y & = <0.5 \ .1 \ .000001 > ; \ x = <0 \ 0 \ 0 > ; \\
\diamond K & = \text{amre}<z,b,q,r,x,y> \\
K & = \\
2.4609 & \quad 0.9988 \\
0.9988 & \quad 1.7316
\end{align*}
\]

**Solution 3. RKSR on CONTROL.lab**

\[
\begin{align*}
\diamond \text{help rksr} \\
\text{RKSR} & \quad \text{Using a certain set of Schur vectors "RKSR" solves the} \\
& \quad \text{continuous-time algebraic Riccati equation} \\
& \quad KA + A^T K - KB \cdot \text{inv}(R) \cdot B^T K + Q = 0 \\
& \quad \text{It is assumed that } (A,B) \text{ is stabilizable and } B \text{ is full rank under these assumptions,} \\
& \quad \text{RKSR}(A,B,Q,R,\ldots) \text{ can solve algebraic Riccati equation.} \\
\diamond K & = \text{rksr}(a,b,q,r) \\
K & = \\
2.4641 & \quad 1.0000 \\
1.0000 & \quad 1.7321
\end{align*}
\]

**Solution 4. RICCATI on MATRIXx**

\[
\begin{align*}
\diamond a & = <0 \ 1 ; \ 0 \ 0 > ; \ b = <0 ; \ 1 > ; \ r = 1/2 ; \ q = <2 \ 1 ; \ 1 \ 4 > ; \\
\diamond K & = \text{Riccati}(a,b,q,r) \\
\diamond K & = \\
2.4641 & \quad 1.0000 \\
1.0000 & \quad 1.7321
\end{align*}
\]
The following is another CAD example using PC-MATLAB:

**CAD Example 6.3**

In this CAD Example, PC_MATLAB's LQR primitive is used to design a state regulator for a fifth order system with three inputs.

```matlab
>> a = [-.4 .2 .6 .1 -.2; 0 -.5 0 0.4; 0 0 -2 0.2; .2 1 .5 -1.25 0; .25 0 -.2 .5 -1];
>> b = [1 -1 0; 2 1 0; 0 0 1; 0 0 -2; 0 0 1];
>> q = diag([1 1 2 5 10]); r = eye(3);
>> [g,k] = lqr(a,b,q,r)

\[
\begin{bmatrix}
g & & \\
0.3594 & 0.7199 & 0.1193 & 0.4472 & 0.7083 \\
-0.3576 & 0.3591 & -0.1100 & -0.3466 & -0.5306 \\
-0.0573 & -0.0048 & 0.1081 & -1.4902 & 1.1197 \\
\end{bmatrix}
\]

\[
k = \\
\begin{bmatrix}
0.3582 & 0.0006 & 0.1131 & 0.3802 & 0.5899 \\
0.0006 & 0.3597 & 0.0031 & 0.0335 & 0.0592 \\
0.1131 & 0.0031 & 0.5850 & 0.2567 & 0.0365 \\
0.3802 & 0.0335 & 0.2567 & 1.5763 & 1.4057 \\
0.5899 & 0.0592 & 0.0365 & 1.4057 & 3.8945 \\
\end{bmatrix}
\]

Note that the defined g matrix is \( g = r^{-1}b^T \) and k is the Riccati matrix.

The next CAD example considered here is a linear state regulator corresponding to the third order system of CAD Ex. 6.1. Here we take advantage of the solution of the time-varying differential matrix Riccati equation which was already saved in an ASCII file, called RIC_ 6 \times 21. DAT and find the optimal (LQ) regulator problem using TIMDOM/PC.

**CAD Example 6.4**

\[ \text{<<TVSTRG>>} \text{ Solves the Time-Varying State Regulator Problem:} \]
\[ \frac{dx}{dt} = A(t)x + B(t)u, \ x(0) = x_0 \ \text{Output} \ y = C(t)x \]

\[ \text{Minimize:} \]
\[ J = \frac{1}{2} \int [x'Q(t)x + u' \ R(t)u]dt \]

\[ \text{via Riccati Formation and Runge-Kutta Integration:} \]

Differential Matrix Riccati Eq.:  
\[ dK(t)/dt = A'(t)K(t) + K(t)A(t) - K(t)B(t)R(t)B(t)'K(t) - 1B(t)'K(t) + Q(t) \]

\[ \text{can be either solved, retrieved or approximated from existing DATA files on your data disk.} \]

Order of Matrix \( A - n = 3 \)
Order of Matrix \( B - m = 2 \)
Initial Time to = 0
Final Time \( tf = 4 \)
Step Size \( dt = .2 \)
Sec. 6.2 Solution of the Optimal Control Problem

Elem. × (0):
1  1.000E+00
2  0.000E+00
3  -1.000E+00

At Time t = 0 Riccati matrix K(t):
3.9068E−01  1.8330E−01  2.256E−02
1.8330E−01  1.5779E+00  6.5055E−02
2.256E−02   6.5055E−02  6.9116E−02

At Time t = .2 Riccati matrix K(t):
4.2150E−01  2.2321E−01  2.3818E−02
2.2321E−01  1.7646E+00  7.0243E−02
2.3818E−02  7.0243E−02  7.1179E−02
.
.
.

At Time t = 4.0 Riccati matrix K(t):
0.0000E+00  0.0000E+00  0.0000E+00
0.0000E+00  0.0000E+00  0.0000E+00
0.0000E+00  0.0000E+00  0.0000E+00

STATE REGULATION PROBLEM SOLUTION – TIME VARYING CASE

*** State Variables ***

At t = 0 x(t) is:
1.0000E+00
0.0000E+00
-1.0000E+00

At t = .2 x(t) is:
1.1503E+00
2.6254E−04
-8.2464E−01
.
.
.

At t = 3.8 x(t) is:
7.6694E−02
1.8845E−01
6.5598E−01
At $t = 4$ $x(t)$ is:
- $2.9430E-02$
- $1.8250E-01$
- $5.8888E-01$

*** Output Variables ***

At $t = 0$ $y(t)$ is:
- $1.0000E+00$
- $-2.0000E+00$

At $t = .2$ $y(t)$ is:
- $1.1096E+00$
- $-1.9748E+00$
- 

At $t = 3.8$ $y(t)$ is:
- $9.8717E-01$
- $6.7351E-01$

At $t = 4$ $y(t)$ is:
- $8.3110E-01$
- $7.0956E-01$

*** Variable ***

At $t = 0$ $u(t)$ is:
- $0.0000E+00$
- $-1.3751E-01$

At $t = .2$ $u(t)$ is:
- $-8.2466E-02$
- $-2.0136E-01$
- 

At $t = 3.8$ $u(t)$ is:
- $2.4398E-02$
- $-1.7099E-03$

At $t = 4$ $u(t)$ is:
- $2.12486E-02$
- $-0.82492E-03$

Optimum value of the performance Index = 1.549399
Figure 6.7 shows time trajectories of the system’s three states, two outputs, and two control signals. As this system is time-varying, the behavior of the system’s variables is somewhat unpredictable.

### 6.3 THE DISCRETE-TIME MAXIMUM PRINCIPLE

Our discussion so far in this chapter has been devoted to the optimal control of continuous-time systems. The minimum principle, discussed in Sec. 6.2.2, is extended to discrete-time systems in this section. Discrete maximum principle can be considered as an extension to the digital control systems design. In a strict sense, the application of discrete maximum principle reduces to an investigation of the system’s convexity. The presentation of this subject here lacks a great deal of abstract mathematics and instead, a working knowledge of the subject would be given.

Consider a nonlinear discrete-time system described by

$$ x(k + 1) = f[x(k), u(k), k] $$

(6.98)

with initial state

$$ x(k_0) = x_0 $$

(6.99)

and an objective function

$$ J = F[x(k_f), k_f] + \sum_{k = k_0}^{k_f} L[x(k), u(k), k] $$

(6.100)

The term $F[x(k_f), k_f]$ is known as the terminal cost. This term would be required, as the terminal condition, if $x(k_f)$ is not fixed. The design problem is to find the optimal control $u^*(k)$ on $[k_0, k_f]$ such that the performance index Eq. (6.100) is minimized, subject to the equality constraints Eqs. (6.98) and (6.99).

Defining an $n \times 1$ costate vector $p(k)$, the above optimization problem is equivalent to minimizing,

$$ J_1 = F[x(k_f), k_f] + \sum_{k = k_0}^{k_f-1} \{L[x(k), u(k), k]

+ p^T(k + 1) [x(k + 1) - f[x(k), u(k), k]] } $$

(6.101)

Now, let us define a scalar Hamiltonian function

$$ H[x(k), u(k), p(k + 1), k] = L[x(k), u(k), k] - p^T(k + 1)f[x(k), u(k), k] $$

(6.102)

For maximum principle, the Hamiltonian must be maximum along the optical trajectory. Substituting $H(\cdot)$ of Eq. (6.102) into Eq. (6.101) leads to,

$$ J_1 = F[x(k_f), k_f] + \sum_{k = k_0}^{k_f-1} \{H[x(k), u(k), p(k + 1), k] - p^T(k + 1)x(k + 1) } $$

(6.103)
Figure 6.7 Optimal trajectories for system of CAD Example 6.4.
(a) State vector.  (b) Output vector.  (c) Control vector.
which is the maximum principle. Let the state vectors $x(k), x(k+1)$, and control vector $u(k)$ and costate vector $p(k+1)$ have the following variations,

$$x(k) = x^*(k) + \epsilon \xi(k)$$  \hspace{1cm} (6.104)

$$x(k+1) = x^*(k+1) + \epsilon \xi(k+1)$$  \hspace{1cm} (6.105)

$$u(k) = u^*(k) + \delta \eta(k)$$  \hspace{1cm} (6.106)

$$p(k+1) = p^*(k+1) + \gamma \mu(k+1)$$  \hspace{1cm} (6.107)

Now, Eq. (6.101) can be written as,

$$J_1 = F[x^*(k_f) + \epsilon \xi(k_f), k_f] + \sum_{k=0}^{k_f-1} H[x^*(k) + \epsilon \xi(k), u^*(k)$$

$$+ \delta \eta(k), p^*(k+1) + \gamma \mu(k+1), k]$$

$$- [p^*(k+1) + \gamma \mu(k+1)]^T [x^*(k+1) + \epsilon \xi (k+1)]$$  \hspace{1cm} (6.108)

Expanding $F[x(k_f), k_f]$ into a Taylor series about $F[x^*(k_f), k_f]$ yields,

$$F[x(k_f), k_f] = F[x^*(k_f), k_f] + \epsilon \xi^T (k_f) F_{x^*}^* + \cdots$$  \hspace{1cm} (6.109)

where $F_{x^*}^* \triangleq \partial F^*/\partial x^*$. Similarly, the Hamiltonian function can be expanded into a Taylor series about $x^*(k), u^*(k), p^*(k+1)$ and $x^*(k+1)$,
\[ H[x(k), u(k), p(k + 1), k] = H[x^*(k), u^*(k), p^*(k + 1), k] + \varepsilon \xi^T(k)H^*_{x^*}(k) + \delta \eta^T(k)H^*_{u^*}(k) + \gamma \mu^T(k + 1)H^*_{p^*}(k + 1) + \cdots \]  

(6.110)

where

\[ H^*(k) = H[x^*(k), u^*(k), p^*(k + 1), k] \]  

(6.111)

Substituting Eqs. (6.110) and (6.111) into Eq. (6.103) and performing the following necessary conditions for minimizing \( J_1 \)

\[ \frac{\partial J_1}{\partial \epsilon} \bigg|_{\epsilon=\delta=\gamma=0} = 0 \]  

(6.112)

\[ \frac{\partial J_1}{\partial \delta} \bigg|_{\epsilon=\delta=\gamma=0} = 0 \]  

(6.113)

\[ \frac{\partial J_1}{\partial \gamma} \bigg|_{\epsilon=\delta=\gamma=0} = 0 \]  

(6.114)

one has

\[ \zeta^T(k_f)F^*_{x^*}(k_f) + \sum_{k=k_0}^{k_f-1} \zeta^T(k)H^*_{x^*}(k) - \sum_{k=k_0}^{k_f-1} p^*^T(k + 1)\zeta(k + 1) = 0 \]  

(6.115)

\[ \eta^T(k)H^*_{u^*}(k) = 0 \]  

(6.116)

\[ \mu^T(k + 1)H^*_{p^*}(k + 1) - x^*(k + 1) = 0 \]  

(6.117)

The last equation leads to

\[ x^*(k + 1) = H^*_{p(k + 1)}(k) \]  

(6.118)

which is the original state Eq. (6.98). Eq. (6.116) yields,

\[ H^*_{u}(k) = 0 \]  

(6.119)

which represents the necessary condition for the maximization (or minimization) of the Hamiltonian function along the optimal trajectory with respect to the optimal control.

The final set of necessary conditions for optimality stems from the last term on the left-hand side of Eq. (6.115) which can be written as

\[ \sum_{k=k_0}^{k_f-1} p^*^T(k + 1)\zeta(k + 1) = \sum_{k=k_0+1}^{k_f-1} p^*^T(k)\zeta(k) \]

\[ = \sum_{k=k_0}^{k_f-1} p^*^T(k)\zeta(k) + p^*^T(k_f)\zeta(k_f) - p^*^T(k_0)\zeta(k_0) \]  

(6.120)
However, since \( x(k_0) \) is known, \( \zeta(k_0) = 0 \), then Eq. (6.120) becomes
\[
\sum_{k=k_0}^{k_r-1} p^T(k+1)\zeta(k+1) = \sum_{k=k_0}^{k_r-1} p^T(k)\zeta(k) + p^T(k_r)\zeta(k_r) \tag{6.121}
\]

Now, by substituting this equation into Eq. (6.115) and rearranging terms, one gets
\[
[F^*(k_r) - p^*(k_r)]^T\zeta(k_r) + \sum [H^*(k) - p^*(k)]^T\zeta(k) = 0 \tag{6.122}
\]

Since the variations of \( p(k) \) and \( p(k_r) \) are mutually independent, the only way to satisfy Eq. (6.122) is,
\[
p^*(k_r) = F^*(k_r) \tag{6.123}
\]
\[
p^*(k) = H^*(k) \tag{6.124}
\]

To summarize, the necessary conditions for the discrete-time maximum principle can be given by,
\[
p^*(k) = \frac{\partial H^*(k)}{\partial x^*(k)} \tag{6.125}
\]
\[
p^*(k_r) = \frac{\partial F^*(k_r)}{\partial x^*(k_r)} \tag{6.126}
\]
\[
x^*(k+1) = \frac{\partial H^*(k)}{\partial p^*(k+1)} \tag{6.127}
\]
\[
x^*(k_0) = x_0, \quad \text{known} \tag{6.128}
\]
\[
H^*(k) = 0 \tag{6.129}
\]

Equations (6.125) to (6.129) represent a 2n-dimensional discrete-time TPBV problem and are refined to as canonical state difference equations. Equation (6.129) provides a relation to find the optimal control, while relations Eqs. (6.128) and (6.126) represent the initial and transversality conditions, respectively. It is noted that if any component of the final state \( x(k_r) \) is fixed, the corresponding transversality condition of \( p^*(k_r) \) no longer applies.

The following example illustrates the application of the discrete maximum principle.

**Example 6.10**

It is desired to find the optimal control \( u^*(k) \), \( k = 0, 1, \ldots, 4 \), such that the performance index
\[
J = \frac{1}{2} \sum_{k=0}^{4} [x^2(k) + 2u^2(k)]
\]

is minimized, subject to the system state equation
\[
x(k+1) = -x(k) + 2u(k)
\]
The initial state is \( x(0) = 1 \) and the final state is \( x(5) = 0 \).

**Solution** The discrete TPBV problem is obtained from Eqs. (6.125) and (6.127), which is

\[
p^*(k + 1) = x^*(k) - p^*(k) \tag{6.130}
\]

\[
x^*(k + 1) = -3x^*(k) + 2p^*(k) \tag{6.131}
\]

From Eq. (6.129), the optimal control is given by

\[
u^*(k) = -p^*(k + 1) \tag{6.132}
\]

Solving for the canonical Eqs. (6.130) and (6.131), one has,

\[
x(k) = 0.21(1 + 2.753p_0)(-0.268)^k \tag{6.133}
\] \[+ 0.79(1 - 0.731p_0)(-3.732)^k\]

\[
p(k) = 0.289(1 + 2.726p_0)(-0.268)^k \tag{6.134}
\] \[ - 0.289(1 - 0.733p_0)(-3.732)^k\]

where \( p(0) = p_0 \) is the initial value of the costate variable. Note that since the end point \( x(5) \) is fixed, the transversality condition Eq. (6.126) is not needed. Now, using the end condition \( x(5) = 0 \) in Eq. (6.127), \( p_0 \) is obtained as \( p_0 = 1.368 \). Using this value, Eqs. (6.133) and (6.132), we have,

\[
x^*(k) = 1.0009(-0.268)^k \]

\[
u^*(k) = -1.368(-0.268)^k
\]

These represent the desired solution.

Here again, a special analytical case would arise when the system Eq. (6.98) is linear time-invariant (or time-varying) and the object function Eq. (6.100) is quadratic.

Consider,

\[
x(k + 1) = A(k)x(k) + B(k)u(k), \quad x(k_0) = x_0 \tag{6.135}
\]

\[
J = \frac{1}{2} x^T(k_f)F(k_f)x(k_f) + \frac{1}{2} \sum_{k = k_0}^{k_f - 1} [x^T(k)Q(k)x(k) + u^T(k)R(k)u(k)] \tag{6.136}
\]

where \( Q(k) \) and \( F(k) \) are nonnegative definite and \( R(k) \) is positive definite matrix function of \( k \), respectively. In order to solve this problem, let us define the Hamiltonian,

\[
H[x(k), u(k), p(k + 1), k] = \frac{1}{2} x^T(k)Q(k)x(k) + \frac{1}{2} u^T(k)R(k)u(k) + p^T(k + 1) [A(k)x(k) + B(k)u(k)]
\]

Applying the necessary conditions Eqs. (6.125) to (6.129) for optimality would lead to
\[ p(k) = Q(k)x(k) + A^T(k)p(k + 1) \quad (6.137) \]
\[ p(k_f) = Fx(k_f) \quad (6.138) \]
\[ x(k + 1) = A(k)x(k) + B(k)u(k) \quad (6.139) \]
\[ x(k_0) = x_0 \quad (6.140) \]
\[ 0 = R(k)u(k) + B^T(k)p(k + 1) \quad (6.141) \]

where the subscript * has been eliminated for simplicity. Once, the control \( u(k) \) is eliminated from Eq. (6.141)
\[ u(k) = -R^{-1}(k)B^T(k)p(k + 1) \quad (6.142) \]
and substituting it into Eq. (6.139) leads to
\[ x(k + 1) = A(k)x(k) - S(k)p(k + 1) \quad (6.143) \]
where \( S(k) \triangleq B(k)R^{-1}(k)B^T(k) \). Equation (6.143) along with Eq. (6.137) constitutes the TPBV problem for the so-called discrete-time state regulator problem. As in the continuous state regulator problem (Sec. 6.2.2), one can assume that,
\[ p(k) = K(k)x(k) \quad (6.144) \]
and eliminating \( p(\cdot) \) in Eqs. (6.143) and (6.137) leads to,
\[ x(k + 1) = A(k)x(k) - S(k)K(k + 1)x(k + 1) \quad (6.145) \]
\[ K(k)x(k) = Q(k)x(k) + A^T(k)K(k + 1)x(k + 1). \quad (6.146) \]

Now, eliminating \( x(k + 1) \) from Eq. (6.145), leads Eq. (6.146) to
\[ K(k)x(k) = Q(k)x(k) + A^T(k)K(k + 1)[I + S(k)K(k + 1)]^{-1}A(k)x(k) \quad (6.147) \]
where \( I \) is an \( n \times n \) identity matrix. For any arbitrary value of \( x(k) \), this equation would hold only if
\[ K(k) = Q(k) + A^T(k)K(k + 1)[I + S(k)K(k + 1)]^{-1}A(k) \quad (6.148) \]
with final condition
\[ K(k_f) = F \quad (6.149) \]
in view of Eqs. (6.138) and (6.144). Equation (6.148) is known as the discrete-time matrix Riccati equation which is equivalent to continuous-time differential matrix Riccati Eq. (6.43). Note that once the sequence of Riccati matrices \( K(k), k = k_0, \ldots, k_f \) is known, the optimal control Eq. (6.142) can be easily found to be
\[ u^*(k) = -R^{-1}(k)B^T(k)K(k + 1)x^*(k + 1) \quad (6.150) \]
which can also be represented in terms of \( x^*(k) \) only as
\[ u^*(k) = -[I + G(k)B(k)]^{-1}G(k)A(k)x^*(k) \quad (6.151) \]
where $G(k) \triangleq R^{-1}(k)B^T(k)K(k + 1)$ is an $m \times n$ feedback matrix. Note that for the time-invariant case, the Riccati Eq. (6.148) would be reduced to

$$K = Q + A^T K(I + SK)^{-1} A$$

(6.152)

which can be solved for $K$ using the discrete-time formulation of the Schur's method discussed in Sec. 6.2.3 and is used in both CONTROL. lab and MATRIXx.

Example 6.10 discussed earlier in fact represents a linear discrete-time state regulator problem. In sequel, a CAD example is presented using commands "REGULATOR" and "DEREGULATOR" of MATRIXx on an IBM PC/AT to optimize a fourth-order system.

**CAD Example 6.5**

Consider an inverted pendulum system shown in Fig. 6.1. The pendulum's nonlinear state model is given in Eq. (6.21). If the angle $\theta$'s variation is small, the model can be linearized by replacing sine of the angle by the angle itself and the cosine by one, that is, $\sin(x) = x$ and $\cos(x) \approx 1$. Thus, Eq. (6.19) would become

$$l \ddot{\theta} + \ddot{\theta} - g \theta = 0$$

(6.153)

and by virtue of the state variables definitions in Eq. (6.20) and the first three equations of Eq. (6.21), the linearized model for the pendulum will be

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{B}{M} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{-g}{l_e} & 0 & \frac{g}{l_e} & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ 0 \end{bmatrix} u$$

(6.154)

Then assuming that $B = M = 1$, $g = 9.8$ and $l_e = 0.84$, the following fourth order linear continuous-time system results

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -11.65 & 0 & 11.65 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u$$

Assuming that a cost function is given by

$$J = \frac{1}{2} \int_0^\infty [2x_1^2(t) + ru^2(t)] dt$$

where $r = 2 \times 10^{-6} m^2/N^2$. The first part of this CAD example constitutes a linear continuous-time state regulator problem and then one finds the impulse response of the output. The actual MATRIXx commands listing is given below:
\[ a = \begin{bmatrix} 0 & 1 & 0 & 0; & 0 & -1 & 0 & 0; & 0 & 0 & 0; & -11.65 & 0 & 11.65 & 0 \end{bmatrix}; \]
\[ b = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}; \]
\[ q = \text{diag } (\begin{bmatrix} 0 & 0 & 2 & 0 \end{bmatrix}); \]
\[ r = 2e-6; \]
\[ \text{eval, } f, k > = \text{regulator } (a, b, q, r); \]
\[ // \text{Riccati Matrix} \]
\[
K =
\begin{bmatrix}
0.0151 & 0.0008 & -0.0709 & -0.0130 \\
0.0008 & 0.0001 & 0.0028 & 0.0006 \\
-0.0709 & -0.0028 & 0.6358 & 0.0798 \\
-0.0130 & -0.0006 & 0.0798 & 0.0128
\end{bmatrix}
\]
\[ // \text{matrix of State Feedback Gains} \]
\[
F =
\begin{bmatrix}
1.0D+03 & * \\
0.3890 & 0.0269 & -1.3890 & -0.2824
\end{bmatrix}
\]
\[ // \text{Vector of closed-loop eigenvalues} \]
\[
\text{EVAL} =
\begin{bmatrix}
-4.0850 + .3293i \\
-4.0850 - .3293i \\
-9.8696 + .8613i \\
-9.8696 - .8613i
\end{bmatrix}
\]

Defining an appropriate output matrix and performing an impulse response for the optimal closed-loop system we have
\[
\dot{x} = (A - SK)x + Bu, \quad x(0) = x_0
\]
\[
y = Cx, \quad S = BR^{-1}B^T
\]

In sequel, the closed-loop system's compound matrix, using \text{MATRXx}'s notation, is
\[
S_c = \begin{bmatrix}
A - SK & B \\
C & 0
\end{bmatrix}
\]

will be used for transient response purposes.
The impulse response of the optimal cart displacement and pendulum angular position are shown in Fig. 6.8.

Next, the inverted pendulum system was discretized using MATRIXx (at sampling time of 0.05 sec.), optimized via discrete state regulator problem, and simulated via an impulse response.

```matlab
<> sd = discretize(sc,4,.05);
<> [ad,bd] = split(sd,4)

BD =
  0.0012
  0.0488
  0.0000
-0.0002

AD
  1.0000   0.0488   0.0000   0.0000
  0.0000   0.9512   0.0000   0.0000
-0.0146 -0.0002  1.0146   0.0502
-0.5853 -0.0144  0.5853  1.0146
<> [evald,fd,kd] = dregulator(ad,bd,q,r);
<> evald

EVALD =
  0.7293 + 0.3668i
  0.7293 - 0.3668i
  0.5986 + 0.1179i
  0.5986 - 0.1179i
<> fd

FD =
  252.5516  19.9265 -763.2420 -166.1825
<> kd

KD =
  0.3220   0.0166  -1.4981  -0.2753
  0.0166   0.0011  -0.0597  -0.0121
-1.4981  -0.0597  14.0515  1.6520
-0.2753  -0.0121  1.6520   0.2681
<> sdd = bd*inv(r)*bd;
<> amskd = ad - sdd*kd;
<> sdcl = amskd/bd;c,d>
<> //check the eigenvalues of closed-loop system
<> eig(splid(sdcl,4))
```
Figure 6.8  (a) Cart displacement impulse response.  
(b) Pendulum angle impulse response.
6.4 AN OPTIMAL POLE PLACEMENT DESIGN

In Chaps. 4 and 5, several classes of multivariable control systems design techniques have been presented. In this section, an effort is made to relate the pole placement design to the optimal control design.

Consider a single-input, controllable linear-time invariant system,

$$\dot{x} = A x + B u$$  

with the performance index

$$J[x(t_0), u(\cdot), t_0] = \int_{t_0}^{\infty} e^{2\alpha t}(x^TQx + u^TRu) \, dt$$  

with the usual conditions on $Q$ and $R$, including that $(A, D)$ be completely observable such that $DD^T = Q$ and $\alpha$ is a positive scalar. The unique positive-definite solution of the corresponding Riccati equation is $K_\alpha$, satisfying

$$K_\alpha (A + \alpha I) + (A^T + \alpha I)K_\alpha - K_\alpha BR^{-1}B^T K_\alpha + Q = 0$$  

and the optimal control law is

$$u = F_\alpha x = -R^{-1}B^T K_\alpha x$$  

where $K_\alpha \triangleq K(\alpha)$ and $K_0 \triangleq K(0)$. It follows from Eqs. (6.157) and (6.158) that

$$K_0(sI - A) + (-sI - A^T)K_0 + F_0 R F_0^T = Q$$
Figure 6.9  (a) Cart displacement impulse response—discrete.
(b) Pendulum angle impulse response—discrete.
Now, multiplying on the left by \( R^{-1/2}B^T(-sI - A)^{-1} \), and on the right by \((sI - A)^{-1}BR^{-1/2}\), adding \(I\) to each side, and observing that \(K_0BR^{-1/2} = -F_0R^{-1/2}\), one gets
\[
[I - F_0^T(-sI - A)^{-1}B][I - F_0^T(sI - A)^{-1}B] = I + B^T(-sI - A)^{-1}Q(sI - A)^{-1}B \tag{6.159}
\]

For a single-input system and an \(R = I\), without any loss of generality, Eq. (6.159) becomes,
\[
[1 - F_0^T(-sI - A)^{-1}B][1 - F_0^T(sI - A)^{-1}B] = 1 + B^T(-sI - A)^{-1}Q(sI - A)^{-1}B \tag{6.160}
\]

In sequel, a general property of the closed-loop poles is exploited which would find an optimal control for a linear single-input time-invariant system without the solution of the matrix Riccati equation.

Consider the completely controllable single-input linear time-invariant system Eq. (6.155) with a quadratic cost functional,
\[
J[x(t_0), u(\cdot), t_0] = \int_{t_0}^{\infty} [u^2 + x^TQx] \, dt \tag{6.161}
\]
where, once again, \(Q = DD^T\) for observable pair \((A, D)\). Under the controllability and observability assumptions, the optimal closed-loop law \(u = Fx\) would establish an asymptotically stable closed-loop system which would satisfy the relation in Eq. (6.160).

Let \(p(s) = \text{det}(sI - A)\) denote the characteristic polynomial of the open-loop system. Then, let us define the polynomial \(\phi(s)\) by
\[
\frac{\phi(s)}{p(s)} = 1 - F^T(sI - A)^{-1}B \tag{6.162}
\]
and polynomial \(q(s)\) by
\[
\frac{q(s)}{p(-s)p(s)} = B^T(-sI - A)^{-1}Q(sI - A)^{-1}B \tag{6.163}
\]
Note that as \(s\) goes to infinity, the right-hand side of this equation is zero, implying that degree of \(q(s)\) is less than twice that of \(p(s)\) which has degree \(n\). On the other hand, by virtue of Eq. (6.162), \(\phi(s)\) and \(p(s)\) have the same order. Moreover, \(p(s)\) is monic (i.e., its highest order is \(s^n\)) and so is \(\phi(s)\).

Now, observe that
\[
[1 - F^T(sI - A)^{-1}B][1 + F^T(sI - A - G^TF^T)^{-1}B] = 1 - F^T(sI - A)^{-1}B \\
+ F^T(sI - A - G^TF^T)^{-1}B - F^T(sI - A)^{-1}[(sI - A) \\
- (sI - A - G^TF^T)(sI - A - B^TF^T)^{-1}
\]
Hence, the inverse of Eq. (6.162) would give,
\[
\frac{p(s)}{\phi(s)} = 1 + F^T(sI - A - B F^T)^{-1}B
\]

and this relation implies that

\[
\phi(s) = \det(sI - A - B F^T)
\] (6.164)

Thus, \(\phi(s)\) is the characteristic polynomial of the closed-loop system \(\dot{x} = (A + B F^T)x\). Since the closed-loop system is asymptotically stable, all the roots of \(\phi(s) = 0\) are in the left-half plane.

Relations Eqs. (6.160), (6.162), and (6.163) are combined to yield

\[
\frac{\phi(-s) \phi(s)}{p(-s) p(s)} = 1 + \frac{q(s)}{p(-s) p(s)}
\] (6.165)

which can also be written as,

\[
\phi(-s) \phi(s) = p(-s) p(s) + q(s)
\] (6.166)

This equation can be used to determine the optimal control feedback gain \(F\), when \(A, B,\) and \(Q\) are known. In this relation, all the quantities in the right-hand side are known, and hence \(\phi(-s) \phi(s)\) is known. Moreover, since \(\phi(s)\) is monic and its roots have negative real parts, then \(\phi(s)\) would be uniquely defined. Once \(\phi(s)\) is defined, Eq. (6.162) can be used to find \(F\). The following algorithm summarizes the calculation of \(F\).

**Algorithm 6.1**

1. Form \(p(s) = \det(sI - A)\) and \(q(s) = B^T(-sI - A^T)^{-1}Q(sI - A)^{-1}B p(s) p(-s)\). Note that \(q(s) = q(-s)\).
2. Determine the right-hand side of Eq. (6.166), and factor the resulting polynomial. This polynomial is even, both \(\lambda\) and \(-\lambda\) are its roots.
3. Select those poles \(\lambda_i\) of the polynomial having negative real parts, and construct \(\prod_i (s - \lambda_i)\), a monic polynomial with roots \(\lambda_i, i = 1, \ldots, n\).
4. Since \(\phi(s)\) is monic and its roots must have negative real parts, it follows that \(\phi(s)\) is uniquely determined by \(\phi(s) = \prod_{i=1}^{n} (s - \lambda_i)\). Note that the constraints on \(\phi(s)\) guarantee that no \(\lambda_i\) in step 3 will have zero real part.
5. Construct \(F\) such that \(1 - F^T(sI - A)^{-1}B = \phi(s)/p(s)\). This step constitutes the solution of a set of linear equations.

While, the pole assignment problem would allow the control designer to place the system's closed-loop poles at arbitrary locations, the situation in optimal control design is not unsimilar. In the latter case, the location of the optimal closed-loop poles will, of course, depend on the weighing matrix \(R\) in the cost function. It has
been shown by Kalman (1964) that the asymptotic behavior of the poles of optimal closed-loop transfer function \( G^*(s) = C(sI - A - BR^{-1}B^TK)^{-1}B \) can be described as follows: As matrix \( R \) approaches a null matrix, the poles of the optimal closed-loop system approach a stable Butterworth configuration.\(^1\)

**Example 6.11**

Perform Algorithm 6.1 for the system

\[
(A, B, Q) = \begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 & 0 \\ 0 & 0 \end{pmatrix}
\]

**Solution** We have \( p(s) = \text{det}(sI - A) = s^2 + 2, \) and \( q(s) = 5, \) an even function as required. Then, \( \phi(-s) \phi(s) = (s^2 + 2)(s^2 + 2) + 5 = s^4 + 4s^2 + 9. \) Now, we perform a factorization \( s^4 + 4s^2 + 9 = (s^2 - \sqrt{2}s + 3)(s^2 + \sqrt{2}s + 3), \) and from this it follows that \( \phi(s) = s^2 + \sqrt{2}s + 3. \) Then, \( 1 - F^T(sI - A)^{-1}B = (s^2 + \sqrt{2}s + 3)(s^2 + 2) = 1 + (\sqrt{2}s + 1)/(s^2 + 2). \) It therefore follows that

\[
(sI - A)^{-1}B = \frac{1}{s^2 + 2} \begin{pmatrix} 1 \\ s \end{pmatrix}
\]

and then, \( F^T = [-1 \quad -\sqrt{2}]. \)

It should be emphasized that this procedure is only good for time-invariant systems. It can be extended to time-varying systems, but this constitutes additional difficulties and is not considered here for lack of space.

**PROBLEMS**

6.1 Consider a bilinear scalar system,

\[
\dot{x} = x + ux + u
\]

with a cost function

\[
J = \frac{1}{2} \int_0^T (x^2 + u^2) \, df
\]

Develop a TPBV problem.

6.2 For a linear time-invariant system,

\[
\dot{x} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u
\]

with a cost function

\[
J = \frac{1}{2} \int_0^\infty (x^TQx + u^2) \, dt
\]

\(^1\) For a discussion of Butterworth filters, see for example, Weinberg (1962).
with $Q = \text{diag}(1, 0)$. Determine (a) the Riccati matrix (b) the optimal control and state equations.

6.3 CAD problem. Repeat Prob. 6.2 using your favorite package (MATRXx, CTRL_C, and so on.) or CONTROL.lab’s AMRE or RKS and REGT.

6.4 Consider an SISO system

$$\dot{x} = x + u, \ y = 2x, \ x(0) = 1$$

and

$$J = \frac{1}{2} \int_0^\infty (y^2 + u^2) \, dt$$

Determine the optimal control $u^*$ which minimizes $J$. Find the optimal $J^*$.

6.5 Derive the Riccati formulation solution of the output regulator problem that is, using an integrand $L = 1/2(y^T Q y + u^T R u)$ in Sec. 6.2.2 leading to optimal control and state equations.

6.6 Show that the matrix $[\phi_{22}(t_f; t) - F\phi_{12}(t_f; t)]$ in the derivation of the Riccati formulation in Sec. 6.2 is nonsingular.

6.7 CAD problem. Use programs RICRKUT and TVSTRG of TIMDOM/PC or your favorite software to solve the following linear time-varying optimal state regulator problem

$$\dot{x} = \begin{pmatrix} e^{-n^2} & 0 \\ 0 & -e^{-n^2} \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ x(0) = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$y = \begin{pmatrix} \sin \frac{t}{2} \\ 1 \\ \cos \frac{t}{2} \end{pmatrix} x$$

for $t_f = 5.0, \ Q = 2 \ I_2$ and $R = I_2$.

6.8 CAD problem. Use REGTO of CONTROL.lab or a similar software program, to solve the following output regulator problem

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \ x(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$J = \frac{1}{2} \int_0^\infty (y_1^2 + 2y_2^2 + u^2) \, dt$$

6.9 Consider the system,

$$\dot{x} = x + u, \ x(0) = 1$$

$$y = x, \ z = 1$$

Find an optimal control which satisfies this system while minimizing,

$$J = \frac{1}{2} \int_0^1 [(y - z)^2 + u^2] \, dt$$

6.10 Consider a nonlinear system $\dot{x} = g(x) + u f(x)$ with performance index
\[ J[x(t), u(\cdot), t] = \int_t^\eta [u^2 + f(x)] \, dt \]

Show that the Hamilton-Jacobi equation is linear in \( \partial J^*/\partial t \) and quadratic in \( \partial J^*/\partial x \).

**6.11** Consider the following quadratic function,

\[ P(u, x, p) = u^T Q u + 2x^T R u + 2u^T S p \]

where \( u \) is \( m \times 1 \), \( x \) and \( p \) are \( n \times 1 \) vectors and matrices \( Q, R, \) and \( S \) have appropriate dimensions. Show that \( P \) has a unique minimum in \( u \) for all \( x \) and \( p \) if, and only if \( 1/2 (Q + Q^T) \) is positive definite.

**6.12** Show that the optimal value of the objective function for a linear regulator problem is given by Eq. (6.92).
7 Large-Scale Systems Design

7.1 INTRODUCTION

Systems complexity in many real-life plants and processes has led to a new class of system theory for the past two decades, called, large-scale systems. A great number of today’s problems are brought about by present-day technology, societal, and environmental processes which are highly complex, “large” in dimension, stochastic, and uncertain by nature.

Two of the more accepted definitions of large-scale systems are (1) a system is large scale if it can be decomposed or decoupled into a finite number of subsystems, (2) a system is large scale if the concept of centrality does not hold, that is, a system in which all its components are not present within the same surroundings. An example of the first definition is a multiregion water resources system, while an example of the second is a multistation power system. These definitions, as will be seen later, play major roles in both modeling and control of large-scale systems.

In the next section, one of the most common methods of modeling of large-scale systems—aggregation will be introduced. Control of large-scale systems by hierarchy, known as hierarchical control, and its associated optimal algorithm will be given in Sec. 7.3. In Sec. 7.4, design of large-scale systems based on decentralized output information, known as decentralized control will be discussed. The field of large-scale systems is quickly getting vast and the main objective of this chapter is to give a brief understanding of this class of systems.
7.2 AGGREGATED MODELS OF LARGE-SCALE SYSTEMS

In Chap. 2, dynamic models of multivariable control systems were introduced and several definitions and notions such as state variables, state transformations, solutions, and so forth were presented. Because of high dimensions of large-scale systems, it has been a common practice to reduce the order of such systems without sacrificing key properties such as stability (Sec. 3.11), and controllability and observability (Secs. 3.3 and 3.4). There are two primary established techniques for reducing large-scale systems models. These are “aggregation,” and “perturbation” schemes. An aggregated model of a system is described by a “coarser” set of variables. The main reason for aggregating a system model is to deal with lower order systems, while retaining the key qualitative properties of the system, such as stability. This approach is viewed as a natural process through the second method of Lyapunov. In other words, the stability of a system described by several state variables is entirely assessed by a single variable—the Lyapunov function. Figure 7.1 shows a pictorial presentation of the aggregation process. The relation \( z = Rx \) as will be seen later, is the aggregation equality. Matrix \( R \) is called the aggregation matrix. The system on the left is described by four variables (circles) and the system on the right represents an aggregated model for it, where two variables now describe the system. The first aggregated variable, called \( z_1 \), can be considered as an average of the full model’s first two variables, while the second aggregated variable is an average of the third and fourth variables.

The other scheme for large-scale systems modeling, perturbation, is based on ignoring certain interactions of the dynamic or structural nature in a system. Here again, the benefits received from reduced computations must not be at the expense

\[
\begin{align*}
\text{AGGREGATION} \\
\begin{array}{c}
\text{x} \\
\text{1} \\
\text{2} \\
\text{3} \\
\text{4}
\end{array} \\
\rightarrow \\
\begin{array}{c}
\text{z} \\
\text{1} \\
\text{2}
\end{array}
\end{align*}
\]

\[
z = Rx = \begin{bmatrix}
1/2 & 1/2 & 0 & 0 \\
0 & 0 & 1/2 & 1/2
\end{bmatrix}
\]

Figure 7.1 A pictorial explanation of the aggregation process.
of sacrificing key system properties. Figure 7.2 presents a pictorial presentation of perturbation of the singular type. There are two main schemes in system perturbation—regular and singular. In regular perturbation, the system is decoupled as a result of weak couplings (interactions) among its potential subsystems, hence the order of the overall system does not actually reduce. In singular perturbation, on the other hand, the system undergoes a reduction of dimension by ignoring the dynamics of fast variables and the slow variables play the dominant role in system's dynamic behavior.

Because of the primary theme of this text—design, we have chosen to present aggregation only and leave the subject of perturbation for the reader to consult other sources (Jamshidi, 1983), see also Problem 7.10. The aggregation seems to be a more practical method for the general design problem of large-scale systems. This point will be illustrated in terms of near-optimum control design of a large-scale system later (see CAD Ex. 7.5).

In sequel, three basic methods of reducing models of multivariable large-scale systems will be presented.

### 7.2.1 General Aggregation

Aggregation has long been a technique for analyzing static economic models. The treatment of aggregation stems from vector space transformations as shown in Fig. 7.3. In this diagram, $X$, $Y$, $Z$, and $V$ are topological (or vector) spaces, $f$ represents
a linear continuous map between the exogeneous variable \( x \in X \) and endogenous variables \( y \in Y \). The aggregation procedures \( h: X \rightarrow Z \), \( k: Z \rightarrow V \), and \( g: Y \rightarrow V \), lead to aggregated variables \( z \in Z \) and \( v \in V \). The aggregation is said to be "perfect" when \( k \) is chosen such that the relation

\[
gf(x) = kh(x)
\]

holds for all \( x \in X \). The notion of perfect aggregation is an idealization at best, and in practice it is approximated through two alternative procedures according to econometricians. These are (a) to impose some restrictions on \( f, g, \) and \( h \) while leaving \( X \) unrestricted and (b) to require Eq. (7.1) to hold on \( X_0 \) some subset of \( X \).

Consider a linear controllable system (full model),

\[
\dot{x}(t) = Ax(t) + Bu(t), \quad x(t_0) = x_0
\]

\[
y(t) = Cx(t)
\]
where \( x(t) \), \( u(t) \), and \( y(t) \) and \( n \times 1 \), \( m \times 1 \), and \( r \times 1 \) state, control, and output vectors, respectively, \( A \), \( B \), and \( C \) are \( n \times n \), \( m \times n \), and \( r \times n \) matrices. Assume that \( n \) is large (\( n > 30 \)). Moreover, consider an aggregated (reduced model) linear time-invariant model of the same system,

\[
\dot{z}(t) = Fz(t) + Gu(t), \quad z(0) = z_0 \quad (7.4)
\]

\[
\dot{\hat{y}}(t) = Dz(t) \quad (7.5)
\]

where \( z(t) \) is the \( k \)-dimensional aggregated state vector, \( F \) and \( G \) are \( k \times k \) and \( k \times m \) dimensional constant matrices, respectively. The vector \( \hat{y}(t) \) is an \( r \times 1 \) approximate output. Assume that \( x \) is observed directly and Eq. (7.2) is, as indicated before, controllable. A constant, \( k \times n \) matrix \( R \) of rank \( k \) relating \( x \) and \( z \) vectors by,

\[
z(t) = Rx(t), \quad z_0 = Rx_0 \quad (7.6)
\]

is called the aggregation matrix. A set of conditions, called dynamic exactness or "aggregability," for Eqs. (7.4) and (7.5) to be an aggregation for model of Eqs. (7.2) and (7.3) can be easily seen to be

\[
FR = RA \quad (7.7)
\]

\[
G = RB \quad (7.8)
\]

\[
DR \equiv C \quad (7.9)
\]

Making use of the generalized (pseudo-) inverse of \( R \), Eq. (7.7), can be reduced to

\[
RA = RAR^+R = RAR^T(RR^T)^{-1}R = FR \quad (7.10)
\]

where \( R^+ \overset{\Delta}{=} R^T(RR^T)^{-1} \) is the generalized inverse of \( R \). A comparison of both sides of the latter part of Eq. (7.10) reveals that,

\[
F = RAR^T(RR^T)^{-1} \quad (7.11)
\]

which indicates that once the aggregation matrix \( R \) is known, the aggregated matrix \( F \) is obtained by Eq. (7.11) and the aggregated control matrix \( G \) is determined from Eq. (7.8). The analysis of these three identities gives some insight in the choice of the aggregation matrix \( R \).

It is noted that aggregated matrix \( F \) is obtained from Eq. (7.11) only if the conditions Eqs. (7.7) to (7.9) are satisfied. Under these circumstances, the eigenvalues of \( F \) constitute a subset of the eigenvalues of \( A \). This is, in fact, what we term as modal aggregation (Sec. 7.2.2) and is considered as a special (full aggregable) case of this model simplification.

It is emphasized that the use of Eq. (7.11) is an approximation which in fact minimizes the square of the norm \( ||FR - RA|| \) unless the consistency relation \( RAR^+R = RA \) is satisfied. If an error vector is defined as \( e(t) = z(t) - x(t) \), then its dynamic behavior is given by \( \dot{e}(t) = Fe(t) + (FR - RA)x(t) + (G - RB)u(t) \), which reduces to \( \dot{e}(t) = Fe(t) \) if conditions Eqs. (7.7) and (7.8) hold. Hence, if \( e(0) = 0 \), then \( e(t) \)
= 0 for all \( t \geq 0 \). Should \( e(0) \neq 0 \) but \( F \) be a stable matrix, then \( \lim_{t \to \infty} e(t) = 0 \); that is, dynamic exactness condition Eqs. (7.7) and (7.8) is asymptotically satisfied.

**Example 7.1**

For the third order unaggregated system described by

\[
\dot{x} = \begin{bmatrix} -0.1 & 1 & 2 \\ 0 & -1.2 & 1 \\ 0 & 0 & -3 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u
\]

find an aggregated system.

**Solution** The first solution is obtained by the use of eigenvalues of \( A \) matrix in Eq. (7.12) which are \( \lambda_1 = -0.1, \lambda_2 = -1.2 \) and \( \lambda_3 = -3 \). From the relative magnitudes of \( \lambda \), it is clear that the slowest modes of the system are the first two, hence a structure for aggregation matrix \( R \) can be

\[
R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 2 & 2 \end{bmatrix}
\]

which implies that the two potential aggregated states \( z_1(t) \) and \( z_2(t) \) are chosen to be the slowest mode of the original system and an average of the next two modes of the full model, respectively. The aggregated matrices \( F \) and \( G \) are obtained from Eqs. (7.11) and (7.8), that is,

\[
\dot{z}(t) = Fz(t) + Gu(t) = \begin{bmatrix} -0.1 & 3 \\ 0 & -1.5 \end{bmatrix} z(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u(t)
\]

It is noted that for this aggregation, the aggregability measure is given by

\[
FR - RA = \begin{bmatrix} 0 & 1 & -1 \\ 2 & 2 \\ 0 & -1 & 1 & -1/4 \end{bmatrix} \neq 0
\]

which indicates that aggregability condition Eq. (7.7) is not satisfied. This would mean that this choice of \( R \) needs to be changed. One can, through trial and error, come with an aggregation matrix such that some desired goal would be satisfied. Among desirable goals are close agreements between time (or frequency) responses of the full- and reduced-order models, or full aggregability.

**CAD Example 7.1**

In this example, a sixth-order system representing a modified form of a flexible booster (Jamshidi, 1983) is reduced to a second- and a fourth-order system; step responses for the full-order and reduced-order models are obtained for comparison purposes. In this example we use PC-MATLAB (see Appendix B).
>> a = [-.42 -2 -.008 0.95 1d-5; 1 -.053 -3d-4 10d-4 3.5d-4 0 0 0 1 ... 0 0 0 -688 -5.9 0 0 0 0 0 1; 0 0 0 -4880 -18.5];
>> b=[-9.5; -5.2d-2; 2;10d-2;2;899;10d-3; -488.5]
>> c=[1 0 0 8.85d-4 0 -9.82d-3;0 1 0 0 0 0];
>> r1=[1 0 0 0 0 0;0 1 0 0 0 0]; f1=r1*a*pinv(r1) %pinv is pseudo-inverse command

f1 =
-0.4200  -0.2000
     1.0000  -0.0530

>> g1=r1*b, h1=c*pinv(r1)
g1 =
-9.5000
-0.0520

h1 =
1
0
0
1

>> d=[0;0]; time = [0:0.1:20.]; % define D matrix and time vector for
% simulation

% Now, a 4th order reduced model
>> r2 = [eye (4) , 0 * ones (4,2)];
>> f2=r2*a*pinv(r2), g2=r2*b, h2=c*pinv(r2)

f2 =
-0.4200  -0.2000  -0.0080   0
     1.0000  -0.0530  -0.0003  0.0010
     0 0 0 1.0000
     0 0 -688.0000  -5.9000

g2 =
-9.5000
-0.0520
0.1000
899.0000

h2 =
1.0000  0 0 0.0009
0 1.0000  0 0

>> % Impulse responses for full and reduced-order models
>> uin = 0 * time; uin (1,1) = 1;
>> yf = lsim (a,b,c,d,uin, time);
>> y1 = lsim (f1,g1,h1,d,uin, time);
>> y2 = lsim (f2,g2,h2,d,uin, time);
>> Plot (time, yf (:1), time, y1 (:1))
>> Plot (time, yf (:2), time, y2 (:2))

The resulting impulse responses for the full-order system and typical output responses of the second- and fourth-order reduced models are shown in Fig. 7.4. Note that the
Figure 7.4 Impulse responses for full-order and three reduced-order models for CAD Example 7.1.
(a) Second-order reduced model—output $y_1$.
(b) Second-order reduced model—output $y_2$.
(c) Fourth-order reduced model—output $y_1$.
(d) Fourth-order reduced model—output $y_2$. 
Simulation of Full & 4th Order Reduced Model ... $y_1(t)$

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure_7_4_e}
\caption{(Continued)}
\end{figure}

Simulation of Full & 4th Order Reduced Model ... $y_2(t)$

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure_7_4_d}
\caption{(Continued)}
\end{figure}
two responses of both aggregated models are very close to full model’s responses in spite of the fact that both cases were not exact, that is, FR ≠ RA. These plots represent one possible criterion for model reduction, i.e., reduce the model such that the two models’ time responses match.

**CAD Example 7.2**

In this example, a seventh order system, representing a single-machine infinite bus system is considered. Three different aggregated models are obtained and their step responses are compared. This example brings up the point that the exact order of the reduced model is very critical and one can not always expect satisfactory results as those of CAD Ex. 7.1.

```matlab
>> a = [−.58 0 0 −.269 0 .2 0;0 −1 0 0 0 1 0;... 
    0 0 −5 2.12 0 0 0;0 0 0 377 0 0;−.141 0 .141 −.2 −.28 0 0; 
    0 0 0 0 .0838 2;−173 66.7 −116 40.9 0 −66.7 −16.7]; 
```

```matlab
>> b = [1;0;1;0;1;0;1]; c = [1 −1 1 0 1 0]; d = 0; 
```

```matlab
>> eig (a) 
ans =
 −8.5478 + 8.2044i 
 −8.5478 − 8.2044i 
 −0.8586 + 8.3793i 
 −0.8586 + 8.3793i 
 −0.3623 + 0.5564i 
 −0.3623 − 0.5564i 
 −3.9388 
```

```matlab
>> % First order reduced model 
>> r1 = [eye(1),0*ones(1,6)]; 
>> f1 = r1*a*pinv(r1),g1 = r1*b,h1 = c*pinv(r1) 
```

```matlab
f1 =
 −0.5800 
```

```matlab
>> eig(f1) 
ans =
 −0.5800 
```

```matlab
>> % Second order reduced model 
>> r2 = [eye(3),0*ones(3,4)]; 
>> f2 = r2*a*pinv(r2),g2 = r2*b,h2 = c*pinv(r2); 
```

```matlab
f2 =
 −0.5800 0 0 
 0 −1.0000 0 
 0 0 −5.00000 
```

```matlab
>> eig(f2) 
```
ans =
-0.5800
-1.0000
-5.0000

>> % Fifth order reduced model
>> r5=[eye(5),0*ones(5,2)];
>> f5=r5*a*pinv(r5),g5=r5*b;h5=c*pinv(r5);

f5 =
-0.5800 0 0 -0.2690 0
0 -1.0000 0 0 0
0 0 -5.0000 2.1200 0
0 0 0 0 377.0000
-0.1410 0 0.1410 -0.2000 -0.2800

>> eig(f5)

ans =
-0.8839 + 8.40851
-0.8839 + 8.40851
-3.7903
-0.3019
-1.0000

Note that in the first-order aggregation, all seven eigenvalues of the system (three complex conjugate pairs and a real one) have been approximated by only one real eigenvalue at -0.58. For a third-order aggregation however, the system’s eigenvalues are approximated by three real ones at -0.58, -1.0, and -5.00. One should not expect good approximation of the full-order models, because none of the dominant complex conjugate eigenvalues, that is, those close to the jw-axis, have been retained. The fifth-order aggregated model, on the other hand, has been approximated by three real and a pair of complex conjugate eigenvalues. Following is the step responses calculations and a plot of the output of the full-order model and three reduced-order models in Fig. 7.5.

>> time=[0:.1:10];
>> yf=step(a,b,c,d,1,time);
>> y1=step(f1,g1,h1,d,1,time);
>> y3=step(f3,g3,h3,d,1,time);
>> y5=step(f5,g5,h5,d,1,time);
>> plot(time,yf,time,y1,time,y3,time,y5)
>> xlabel('Time'),ylabel('yf & y1,3,5')
>> title('Step Responses Full and Three Reduced-Order Models')

The time response verifies the points brought up earlier regarding the eigenvalues of the full-order and reduced-order models. These plots make it clear that as the number of retained modes of the system increases, the step response of reduced model gets
closer to that of the full model. On the other hand, with only a few modes (1 or 2) retained there are great discrepancies between the two models.

7.2.2 Modal Aggregation

Next, we concentrate on a special case of the general aggregation considered in Sec. 7.2.1. The method is based on the early works of Davison (1968) and Chidambaram (1969). It is essentially a mapping based on a submatrix of the full-order model's modal matrix. The columns of this matrix are its eigenvectors, and can be ordered in accordance with their dominancy, that is, eigenvector with eigenvalues closest to the $j\omega$-axis.

Consider the linear time-invariant system Eqs. (7.2) and (7.3) repeated here for convenience,

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(t_0) = x_0 \quad (7.13)$$

$$y(t) = C \, x(t) \quad (7.14)$$

Let the large-scale linear system Eq. (7.13) be written as
\begin{align}
\dot{x} &= \begin{bmatrix} \dot{z}(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} A_1 & A_{12} \\ A_{21} & A_2 \end{bmatrix} \begin{bmatrix} z(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u(t) \\
y(t) &= [C_1 \ C_2] \begin{bmatrix} z(t) \\ x_2(t) \end{bmatrix}
\end{align}

where \( z \) is the \( k \times 1 \) aggregated state and \( x_2(t) \) is \((n - k)\)th order residual state.

System Eqs. (7.15) and (7.16) can be transformed to its modal form,

\begin{align}
\dot{\hat{w}} &= \begin{bmatrix} J_1 & 0 \\ 0 & J_2 \end{bmatrix} \begin{bmatrix} \hat{w} \\ \hat{v}_2 \end{bmatrix} + \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} u \\
y &= [D_1 \ D_2] \begin{bmatrix} \hat{w} \\ \hat{v}_2 \end{bmatrix}
\end{align}

where \( \hat{w} \) is the vector of retained dominant variables,

\( x = M \nu = M[\nu_1; \nu_2]^T, J = \text{Block-diag}(J_1, J_2) \)

\( = M^{-1} A M, \Gamma = [\Gamma_1; \Gamma_2]^T = M^{-1} B \)

\( D = [D_1 \ D_2] = CM \)

and \( M \) is the ordered modal matrix, given by

\begin{align}
M &= [u_1^d; u_2^d; \cdots; u_n^d] = \begin{bmatrix} M_1 & M_{12} \\ M_{21} & M_2 \end{bmatrix}
\end{align}

where \( u_i^d, i = 1, \ldots, n \) are the dominant set of eigenvectors (or real and imaginary parts of a complex conjugate eigenvector or generalized eigenvector) and the partition of \( M \) in Eq. (7.19) is identical to that of \( A \) in Eq. (7.15). Assume that it is desired to retain \( k (k < n) \) dominant modes (vector \( w \)) of Eq. (7.17), that is,

\begin{align}
\dot{\hat{w}} &= P J P^T \hat{w} + P \Gamma u \\
\end{align}

where

\begin{align}
P &= [I_k \ 0_{k \times (n-k)}] \\
\end{align}

and \( w = P \nu \). Let us take the Laplace transform of the lower half of Eq. (7.17) to yield

\begin{align}
V_2(s) &= (sI - J_2)^{-1} \Gamma_2 U(s)
\end{align}

If only dc transmission between \( u(t) \) and \( v_2(t) \) is of interest, and since \( J_2 \) represents nondominant (fast) modes, Eq. (7.22) can be approximated by

\begin{align}
v_2(t) &= -J_2^{-1} \Gamma_2 u(t) \overset{\Delta}{=} L u(t)
\end{align}
The partitioned forms of $x$ and $v$ lead to

\[
\begin{bmatrix}
  z \\
  x_2
\end{bmatrix}
= \begin{bmatrix}
  M_1 & M_{12} \\
  M_{21} & M_2
\end{bmatrix}
\begin{bmatrix}
  w \\
  v_2
\end{bmatrix}
\]

(7.24)

\[z = M_1 w + M_{12} v_2, \quad x_2 = M_{21} w + M_2 v_2\]

(7.25)

Now, assuming that $M_1$ is nonsingular and solving for $w$ in the first equation of Eq. (7.25) and substituting it in the second while using Eq. (7.23) leads to

\[x_2 = M_{21} M_1^{-1} z + \left( M_2 + M_{21} M_1^{-1} M_{12} \right) L u \triangleq N z + E u\]

(7.26)

Eliminating $x_2$ in Eqs. (7.15) and (7.16) using Eq. (7.26) leads to the aggregated model

\[
\dot{z} = F z + G u
\]

(7.27)

\[
y = H z + K u
\]

(7.28)

where

\[
F = A_1 + A_{12} N
\]

(7.29a)

\[
G = B_1 + A_{12} E
\]

(7.29b)

\[
H = D_1 M_1^{-1}
\]

(7.29c)

\[
K = (D_1 M_1^{-1} M_{12} + D_2) L
\]

(7.29d)

In this method, the effects of the nondominant modes have been neglected to result in the reduced model.

In the case when the steady-state response is not of great importance, one can use the original modal reduction scheme suggested by Davison (1966, 1968). In an abbreviated fashion, the aggregated system matrices are given by,

\[
F = M_1 P J P^T M_1^{-1}
\]

(7.30)

\[
G = M_1 P M^{-1} B
\]

(7.31)

\[
R = M_1 P M^{-1}
\]

(7.32)

\[
H \equiv C R^+
\]

(7.33)

where $R$ is the aggregation matrix and $R^+$ is the pseudo inverse of $R$.

**Example 7.2**

Consider the following seventh-order system
Sec. 7.2 Aggregated Models of Large-Scale Systems

\[
\dot{x} = \begin{bmatrix}
-0.58 & 0 & 0 & -0.269 & 0 & 0.2 & 0 \\
0 & -1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & -5 & 2.12 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 377 & 0 & 0 \\
-0.141 & 0 & 0.141 & -0.2 & -0.28 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.0838 & 2 \\
-173 & 66.7 & -116 & 40.9 & 0 & -66.7 & -16.7 \\
\end{bmatrix} x + \begin{bmatrix}
1 \\
0 \\
1 \\
0 \\
1 \\
0 \\
1 \\
\end{bmatrix} u
\]

\[
y = [1 \quad -1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0] x
\]

whose unforced form represents a single-machine infinite bus power system (Jamshidi, 1983). It is desired to find some modally aggregated models.

**Solution** The eigenvalues of the system matrix are \(-0.362 \pm j0.556\), \(-0.858 \pm j8.38\), \(-3.94\), and \(-8.55 \pm j8.2\) which indicates that the system has as few as 2 or as many as 4 or 5 dominant modes. The modal matrix, set in the order of dominance of the eigenvalues is given by

\[
M = \begin{bmatrix}
0.1692 & 0.2416j & 0.0080 & 0.0087j & 0.0400 & 0.0001 + 0.0029j \\
1.0000 & 0.0000j & -0.0282 & 0.0027j & 0.1692 & -0.0002 + 0.0151j \\
-0.0592 & 0.1256j & -0.0583 & 0.0942j & 0.2593 & -0.0002 + 0.0001j \\
-0.1624 & 0.2593j & 0.2584 & 0.4144j & 0.1298 & 0.0008 + 0.0006j \\
-0.0002 & 0.0005j & 0.0086 & 0.0067j & -0.0014 & -0.0000 + 0.0000j \\
0.6377 & 0.5564j & -0.0265 & 0.2357j & -0.4972 & -0.1217 - 0.1157j \\
-0.2970 & 0.0533j & 1.0000 & 0.0000j & 1.0000 & 1.0000 - 0.0000j \\
\end{bmatrix}
\]

and the block-diagonal Jordan matrix \(J\) is given by

\[
J = \begin{bmatrix}
-0.362 & 0.556 & 0 & 0 & 0 & 0 & 0 \\
-0.556 & -0.362 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -0.858 & 8.38 & 0 & 0 & 0 \\
0 & 0 & -8.38 & -0.858 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -3.94 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -8.55 & 8.2 \\
0 & 0 & 0 & 0 & 0 & -8.2 & -8.55 \\
\end{bmatrix}
\]

Then, three of the aggregated models using the modal approach, that is, Eqs. (7.29a) to (7.29d) will be,

\[k = 2\]

\[
(F, G, H, K) = \begin{bmatrix}
-0.752 & 0.200 \\
-2.303 & 0.027 \\
\end{bmatrix}, \begin{bmatrix}
-0.76 \\
-0.112 \\
\end{bmatrix}, \begin{bmatrix}
-2.90 & 0.075 \\
\end{bmatrix}, \begin{bmatrix}
-1.04 \\
\end{bmatrix}
\]

\[k = 4\]

\[
(F, G, H, K) = \begin{bmatrix}
-.86 & 0.2 & 0.22 & -0.21 \\
-1.41 & -0.01 & 1.1 & 0.29 \\
0 & 0 & -5.0 & 2.1 \\
-18.18 & 0.973 & -43.5 & 3.4 \\
\end{bmatrix}, \begin{bmatrix}
1.4 \\
2.1 \\
1 \\
-124 \\
\end{bmatrix}, \begin{bmatrix}
-0.41 & -0.01 & 2.1 & 1.3 \\
\end{bmatrix}, \begin{bmatrix}
7.22 \\
\end{bmatrix}
\]
\[ k = 5 \]

\[
(F, G, H, K) = \begin{pmatrix}
-1.1 & 0.21 & -0.52 & -0.15 & -6.45 \\
-3.0 & 0.07 & -2.6 & 0.59 & -3.22 \\
0 & 0 & -5.0 & 2.12 & 0 \\
0 & 0 & 0 & 377 & 0 \\
-0.14 & 0 & 0.14 & -0.20 & -0.28 \\
\end{pmatrix} \begin{pmatrix}
1.55 \\
2.8 \\
1 \\
0 \\
1 \\
\end{pmatrix},
\]

\[
\begin{pmatrix}
-1.97 & 0.07 & -1.62 & 1.6 & -32.24
\end{pmatrix}, [0]
\]

**CAD Example 7.3**

In this CAD example, the single-machine system of Ex. 7.2 will be used with PC-MATLAB to illustrate both modal aggregation cases.

\[
>> \text{% modal aggregation No. 1} \\
>> a = [-0.58 0 0 -0.269 0.2; 0 -1 0 0 0 1 0; 0 0 0 0 5 2.12 0 0 0; 0 0 0 0 377 0 0; \ldots \\
-0.141 0.141 -2 -0.28 0 0; 0 0 0 0 0.0838 2; -173 66.7, -116 40.9 0 -66.7 -16.7]; \\
>> b = [1 0 1 0 1 0 1 0]'; c = [1 -1 1 1 0 1 0]; \\
>> \text{% modal + Jordan matrices} \\
>> [m, j] = \text{eig}(a); \\
>> \text{% eigenvalues are:} \\
>> \text{diag}(j)
\]

\[
\text{ans} = \\
-8.5478 + 8.2044i \\
-8.5478 - 8.2044i \\
-0.8586 + 8.3793i \\
-0.8586 - 8.3793i \\
-0.3623 + 0.5564i \\
-0.3623 - 0.5564i \\
-3.9388 + 0.0000i
\]

\[
\text{>> % order eigenvectors in modal matrix} \\
\text{>> m_ord = [m(:,5),m(:,6),m(:,3),m(:,4),m(:,7), m(:,1),m(:,2)];} \\
\text{>> %extract 2nd order modal matrix} \\
\text{>> m1 = m_ord (1:2, 1:2); m2 = m_ord (1:2, 3:7); m21 = m_ord (3:7,1:2);}
\]

\[
\text{>> m2 = m_ord(3:7, 3:7);} \\
\text{m1 = Columns 1 through 2} \\
0.1692 - 0.2416i \\
0.1692 + 0.2416i \\
1.0000 + 0.0000i \\
1.0000 - 0.0000i
\]
Sec. 7.2 Aggregated Models of Large-Scale Systems

```matlab
>> p1 = [eye(2),0*ones(2,5)]; gama = inv(m_ord)*b;
>> gama 2 = gama (3:7); d = c * m_ord; js = j(3:7,3:7);
>> L = - inv(j2) * gama2; n = m21*inv(m1); e = (m2 + m21 * inv(m1) * m12) * 1;
   al = a(1,2,1:2);
>> a12 = a(1:2,3:7); d1 = d (1:2); b1 = b(1:2);
>> % Second-order aggregated model
>> f2 = a1 + a12 * n, g2 = b1 + a12 * e, h2 = d1 * inv(m),
   k2 = d1 * inv(m1) * m12 * L

f2 =
-0.7520 + 0.0000i  0.2003 - 0.0000i
-2.3036 + 0.0000i  0.0275 - 0.0000i

g2 =
-0.7508 - 0.0000i
-0.1125 - 0.0000i

h2 =
-2.897i + 0.0000i  0.0754 - 0.0000i

k2 =
-1.0407 - 0.0000i

:

>> % Fourth-order aggregated model

f4 =
-0.8630 + 0.0000i  0.1980 - 0.0000i  0.2193 + 0.0000i  0.2097 - 0.0000i
-1.4151 + 0.0000i  -0.0098 - 0.0000i  1.0967 + 0.0000i  0.2964 - 0.0000i
  0          0      0.0000  2.1200
-18.1811 - 0.0000i  0.9728 - 0.0000i  -43.5067 - 0.0000i  3.4310 + 0.0000i

g4 =
1.0E+02 *
0.0142 + 0.0000i
0.0211 + 0.0000i
0.0100
-1.2399 - 0.0000i

h4 =
-0.4151 + 0.0000i  -0.0098 - 0.0000i  2.0967 + 0.0000i  1.2964 - 0.0000i

k4 =
7.2197 + 0.0000i
Figure 7.6   Impulse responses for CAD Example 7.3 using first method of modal aggregation.

```matlab
>> % impulse response comparison
>> time = [0:0.1: 10];
>> imf = impulse(a,b,c,d,1,time);
>> ial = impulse(f,g,h,k,1, time);
...
>> Plot(time, imf, time, ial, time, . . ., ias)
>> xlabel('Time'), ylabel('yf & ya')
>> title('Impulse Response - Full & 3 Modally Reduced Models')
```

The impulse responses of the full-order and three modally-reduced models using the relations Eq. (7.29) are shown in Fig. 7.6.

In order to implement the second modal aggregation, given by Eqs. (7.30) to (7.33), a *.m file in PC-MATLAB was written which is created using any screen editor and saved on the same directory as PC-MATLAB itself. The file mod.m is shown.

```matlab
function [f,g,h,r] = mod(a,b,c,d,m,k)
% Modal aggregation Case 2 (Eqs. 7.30 - 7.33)
% M is modal matrix whose columns are put in order of dominancy
% r is lxn aggregation matrix
[n,mx]=size(b);
m1=m(1:k,1:k);
```
Sec. 7.2  Aggregated Models of Large-Scale Systems

\begin{verbatim}
jord = inv(m)*a*m; p = [eye(k),0*ones(k,n-k)]; f = m1*p*jord*p*inv(m1); g = m1*p*inv(m)*b; r = m1*p*inv(m); h = c*pinv(r);
end

Once mod.m has been created and debugged within PC-MATLAB, the following
statements would provide three reduced-order models.

\begin{verbatim}
>> [f2,g2,h2,r2] = mod(a,b,c,df,m_ord,2)

f2 =
      -0.7520 - 0.0000i  0.2003 + 0.0000i
      -2.3036 - 0.0000i  0.0274 + 0.0000i

g2 =
      -1.1999 - 0.0000i
      -2.1071 + 0.0000i

h2 =
      1.0358 - 0.0000i  -1.2322 - 0.0000i

r2 =
  Columns 1 through 4
      0.9956 + 0.0000i  -0.0200 - 0.1197 + 0.0000i  0.0017 - 0.0000i
      0.0630 + 0.0000i  0.8940 - 0.4375 + 0.0000i  0.0113 - 0.0000i

  Columns 5 through 7
      -2.0786 - 0.0000i  0.0223 - 0.0000i  0.0026 + 0.0000i
      -1.7471 + 0.0000i  0.1179 + 0.0000i  0.0145 + 0.0000i

>> imz2 = impulse (f2,g2,h2,r2,1,t);
>> [f4,g4,h4,r4] = mod(a,b,c,df,m_ord,4)

f4 =
      -0.8630 + 0.0000i  0.1980 - 0.0000i  0.2193 + 0.0000i  -0.2097 - 0.0000i
      -1.4151 + 0.0000i  -0.0098 - 0.0000i  1.0967 + 0.0000i  0.2964 - 0.0000i
      0.0000 - 0.0000i  -0.0000 + 0.0000i  -5.0000 - 0.0000i
      -18.1811 - 0.0000i  0.9728 - 0.0000i  -43.5067 - 0.0000i  3.4310 + 0.0000i

g4 =
      -0.0633 - 0.0000i
      -4.0446 + 0.0000i
      -9.6787 + 0.0000i
      -5.3362 + 0.0000i

h4 =
      1.1791 + 0.0000i  -1.3725 - 0.0000i  0.1763 + 0.0000i  0.5019 - 0.0000i
\end{verbatim}
\( r4 = \)

Columns 1 through 4
\[
\begin{align*}
0.9934 + 0.0000i & -0.0198 + 0.0000i & -0.1107 + 0.0000i & -0.0032 - 0.0000i \\
0.0277 + 0.0000i & 0.8944 + 0.0000i & -0.4238 + 0.0000i & -0.0293 - 0.0000i \\
-0.4599 + 0.0000i & 0.0337 - 0.0000i & -0.0578 - 0.0000i & 0.0890 + 0.0000i \\
-0.2389 + 0.0000i & 0.0209 - 0.0000i & -0.5350 + 0.0000i & 1.0485 - 0.0000i \\
\end{align*}
\]

Columns 5 through 7
\[
\begin{align*}
-0.9487 - 0.0000i & 0.0226 - 0.0000i & 0.0027 + 0.0000i \\
-3.6628 + 0.0000i & 0.1168 + 0.0000i & 0.0143 + 0.0000i \\
-9.1596 + 0.0000i & -0.0102 - 0.0000i & -0.0014 - 0.0000i \\
-4.5601 + 0.0000i & -0.0093 - 0.0000i & -0.0021 - 0.0000i \\
\end{align*}
\]

\[>> \text{ imz4 = impulse (f4,g4,h4,r4,1,t); }\]
\[>> [f5,g5,h5,r5]=	ext{mod(a,b,c,df,m_ord,5)}\]

\( f5 = \)

\[
1.0E + 002 * \\
\]

Columns 1 through 4
\[
\begin{align*}
-0.0117 - 0.0000i & 0.0021 + 0.0000i & -0.0052 - 0.0000i & -0.0015 + 0.0000i \\
-0.0297 - 0.0000i & 0.0007 + 0.0000i & -0.0262 - 0.0000i & 0.0059 - 0.0000i \\
0.0000 + 0.0000i & -0.0000 - 0.0000i & -0.0500 + 0.0000i & 0.0212 + 0.0000i \\
-0.0000 - 0.0000i & -0.0000 - 0.0000i & 0.0000 + 0.0000i & -0.0000 - 0.0000i \\
-0.0014 - 0.0000i & 0.0000 - 0.0000i & 0.0014 - 0.0000i & -0.0020 - 0.0000i \\
\end{align*}
\]

Column 5
\[
-0.0645 - 0.0000i \\
-0.3224 - 0.0000i \\
0.0000 + 0.0000i \\
3.7700 + 0.0000i \\
0.0028 + 0.0000i \\
\]

\( g5 = \)

\[
1.5767 - 0.0000i \\
2.8918 + 0.0000i \\
0.9509 - 0.0000i \\
-0.0155 + 0.0000i \\
1.0054 + 0.0000i \\
\]

\( h5 = \)

Columns 1 through 4
\[
1.2694 - 0.0000i & -0.9522 + 0.0000i & 1.1939 - 0.0000i & 0.9288 - 0.0000i \\
\]

Column 5
\[
1.6373 + 0.0000i \\
\]

\( r5 = \)

Columns 1 through 4
\[
1.0639 + 0.0000i & -0.0248 - 0.0000i & 0.0520 + 0.0000i & -0.0169 - 0.0000i \\
0.3256 + 0.0000i & 0.8732 - 0.0000i & 0.2645 + 0.0000i & -0.0870 - 0.0000i \\
-0.0034 - 0.0000i & 0.0012 - 0.0000i & 0.9969 - 0.0000i & 0.0006 - 0.0000i \\
-0.0104 + 0.0000i & 0.0046 - 0.0000i & -0.0071 + 0.0000i & 1.0042 - 0.0000i \\
0.0007 - 0.0000i & -0.0003 + 0.0000i & 0.0005 + 0.0000i & -0.0002 + 0.0000i \\
\]
Figure 7.7  Impulse responses for CAD Example 7.3 using second method of modal aggregation.

Columns 5 through 7

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4579</td>
<td>0.0000i</td>
<td>0.0210</td>
<td>0.0000i</td>
<td>0.0029</td>
</tr>
<tr>
<td>2.2865</td>
<td>0.0000i</td>
<td>0.1227</td>
<td>0.0000i</td>
<td>0.0153</td>
</tr>
<tr>
<td>-0.0427</td>
<td>0.0000i</td>
<td>-0.0011</td>
<td>0.0000i</td>
<td>0.0001</td>
</tr>
<tr>
<td>0.0034</td>
<td>0.0000i</td>
<td>-0.0047</td>
<td>0.0000i</td>
<td>-0.0014</td>
</tr>
<tr>
<td>1.0042</td>
<td>0.0000i</td>
<td>0.0003</td>
<td>-0.0000i</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

```matlab
>> imz5 = impulse (f5,g5,h5,r5,t1,t);
>> impz = impulse(a,b,c,d,1,t);
>> plot (t,impz,t,imz2,t,imz4,t,imz5),xlabel ('Time),ylabel ('yf & ya')
>> title('Impulse Responses - Full & 3 Modally Reduced Models # 2')
```

The impulse responses of the full-order and three reduced models are shown in Fig. 7.7. Note that the f matrices are the same as before, but as expected, g and h matrices are not. Here again, the responses of reduced models and full model match better when there are sufficient number of modes retained. The best order of reduced model would be when a criterion such as minimum error response can be satisfied.
7.2.3 Balanced Aggregation

One of the main shortcomings of model reduction methods is the lack of a strong numerical tool to go with the well-developed theory. For example, minimal realization theory of Kalman (see Sec. 3.9), offers a clear understanding of the internal structure of linear systems. The associated discussions on controllability, observability, and minimal realization often illustrate the points, but numerical algorithms are adequate only for low-order textbook examples. Furthermore, there has been little connection made between minimal realization, controllability and observability, and model reduction on the other hand. Moore (1981) proposed to use the "Principle Component Analysis" of statistics along with some algorithms for the computation of "singular value decomposition" (see Appendix A) of matrices to develop a model reduction scheme which makes the most controllable and observable modes of the system transparent. Under a certain matrix transformation, the system is said to be "balanced" and the most controllable and observable modes would become prime candidates for reduced-order model states.

Consider an asymptotically stable, controllable and observable linear time-invariant system \((A, B, C)\) defined by

\[
\dot{x}(t) = Ax(t) + Bu(t)
\]

(7.34)

\[
y(t) = Cx(t)
\]

(7.35)

where \(x, u, y, A, B,\) and \(C\) are defined as before. The balanced matrix method (Moore, 1981) is based on the simultaneous diagonalization of the positive definite controllability and observability Gramians of Eqs. (7.34) and (7.35), (see also Secs. 3.3 and 3.4), that is,

\[
G_c = \int_0^\infty e^{At}BB^T e^{At} dt
\]

(7.36)

and

\[
G_o = \int_0^\infty e^{At}C^T Ce^{At} dt
\]

(7.37)

The Gramian matrices \(G_c\) and \(G_o\) satisfy the following Lyapunov-type equations (see Prob. 7.6):

\[
G_c A^T + A G_c + BB^T = 0
\]

(7.38)

\[
G_o A + A^T G_o + C^T C = 0
\]

(7.39)

The balanced approach of model reduction is essentially the computation of a similarity transformation matrix \(S\) such that both \(G_c\) and \(G_o\) become equal and diagonal, that is, balanced. This transformation matrix is given by (Moore, 1981),

\[
S = V D P \Sigma^{-1/2}
\]

(7.40)
where orthogonal matrices $V$ and $P$ satisfy the following symmetric eigenvalue/eigenvector problems.

\begin{equation}
V^T G_c V = D^2
\tag{7.41}
\end{equation}

\begin{equation}
P^T [(VD)^T G_0 (VD)] P = \Sigma^2
\tag{7.42}
\end{equation}

and

\begin{equation}
\Sigma = S^T G_0 S = S^{-1} G_c (S^{-1})^T
\tag{7.43}
\end{equation}

\begin{equation}
= \text{diag} (\sigma_1, \sigma_2, \ldots, \sigma_n)
\end{equation}

Here $D$ is a diagonal matrix like $\Sigma$. The diagonal elements of $\Sigma$ have the property that $\sigma_1 \leq \sigma_2 \leq \ldots \leq \sigma_n > 0$ and are called second-order modes of the system by Moore (1981). Using the transformation $\hat{x} = S^{-1} x$, one obtains the following full-order equivalent system,

\begin{equation}
\dot{\hat{x}} = \hat{A} \hat{x} + \hat{B} u
\tag{7.44}
\end{equation}

\begin{equation}
y = \hat{C} \hat{x}
\tag{7.45}
\end{equation}

where

\begin{equation}
\hat{A} = S^{-1} A S, \hat{B} = S^{-1} B, \hat{C} = C S
\tag{7.46}
\end{equation}

Now, if $\sigma_r \gg \sigma_{r+1}$ for a given $r$, and internally dominant reduced-order model of order $r$ can be obtained from Eqs. (7.44) and (7.45) by

\begin{equation}
\dot{z} = F z + G u
\tag{7.47}
\end{equation}

\begin{equation}
\dot{y} = Hz
\tag{7.48}
\end{equation}

where $(F, G, H)$ matrices are represented by the following partitioned matrices

\begin{equation}
\hat{A} = \begin{bmatrix}
F & \hat{A}_{12} \\
- & \hat{A}_{21} & \hat{A}_{22}
\end{bmatrix}, \quad 
\hat{B} = \begin{bmatrix}
G \\
- & \hat{B}_2
\end{bmatrix}, \quad 
\hat{C} = [H | \hat{C}_2]
\tag{7.49}
\end{equation}

Although this partitioning of second-order models leading to a reduced and a residual model are somewhat arbitrary, but grouping the most controllable and observable modes together does, as it is seen later, provide a reasonable criterion for model reduction.

Laub (1980) has proposed a more efficient method to compute the balancing transformation matrix given by

\begin{equation}
S = L_c U \Sigma^{-1/2}
\tag{7.50}
\end{equation}

where $L_c$ is lower triangular of the Cholesky decomposition (Moler, 1980) of the controllability Gramian $G_c$, $U$ is an orthogonal modal matrix and $\Sigma$ is the diagonal matrix to the symmetric eigenvalue/eigenvector problem of
\[ U^T(L_c^T G_o L_c) U = \Sigma^2 \]  

(7.51)

To show that \( G_o \) and \( G_c \) are diagonalized and equal, we note that

\[ \hat{G}_o = S^T G_o S \]  

(7.52)

\[ \hat{G}_c = S^{-1} G_c S \]  

(7.53)

It can be easily verified (see Prob. 7.8), that \( \hat{G}_c = \hat{G}_o = \Sigma \).

Laub et al. (1987) have presented an algorithm for computing state-space balancing transformation directly from a state-space realization. This algorithm is computationally much more efficient than the one reported earlier by Laub (1980). One difference is that it avoids the so-called "squaring up" problem, that is, neither \( BB^T \) nor \( C^T C \) in Eqs. (7.38) and (7.39) need to be formed explicitly, respectively. The new algorithm is given below.

**Algorithm 7.1**

1. Compute the Cholesky factors of the Gramians. Let \( L_c \) and \( L_o \) denote the lower triangular Cholesky factors of \( G_c \) and \( G_o \), that is,

\[ G_c = L_c L_c^T \quad G_o = L_o L_o^T \]  

(7.54)

This step is accomplished using an algorithm by Hammarling (1982) which would avoid the formation of \( G_c \) or \( G_o \) themselves.

2. Compute the singular value decomposition of the product of the Cholesky factors; that is,

\[ L_o^T L_c = V \Sigma U^T \]  

(7.55)

3. Form the balancing transformation

\[ S = L_c U \Sigma^{-1/2} \]  

(7.56)

it is noted that

\[ S^{-1} = \Sigma^{-1/2} V^T L_o^T \]  

(7.57)

4. Form the balanced state-space matrices

\[ \hat{A} = S^{-1} A S = \Sigma^{-1/2} V^T L_o^T A L_c U \Sigma^{-1/2} \]  

(7.58)

\[ \hat{B} = S^{-1} B = \Sigma^{-1/2} V^T L_o^T B \]  

(7.59)

\[ \hat{C} = C S = C L_c U \Sigma^{-1/2} \]  

(7.60)

There are two major differences between this algorithm and Laub’s (1980) earlier one. One is, as mentioned before, the elimination of the “squaring up” problem in finding \( G_c \) and \( G_o \) in Eqs. (7.38) and (7.39). The other is the difference between singular-value decomposition algorithm of Step 2, that is, Eq. (7.55) versus the solution of the symmetric eigenvalue/eigenvector problem of Eq. (7.51). For further details on this scheme of model reduction, refer to Laub et al. (1987).
Example 7.3

Consider a fourth-order SISO system,

\[
\dot{x} = \begin{bmatrix}
0 & 0 & 0 & -150 \\
1 & 0 & 0 & -245 \\
0 & 1 & 0 & -113 \\
0 & 0 & 1 & -19
\end{bmatrix} x + \begin{bmatrix}
4 \\
1 \\
0 \\
0
\end{bmatrix} u
\]

\[
y = [0\ 0\ 0\ 1] x
\]

Find a reduced-order model using the balanced method.

Solution  The controllability and observability Gramians are obtained by solving Eqs. (7.38) and (7.39), that is,

\[
G_c = \begin{bmatrix}
13.5 & 6.8 & 1.05 & 0.05 \\
6.8 & 3.6 & 0.58 & 0.03 \\
1.05 & 0.58 & 0.098 & 0.005 \\
0.05 & 0.03 & 0.005 & 0.0003
\end{bmatrix}
\]

\[
G_o = \begin{bmatrix}
6.7 & 10.9 & 4.5 & 0 \\
10.9 & 18.4 & 8.2 & 0.045 \\
4.5 & 8.2 & 4.75 & 0.073 \\
0 & 0.045 & 0.073 & 0.03
\end{bmatrix}
\]

The transformation matrix \( S \) is given by

\[
S = \begin{bmatrix}
29.1 & -4.0 & 0.55 & -0.31 \\
14.8 & 5.4 & -0.55 & 0.42 \\
2.3 & 2.1 & -0.03 & -0.12 \\
0.12 & 0.13 & 0.05 & 0.007
\end{bmatrix}
\]

The transformed state-space matrices are given by

\[
\hat{A} = \begin{bmatrix}
-0.44 & -1.17 & -0.41 & -0.05 \\
1.17 & -3.13 & -2.83 & -0.33 \\
-0.41 & 2.83 & -12.5 & -3.2 \\
0.05 & -0.33 & 3.2 & -2.95
\end{bmatrix}
\]

\[
\hat{B} = \begin{bmatrix}
0.12 \\
-0.13 \\
0.05 \\
-0.007
\end{bmatrix}, \hat{C} = [0.12\ 0.13\ 0.05\ 0.007]
\]

One can now extract a third, second, or a first-order reduced model. The best reduced-order model is subject to other research (Jamshidi, 1984), but through the relative magnitudes of second-order modes \( \Sigma^2 = \text{diag} (0.016, 0.003, 0.001, 0.0) \) one may often find an appropriate order for the reduced model. For example, here it is anticipated that a second order reduced model is fairly close to the full model. This is also evident by a comparison of the step responses of the full-order model and the first, second and third reduced models, shown in Fig. 7.8.

The balanced realization and its reduced-order modeling has been programmed into a command in most CAD packages such as MATRIXx, CTRL-C, PRO-MATLAB,
and CONTROL.lab. However, one can use commands such as LYAP or GRAM, CHOL, SVD, and so on, (see Prob. 7.9), to write one's own set of commands to find a balanced state-space formulation of a linear time-invariant system. Moreover, the extension of the balanced realization to unstable systems as well as to discrete-time case can be easily accomplished. For the former, refer to Santiago and Jamshidi (1986) and Probs. 7.11 and 7.12; while for the latter, refer to the work of Laub, et al. (1987).

**CAD Example 7.4**

Using MATRIXx and PC-MATLAB, the fourth-order system of Ex. 7.3 is reduced to a lower order (1, 2 or 3) model. The impulse responses of both full and reduced-order models will be compared.

**MATRIXx**

\[
<> \ a = < 0 \ 0 \ 0 \ -150; \ 1 \ 0 \ 0 \ -245; \ 0 \ 1 \ 0 \ -113; \ 0 \ 0 \ 1 \ -19>;
\]
\[
<> \ b = < 4; \ 1; \ 0 >; \ c = < 0 \ 0 \ 0 \ 1 >; \ d = < 0 >; \ s = < a,b;c,d>; \ ns = 4;
\]
\[
<> \ <> \ sb, \ sigsq, \ t > = \text{balance} \ (<s,ns>)
\]

<table>
<thead>
<tr>
<th>T</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>29.0903</td>
<td>-4.0562</td>
<td>0.5526</td>
<td>-0.3095</td>
</tr>
<tr>
<td>14.7840</td>
<td>5.4494</td>
<td>-0.5565</td>
<td>0.4256</td>
</tr>
<tr>
<td>2.3226</td>
<td>2.0930</td>
<td>-0.0296</td>
<td>-0.1217</td>
</tr>
<tr>
<td>0.1181</td>
<td>0.1307</td>
<td>0.0563</td>
<td>0.0069</td>
</tr>
</tbody>
</table>
Sec. 7.2 Aggregated Models of Large-Scale Systems

SIGSQC =
0.0159 0.0000 0.0000 0.0000
0.0000 0.0027 0.0000 0.0000
0.0000 0.0000 0.0001 0.0000
0.0000 0.0000 0.0000 0.0000

SB =
-0.4378 -1.1685 -0.4143 -0.0510 0.1181
1.1685 -3.1353 -2.8352 -0.3289 -0.1307
-0.4143 2.8352 -12.4753 -3.2492 0.0563
0.0510 -0.3289 3.2492 0.0069 0.0000
0.1181 0.1307 0.0563 0.0000 0.0000

// 1st order reduced model
<> s1 = <sb(1,1), sb(1,5); sb(5,1), 0>;
<> // 2nd order reduced model
<> a2 = split (sb,2); b2 = <sb(1,5); sb(2,5)>; c2 = <sb(5,1), sb(5,2)> ;
    <> s2 = <a2, b2; c2, 0>

S2 =
-0.4378 -1.1685 0.1181
1.1685 -3.1353 -0.1307
0.1181 0.1307 0.0000

<> // 3rd order reduced model
<> a3 = split (sb,3); b3 = <b2; sb(3,5)>; c3 = <c2, sb(5,3)>;
    <> s3 = <a3, b3; c3, 0>

S3 =
-0.4378 -1.1685 -0.4143 0.1181
1.1685 -3.1353 -2.8352 -0.1307
-0.4143 2.8352 -12.4753 0.0563
0.1181 0.1307 0.0563 0.0000

<> // step responses
<> < time, y1 > = step (s1,1,10,100);
<> < time, y2 > = step (s2,2,10,100);
<> < time, y3 > = step (s3,3,10,100);
<> < time, yf > = step (s, ns, 10, 100);
<> y = < y1(:,1), y2(:,1), y3(:,1), yf(:,1)>;
<> Plot (time, y; ‘xlab/time/ylab/yf,y1,y2,y3/Chart = 10, 100, 0, 601’)

The impulse responses of the full order and all three possible reduced-order models are shown in Fig. 7.9. The response for third-order reduced model is fairly close.
Figure 7.9  Impulse responses of a full-order and three reduced-order models for CAD Example 7.4. Balanced realization.

PC-MATLAB

```matlab
>> a = [0 0 0 -150; 1 0 0 -245; 0 1 0 -113; 0 0 1 -19];
>> b = [4; 1; 0; 0]; c = [0 0 0 1]; d = 0;
>> [ab,bb,cb,sig,tr] = balreal(a,b,c);
>> % second-order reduced model
>> f1 = ab(1:1, 1:1); g1 = bb(1:1,:); h1 = cb(:,1:1);
>> f2 = ab(1:2, 1:2), g2 = bb(1:2,:), h2 = cb(:,1:2)

f2 =
-0.4378  -1.1681
 1.1681  -3.1353

g2 =
 0.1179
-0.1304

h2 =
 0.1183  0.1310
```

...
Sec. 7.2 Aggregated Models of Large-Scale Systems

\[
\begin{align*}
&\gg \text{ia1} = \text{impulse (f1,g1,h1,d1,t);} \\
&\gg \text{ia2} = \text{impulse (f2,g2,h2,d1,t);} \\
&\gg \text{ia3} = \text{impulse (f3,g3,h3,d1,t);} \\
&\gg \text{Plot (t, im, t, ia1)} \\
&\gg \text{Plot (t, im, t, ia2)} \\
&\gg \text{Plot (t, im, t, ia3)} \\
&\text{:}
\end{align*}
\]

The impulse responses of the full-order model with three successive reduced-order models are shown in Fig. 7.10. These plots are identical to those of Fig. 7.9. It is noted that the impulse response of the first-order reduced model is somewhat far apart from the fourth-order full model. However, the second order response is fairly close, while the third-order reduced model has a very close and indistinguishable response. This characteristic is also clear from the relative magnitude of third and fourth Moore’s second-order modes that is, \(d_{33}\) and \(d_{44}\) in matrix \(D^2\) (or SIGSQ on page 311).

7.2.4 Near-Optimum Design via Aggregated Models

So far, a number of model reduction methods have been proposed. Here the aggregation procedure is used to design a near-optimum control system. Consider a large-scale linear time-invariant system

\[
\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0 \tag{7.61}
\]

with a quadratic cost functional

\[
J = \frac{1}{2} \int_0^\infty \left[ x^T(t) \tilde{Q} x(t) + u^T(t) \tilde{R} u(t) \right] dt \tag{7.62}
\]

where \(A\), \(B\), \(x\), and \(u\) are \((n \times n)\), \((n \times m)\), \(n\), and \(m\)-dimensional system matrix, control matrix, state vector, and control vector, respectively, \(\tilde{Q}\) and \(\tilde{R}\) are \(n \times n\) positive semidefinite and \(m \times m\) positive-definite matrices. The optimal control problem, as discussed in Chap. 6, is to find a control vector \(u^*(t)\) such that Eq. (7.61) is satisfied while the cost functional Eq. (7.62) is minimized. The solution to this “state regulator” problem is given by:

\[
u^*(t) = -\tilde{R}^{-1}B^T K x(t) \tag{7.63}\]

Here, \(K\) is an \(n \times n\) symmetric positive-definite matrix solution of the algebraic matrix Riccati equation (AMRE)

\[
KA + A^T K - KSK + \tilde{Q} = 0 \tag{7.64}
\]

where \(S = B \tilde{R}^{-1}B^T\). The solution of AMRE Eq. (7.64) was discussed in detail in Sec. 6.2.3.

For any value of \(n\), the optimal control problem Eqs. (7.61) and (7.62) and its solution Eqs. (7.63) and (7.64) requires a set of at least \(n(n + 1)/2\) elemental values of the \(n \times n\) symmetric positive-definite Riccati matrix \(K\). Clearly, for a large \(n\), the task of solving the AMRE Eq. (7.64), as seen in Chap. 6, calls for considerable computational effort. Although there are several approximate solutions of the AMRE Eq. (7.64) through regular or singular perturbations techniques (Jimshidi, 1983), here
Figure 7.10  Impulse responses of full model and the balanced models for:
(a) First-order balanced model CAD Example 7.4 (PC-MATLAB);
(b) Second-order; and (c) Third-order balanced model.
the approximation is assumed through the representation of the state model Eq. (7.61) by a 'coarser' set of states called 'aggregated' system,

\[ \dot{z}(t) = Fx(t) + Gu(t), \quad z(0) = z_0 \]  

(7.65)

where \( z = Rx \) is the \( k \)-dimensional aggregated state vector, \( R \) is the \( k \times n \) aggregation matrix, and \( F \) and \( G \) are \( k \times k \) and \( k \times m \) aggregated state and control matrices obtained from (see Sec. 7.2.1).

\[ F = RAR^T(RR^T)^{-1}, \quad G = RB \]  

(7.66)

For the full model, the optimal control is given by

\[ u^*(t) = -\tilde{R}^{-1}B^T K_f x(t) = -F^*x(t) \]  

(7.67)

where \( K_f \) is the solution of the full model's AMRE

\[ A^T K_f + K_f A - K_f S K_f + \dot{Q} = 0 \]  

(7.68)

where \( S = B\tilde{R}^{-1}B^T \). The control for the aggregated model is

\[ u^a(t) = -\tilde{R}^{-1}G^T K_a z(t) \]  

(7.69)

where \( K_a \) is the solution of the aggregated model's AMRE

\[ F^T K_a + K_a F - K_a G \tilde{R}^{-1} G^T K_a + \dot{Q}_a = 0 \]  

(7.70)
Using the aggregation conditions Eqs. (7.7) and (7.8), that is, \( FR = RA, \ G = RB, \) pre- and postmultiplying Eq. (7.70) by \( R^T \) and \( R, \) respectively, and simplifying, the aggregated model’s AMRE becomes

\[
A^T(R^TK_aR) + (R^TK_aR)A - (R^TK_aR)S (R^TK_aR) + R^T\tilde{Q}_aR = 0
\]

(7.71)

which is identical to the full model’s AMRE Eq. (7.68) if the following relations hold

\[
K_f = R^TK_aR, \ \ \ \ \tilde{Q} = R^T\tilde{Q}_aR
\]

(7.72)

Using the pseudoinverse of \( R, \ \ \tilde{Q}_a \) can be written as

\[
\tilde{Q}_a = (RR^T)^{-1}R\tilde{Q}R^T(RR^T)^{-1}
\]

(7.73)

Thus, the aggregated model’s near-optimum control is given by

\[
u^a(t) = -\tilde{R}^{-1}G^TK_aRx(t) = -F^a x(t)
\]

(7.74)

The following example illustrates this near-optimum control based on aggregation.

Example 7.4

Consider a fifth-order system

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\dot{x}_4 \\
\dot{x}_5
\end{bmatrix} =
\begin{bmatrix}
-0.2 & 0.5 & 0 & 0 & 0 \\
0 & -0.5 & 1.6 & 0 & 0 \\
0 & 0 & -14.28 & 85.71 & 0 \\
0 & 0 & 0 & -25 & 75 \\
0 & 0 & 0 & 0 & -10
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
30
\end{bmatrix}
\]

(7.75)

which represents a voltage regulator system (Jamshidi, 1983). It is desirable to find an aggregated model for Eq. (7.75) and a near-optimum control with a quadratic cost

\[
J = \frac{1}{2} \int_0^\infty (0.1x_1^2 + 0.01x_3^2 + u^2) \, dt
\]

(7.76)

Solution  A careful look at Eq. (7.75) indicates that the system matrix is upper triangular and its eigenvalues are \(-0.2, -0.5, -14.28, -25, \) and \(-10.\) Thus a two-dimensional modally reduced-order model is sought. For a two-dimensional (modally reduced model), the aggregation and aggregated matrices for Eq. (7.75) turn out to be

\[
R =
\begin{bmatrix}
1 & 0 & -0.004 & -0.00224 & -0.336 \\
0 & 1 & 0.115 & 0.40400 & 3.22
\end{bmatrix}
\]

(7.77)

\[
F =
\begin{bmatrix}
-0.2000 & 0.50 \\
0.0087 & -0.58
\end{bmatrix}, \ G =
\begin{bmatrix}
-10.08 \\
96.60
\end{bmatrix}
\]

Using the full model matrices \( (A, B, \tilde{Q}, \tilde{R}) \) defined in Eqs. (7.75) and (7.76), a fifth-order AMRE Eq. (7.68) is solved and an optimal feedback law is obtained

\[
u^*(t) = -\tilde{R}^{-1}B^TK_fx(t) = -0.26x_1 -0.11x_2 -0.04x_3 -0.15x_4 -0.59x_5
\]

(7.78)
where $K_f$ corresponds to the full model’s Riccati matrix. Next, by virtue of matrices $(F, G, \tilde{Q}_a, \tilde{R})$ defined in Eqs. (7.77) and (7.76), and using Eq. (7.73) to find $\tilde{Q}_a = \text{diag}(0.1\ 0.0)$, a second-order AMRE is solved. A feedback law is obtained for the aggregated system

$$u^a(t) = -\tilde{R}^{-1}G^T\tilde{K}_a x$$

which can result in an approximate feedback law

$$u^a(t) = -\tilde{R}^{-1}G^T\tilde{K}_a Rx = -0.60x_1 -0.27x_2 -0.0288x_3 -0.096x_4 -0.67x_5 \quad (7.79)$$

The two control laws Eqs. (7.78) and (7.79) provide the optimum and near-optimum controllers, respectively. The optimum and near-optimum (aggregated) output responses are shown in Fig. 7.11, respectively. The initial state was chosen to be $x(0) = (0.5 \ 0 \ 0 \ 0 \ 0)^T$, and the output was $y = x_1$ in both cases.

The following CAD example illustrates the use of this near-optimum design methodology for a balanced aggregation problem using PC-MATLAB.

**CAD Example 7.5**

In this CAD example, the fourth-order system of Ex. 7.3 which was reduced using balanced realization is used to design a near-optimum controller. To achieve this, a PC_MATLAB .m file called nearo was written to extract the reduced-model from an already balanced
system, find aggregation matrix, find $\tilde{Q}_a$ [see Eq. (7.73)], find the near-optimum feedback, find the Riccati matrix, and simulate for an output response, all in the same piece of code

```matlab
g >> a = [0 0 0 -150; 1 0 0 -245; 0 1 0 -113; 0 0 1 -19];
g >> b = [4 1 0 0]; c = [0 0 0 1]; d = 0;
g >> [ab,bb,cb] = balreal (a,b,c);
g >> qt = diag ([0.05 0.1 0.01 0.1]); rt = 1;
g >> [fbf, ric] = lqr (a,b,qt,rt);
g >> fbf

fbf =

```

0.1735 0.0035

-0.2207 -0.1343
g >> ac = a - b * fbf;
g >> % simulate optimum system
g >> t = [0:0.1:10]; uin = 0 * t; xo = [1 0 0 0];
g >> [y,x] = lsim (ac,b,c,d,uin,t,xo);
g >> % find 3 near-optimum control designs
g >> [y1,x1] = nearo (ab,bb,cb,d,qt,rt,xo,1);
g >> [y2,x2] = nearo (ab,bb,cb,d,qt,rt,xo,2);
g >> [y3,x3] = nearo (ab,bb,cb,d,qt,rt,xo,3);
g >> Plot (t,y,t,y1,t,y2,t,y3)
g >> title ('Optimum & Near-Optimum Responses - Full & 3 Balanced Models')
g >> xlabell('Time'), ylabell('y* & yn')
```

Following is a listing of "nearo.m" file for this CAD example.

```matlab
function [y,x] = nearo (ab,bb,cb,d,q,v,xo,k)
% dx/dt = Ax + Bu, y = cx + Du - balanced
% minimize J = 1/2 Int (x'Qtx + u' Rtu)dt
% Aggregated Model: dz/dt = Fz + Gu, A = Rx
% Extract kth order balanced reduced model
f = ab [1 : k, 1 : k]; g = bb [1 : k, :];
% Find aggregation matrix using controllability method
% R = Wf * Wat, Wf = [b, ab, . . ., a \wedge n-1 b] and
% Wa = [g,fg, . . ., f \wedge k-1 g], + is pseudo-inverse
Wa = [bb, ab * bb, ab \wedge 2 * bb, ab \wedge 3 * bb];
if k = 1
    WF = [g , f * g];
if k = 2
    Wf = [g, f * g, f \wedge 2 * g];
if k = 3
    Wf = [g, f * g, f \wedge 2 * g, f \wedge 3 * g];
r = wfr * pinv (wa);
```
Figure 7.12 A time response comparison of an optimum and three near-optimum designs based on balanced methods for CAD Example 7.5.

% linear quadratic regulator
qa = inv (r * r') * r * q * r' * inv (r * r')
[fba, rica] = lqr (f, q, qa, r);

% closed-loop matrix
fb = fba * r;
ac = a - b * fb;

t = [0 : 0.1 : 10]; uin = 0 * t;
[y,x] = lsim (ac,bb,cb,d,t,x0);
end

Figure 7.12 shows a time response of optimum and three balanced method-based near-optimum controls. As seen, there is a definite improvement of performance for higher-order models as expected.

7.3 HIERARCHICAL CONTROL

The notion of a large-scale system, as it was briefly discussed in Sec. 7.1, may be described as a complex system composed of a number of constituents or smaller subsystems serving particular functions, and sharing resources and are governed by interrelated goals and constraints. Although interconnection among subsystems can take on many forms, one of the most common ones is hierarchical, which appears somewhat natural in complex industrial, economic, management, and organization systems.
Within this hierarchical structure, the subsystems are positioned on levels with different degrees of hierarchy. A subsystem at a given level controls or coordinates the units on the level below it and is, in turn, controlled or coordinated by the unit on the level immediately above it. Figure 7.13 shows a typical hierarchical (multilevel) system. The highest level coordinator, sometimes called the *supremal coordinator*, can be thought of as the board of directors of a corporation, while other level’s coordinators may be the president, vice president, directors, and so on. The lower levels can be occupied by plant managers, shop managers, and so forth, while the large-scale system is the plant itself. In spite of this seemingly natural representation of a hierarchical structure, its exact behavior has not been well understood mainly because of the fact that relatively little quantitative work has been done on these large-scale systems.

There is no uniquely or universally accepted set of properties associated with the hierarchical systems. However, the following are some of the key properties.

1. A hierarchical system consists of decision-making components structured in a pyramid shape (Fig. 7.13).
2. The system has an overall goal which may or may not be in harmony with all its individual components.
3. The various levels of hierarchy in the system exchange information (usually vertically) among themselves interactively.

4. As the level of hierarchy goes up, the time constant increases: that is, the lower level components are faster than the higher level ones.

Based on this discussion, one can make a tentative conclusion that a successful operation of hierarchical system is best described by two processes known as decomposition and coordination. A pictorial representation of these two notions is shown in Fig. 7.14. In summary, the basic notions behind hierarchical control are to: (a) decompose a given large-scale system into a number of small-scale subsystems, and (b) coordinate these subsystems' solutions until feasibility and optimality of the overall system is achieved through a multilevel iterative algorithm.

### 7.3.1 Goal Coordination—Interaction Balance

The key issue in hierarchical control is to achieve a coordinated feasible optimal control among a finite number of decomposed subsystems. Consider a linear large-scale system described by the following state equation,

![Diagram of an interconnected and hierarchically structured system](image)

**Figure 7.14** A pictorial representation of notions of “decomposition” and “coordination” in a hierarchical control system.
\[ \dot{x}(t) = Ax(t) + Bu(t), \quad x(t_0) = x_0 \]  
(7.80)

and a quadratic cost function (see Sec. 6.2), to be minimized,

\[ J = \frac{1}{2} x^T(t_f) F x(t_f) + \frac{1}{2} \int_{t_0}^{t_f} (x^T Q x + u^T R u) \, dt \]  
(7.81)

where \( F \succeq 0, \quad Q \succeq 0, \quad R > 0, \quad t_0, \quad t_f \) are initial and final values of time and \( x_0 \) is the initial state. All the remaining terms are as defined before. Assume that the order \( n \) of system Eq. (7.80) is too large, that is, \( n > 100 \), and for computational simplicity, one can decompose it into \( N \) subsystems described by

\[ \dot{x}_i(t) = A_i x_i(t) + B_i u_i(t) + z_i(t), \quad x_i(t_0) = x_{i0}, \quad i = 1, \ldots, N \]  
(7.82)

where vector \( z_i(t) \) defined by

\[ z_i(t) = \sum_{j \neq 1}^{N} G_{ij} x_j(t) \]  
(7.83)

describes the \( i \)th subsystem’s interaction with the remaining \( N - 1 \) subsystems. The original optimal control problem for the overall system is reduced to the optimization of \( N \) subsystems, which collectively satisfy Eqs. (7.82) and (7.83) while minimizing

\[ J = \sum_{i=1}^{N} \left\{ \frac{1}{2} x_i^T(t_f) F_i x_i(t_f) + \frac{1}{2} \int_{t_0}^{t_f} [x_i^T(t) Q_i x_i(t) \\
+ u_i^T(t) R_i u_i(t) + z_i^T(t) S_i z_i(t)] \, dt \right\} \]  
(7.84)

where \( Q_i \) and \( F_i \) are \( n_i \times n_i \) positive semidefinite matrices, \( R_i \) and \( S_i \) are, respectively, \( m_i \times m_i \) and \( n_i \times n_i \) positive definite matrices with \( n = \sum n_i \) and \( m = \sum m_i \). It is noted that matrices \( Q_i \) and \( R_i \) are block diagonal, while \( S_i \) is antiblock-diagonal. This would account for a decomposition of \( Q \) and \( R \) matrices from Eqs. (7.81) to (7.84).

In this decomposition of a large interconnected linear system, the common coupling factors among its \( N \) subsystems are the interaction variables \( z_i(t) \) in Eq. (7.83), which, along with Eqs. (7.82) and (7.84) is called the global problem and is denoted by \( S_G \). The following assumption is considered to hold. The global problem is replaced by a family of \( N \) subproblems coupled together through a parameter vector \( \alpha = (\alpha_1, \ldots, \alpha_{m_i})^T \) and denoted by \( S_i(\alpha) \), \( i = 1, \ldots, N \). In other words, the global system problem is imbedded into a family of subsystem problems through an imbedding parameter \( \alpha \) in such a way that for a particular value of \( \alpha^* \) of \( \alpha \), the subsystems \( s_i(\alpha^*), \quad i = 1, 2, \ldots, N \), yield the desired solution to \( S_G \). In terms of hierarchical control notation, this imbedding concept is nothing but the notion of coordination. Figure 7.15 shows a two-level control structure of a large-scale system. Under this strategy at the \( k \)th iteration (or information exchange step) each local controller \( i \) receives \( \alpha_i^k \) from the coordinator (second-level hierarchy), solves \( S_i(\alpha_i^k) \),
and transmits (reports) some function $y^k_i$ of its solution to the coordinator. The coordinator's problem is basically a "dual" to the one at subsystem's level, that is, adjust the vector of language multipliers, $\alpha$ such that the interaction error $e = z - \Sigma_{ij}G_{ij}x_j$ is driven down to zero. This can be achieved by any functional minimization scheme.

The minimization of a function of several variables falls under the general area of optimization which deserves a great deal of attention. However, because of the lack of space, a detailed treatment of this subject can not be made and is subject to numerous texts, see for example, Fletcher (1980). In order to choose a proper optimization method, several considerations on the type of algorithm, convergence rate, and so on, will need to be made. One of the more popular classes of optimization methods is based on the gradient of the functional $f(x)$ of unknown vector $x = (x_1, x_2, \ldots x_n)^T$. Among the gradient methods, *steepest descent method* is one of the most logical schemes. In this method, the direction of search for an optimum (minimum or maximum) is given by $s^{(k)} = -g^{(k)}$, where $s^{(k)}$ is the search direction during the $k$th iteration and $g^{(k)} = \partial f(x^{(k)})/\partial x^{(k)}$ is the gradient vector. Here, the method searches in the steepest descent direction, along which the objective function decreases most rapidly local to $x^{(k)}$. This consideration, although very appealing, in practice exhibits oscillatory behavior. Moreover, even though there is a convergence proof for the steepest descent method, it usually falls too far from the solution as a result of rounded-off errors. Thus, this method is often unreliable and inefficient.

To avoid the inefficiency of steepest descent or gradient method, one can either resort to quadratic model of the function which leads to Newton-type approaches, or rely on the concept of the *conjugacy* of a set of nonzero vectors $s^{(i)}$, $i = 1, 2, \ldots, k$ to a given positive definite matrix $G$. This property is mathematically described by

$$s^{(i)} G x^{(j)} = 0$$

(7.85)

for all $i \neq j$. A *conjugate direction method* generates these directions when applied to a quadratic function with Hessian $G = \partial^2 f(\cdot)/\partial x^2$.

In the coordination of hierarchical control structures, conjugate gradient method is often used (Jamshidi, 1983). The unknown optimizing variables are the subvectors
\( \alpha_i, i = 1, 2, \ldots, N \) of the coordination vector \( \alpha = (\alpha_1^T \; \alpha_2^T \; \ldots \; \alpha_N^T)^T \). In this way, the coordinator evaluates the next updated value of \( \alpha \), that is,

\[
\alpha^{k+1} = \alpha^k + \epsilon^k d^k
\]  

(7.86)

where \( \epsilon^k \) is the \( k \)th iteration step size, and the update term \( d^k \) is defined from the following conjugate gradient iteration (Wismer, 1971)

\[
d^{k+1}(t) = \epsilon^{k+1}(t) + \gamma^{k+1} d^k(t), \quad 0 \leq t \leq t_f
\]  

(7.87)

where the step size \( \gamma^{k+1} \) in Eq. (7.87) is known to be

\[
\gamma^{k+1} = \frac{\int_{t_0}^{t_f} [\epsilon^{k+1}(t)]^T e^{k+1}(t) \ dt}{\int_{t_0}^{t_f} [e^k(t)]^T e^k(t) \ dt}
\]  

(7.88)

and \( d^0 = e^0 \). Note that for the first iteration, \( k = 1 \), the algorithm is equivalent to the steepest descent. The vector \( e(t) \) represents the "interaction error," whose \( i \)th component is defined by

\[
e_i(\alpha(t), t) = z_i(\alpha(t), t) - \sum_{j=1, j \neq i}^{N} G_{ij} x_j(\alpha(t), t)
\]  

(7.89)

The relations Eqs. (7.86) to (7.88) constitute the so-called second-level or coordinator problem. The "first-level" or "subsystem" problem is represented by Eqs. (7.82) to (7.84) which constitute a linear regulator problem and can be handled by a Riccati formulation (Sec. 6.2). By virtue of Fig. 7.13, each time the coordinator passes down a new (updated) coordination vector \( \alpha^* = (\alpha_1^* T \; \ldots \; \alpha_N^* T) \) until the "interaction balance" is achieved, that is, the normalized interaction error.

\[
Error = \frac{\left\{ \sum_{i=1}^{N} \int_{t_0}^{t_f} \left[ z_i(t) - \sum_{j=1}^{N} G_{ij} x_j(t) \right]^T \left[ z_i(t) - \sum_{j=1, j \neq i}^{N} G_{ij} x_j(t) \right] \ dt \right\}}{\Delta t}
\]  

(7.90)

is sufficiently small. Here \( \Delta t \) is the step size of integration.

### 7.3.2 Interaction Prediction

The interaction balance method just described requires excessive computations at both levels. The second-level problem, in particular, requires a computationally intensive method such as conjugate gradient. To initiate the first-level problem for \( i \)th subsystem, consider the following Hamiltonian function.
\[ H_i = \frac{1}{2} x_i^T(t) Q_i x_i(t) + \frac{1}{2} u_i^T(t) R_i u_i(t) + \alpha_i^T z_i \]
\[ - \sum_{j=1 \atop j \neq i}^N \alpha_j^T G_{ji}^T x_i + p_i^T (A_i x_i + B_i u_i + C_i z_i) \]  
(7.91)

Then utilizing the necessary conditions of optimality, through a Riccati formulation (see Sec. 6.3)

\[ p_i(t) = K_i(t) x_i(t) + g_i(t) \]  
(7.92)

and simplifying the resulting TPBV problem, one obtains

\[ \dot{K}_i(t) = - K_i(t) A_i - A_i^T K_i(t) + K_i(t) S_i K_i(t) - Q_i \]  
(7.93)

\[ \dot{g}_i(t) = - [A_i - S_i K_i(t)]^T g_i(t) - K_i(t) z_i(t) + \sum_{j=1 \atop j \neq i}^N G_{ji}^T \alpha_j^T(t) \]  
(7.94)

whose final conditions \( K_i(t_f) \) and \( g_i(t_f) \) follow from Eq. (7.84)

\[ p_i(t) = \frac{\partial}{\partial x_i(t_f)} \left[ \frac{1}{2} x_f^2(t_f) F_i x_i(t_f) \right] = F_i x_i(t_f) \]  
(7.95)

and through Eq. (7.92), that is,

\[ K_i(t_f) = F_i, \quad g_i(t_f) = 0 \]  
(7.96)

Following this formulation, the first-level optimal control function is given by

\[ u_i(t) = - R_i^{-1} B_i^T K_i(t) x_i(t) - R_i^{-1} B_i^T g_i(t) \]  
(7.97)

which has a partial feedback (closed-loop) term and a feedforward (open-loop) term. Two points are made here. First, the solution of the differential matrix Riccati equation is independent of the initial state \( x_i(0) \). The second point is that unlike \( K_i(t) \), \( g_i(t) \) in Eq. (7.94), by virtue of \( z_i(t) \), depends on \( x_i(0) \).

The second-level problem is essentially that of updating the new coordination vector. For this purpose, consider the \( i \)th subsystem Hamiltonian given by Eq. (7.91) and since the second-level problem is the dual of the first level (Jamshidi, 1983), the following relations result in the desired solution:

\[ \frac{\partial H_i}{\partial z_i} = \alpha_i + C_i^T p_i = 0 \]  
(7.98)

\[ \frac{\partial H_i}{\partial \alpha_i} = z_i - \sum_{j=1 \atop j \neq i}^N G_{ij} x_j = 0 \]  
(7.99)

which provide
Figure 7.16  Optimum time responses for an interaction prediction solution of CAD Example 7.6.

Part (a) Subsystem No. 1 state variables.
Part (b) Subsystem No. 1 control variables.
Part (c) Subsystem No. 2 state variables.
Part (d) Subsystem No. 3 control variables.
Figure 7.16 (Continued)
\[
\alpha_i(t) = -C_i^T p_i(t), \quad z_i(t) = \sum_{j=1, j \neq i}^{N} G_{ij} x_j
\] (7.100)

Thus, the second-level coordination procedure at the \((k + 1)\)th iteration is given by,

\[
\begin{bmatrix}
\alpha_i(t) \\
z_i(t)
\end{bmatrix}^{k+1} = \begin{bmatrix}
-C_i^T p_i(t) \\
\sum_{j=1, j \neq i}^{N} G_{ij} x_j(t)
\end{bmatrix}^k
\] (7.101)

The interaction prediction method is formulated by the following algorithm:

**Algorithm 7.2 Interaction Prediction**

Step 1. Solve \(N\) independent differential matrix Riccati Eq. (7.93) with final condition Eq. (7.96) and store \(K_i(t), i = 1, 2, \ldots, N\) and \(t_o \leq t \leq t_f\).

Step 2. For initial \(\alpha_i^k(t), z_i^k(t)\) solve the "adjoint" Eq. (7.94) with final condition Eq. (7.96). Evaluate and store \(g_i(i), i = 1, 2, \ldots, N\) and \(t_o \leq t \leq t_f\).

Step 3. Solve the state equation

\[
\dot{x}_i(t) = [A_i - S_i K_i(t)] - S_i g_i(t) + z_i(t), \quad x_i(0) = x_i
\] (7.102)

Step 4. At the second level, use the results of steps 2 and 3 and Eq. (7.101) to update the overall interaction error

\[
e(t) = \frac{1}{\Delta t} \sum_{i=1}^{N} \int_{t_o}^{t_f} \left[ z_i(t) - \sum_{j=1, j \neq i}^{N} G_{ij} x_j(t) \right] dt
\] (7.103)

where \(\Delta t\) is the step size of integration. It must be noted that depending on the type of digital computer and its operating system, subsystem calculations may be done in parallel and that the \(N\) matrix Riccati equations at step 1 are independent of \(x_i(0)\), and hence they need to be computed once regardless of the number of second-level iterations in the interaction prediction algorithm Eq. (7.101).

**Example 7.5**

In this simple example, the steps of the interaction prediction algorithm are illustrated. Consider a second-order system,

\[
\begin{align*}
\dot{x}_1 &= x_1 + 0.2x_2 + u_1, \quad x_1(0) = -1 \\
\dot{x}_2 &= 0.2x_1 + x_2 + u_2, \quad x_2(0) = -1
\end{align*}
\]

with cost function,
where \( x = (x_1, x_2)^T \) and \( u = (u_1, u_2)^T \). Find a hierarchical control policy.

**Solution** Following the steps of the algorithm, one needs to solve two Riccati equations Eq. (7.93). Assuming that \( t_f = 5 \) is approximately infinity, the differential matrix Riccati equations would reduce to algebraic. For the two subsystems, that is, \((A_i, B_i, Q_i, R_i) = (1, 1, 1, 1), i = 1, 2\), we have scalar Riccati values \( k_i = 2.4, i = 1, 2\). Using initial values \( \alpha_i^0 = \alpha_i^0 = 1 \) and \( z_i^0 = z_i^0 = \frac{1}{12} \), then the adjoint equations, Eq. (7.94) reduce to: \( \dot{g}_i = 1.4 \, g_i, g_i(5) = 0.0, i = 1, 2 \), whose solution for \( 0 \leq t \leq 5 \) would be, \( g_i(t) = \exp(1.4t) - \exp(7) \). Then, the third step calls for the solution of closed-loop system equations, Eq. (7.102),

\[
\dot{x}_i = -1.4x_i - e^{1.4t} + e^7 + \frac{1}{12}, \quad x_i(0) = -1, \quad i = 1, 2
\]

After solving these equations for \( 0 \leq t \leq 5 \), the interaction error

\[
e = 10 \left[ \int_0^5 (z_1 - 0.2x_2)dt + \int_0^5 (z_2 - 0.2x_1)dt \right]
\]

must be computed. Note that an arbitrary step size, \( \Delta t = 0.1 \) can be chosen in Eq. (7.103). If error \( e \) is not small enough, then the second-level problem must be solved. This is accomplished by updating \( \alpha_i \) and \( z_i \), \( i = 1, 2 \) by virtue of Eq. (7.101), that is,

\[
\alpha_i(t) = -p_i^0(t) = -2.4x_i(t) + g_i(t), \quad i = 1, 2
\]

\[
z_i(t) = \sum_{j=1}^{2} 0.2x_j(t), \quad i = 1, 2
\]

and the entire algorithm is repeated.

**CAD Example 7.6**

Consider a fourth-order system

\[
x = \begin{pmatrix} 2 & 0.1 \\ 0.2 & -1 \end{pmatrix} x + \begin{pmatrix} 0.01 & 0 \\ 0.10 & -0.5 \end{pmatrix} u
\]

with \( x(0) = (-1, 0.1, 1.0, -0.5)^T \) and a quadratic cost function with \( Q = \text{diag}(2, 1, 1, 2), R = \text{diag}(1, 2) \) and no terminal penalty. It is desired to use the interaction prediction method to find an optimal control with \( t_f = 2 \).

The system is divided into two second-order subsystems and the subsystem's Riccati equations are solved using RICRKUT of LSSPAK/PC and their solutions are fitted into fourth-order polynomial for computational convenience. Using program
INTRPRD\(^1\) of LSSPAK/PC, the interaction prediction algorithm is realized and converged in five iterations. The exact excerpts from running this CAD example follow. Instructions for plotting with interaction prediction program.

When you get a plot on the screen, hit return to return to the menu.

If you plan to dump plots to the printer, you must run the DOS file GRAPHICS prior to running this program. Then, when you wish to dump a plot hit shift-PrtSc. Optimization via the interaction prediction method.

Initial time (to): 0
Final time (tf): 2
Step size (Dt): .1
Total no. of 2nd level iterations = 6
Error tolerance for multi-level iterations = .00001
Order of overall large scale system = 4
Order of overall control vector \((r)\) = 2
Number of subsystems in large scale system = 2

Matrix Subsystem state orders--n sub i
0.200D + 01
0.200D + 01

Matrix Subsystem input orders--r sub i
0.100D + 01
0.100D + 01

Polynomial approximation for the Ricatti matrices to be used.

Matrix Ricatti coefficients for SS# 1
\[
\begin{array}{cccccc}
0.453D + 01 & -.259D + 01 & .794D + 01 & -.762D + 01 & .186D + 01 \\
.978D - 01 & -.793D - 01 & .252D + 00 & -.233D + 00 & .571D - 01 \\
.490D + 00 & .759D - 02 & -.109D + 00 & .975D - 01 & -.531D - 01
\end{array}
\]

Matrix Ricatti coefficients for SS# 2
\[
\begin{array}{cccccc}
.112D + 01 & -.815D + 01 & -.361D + 01 & .455D + 01 & -.105D + 01 \\
-.149D + 00 & -.322D - 01 & .697D - 01 & .284D - 01 & -.183D - 01 \\
.815D + 00 & .642D - 01 & -.295D + 00 & .305D + 00 & -.138D + 00
\end{array}
\]

System Matrix A
\[
\begin{array}{cccccc}
.200D + 01 & .100D + 00 & .100D - 01 & .000D + 00 \\
.200D + 00 & -.100D + 01 & .100D + 00 & -.500D + 00 \\
.500D - 01 & .150D + 00 & .100D + 01 & .500D - 01 \\
.000D + 00 & -.200D + 00 & -.250D + 00 & -.120D + 01
\end{array}
\]

Matrix Input Matrix B
0.100D+01  0.000D+00
0.100D+00  0.000D+00
0.000D+00  0.250D+00

Matrix Input Cost Function R
0.100D+01  0.000D+00
0.000D+00  0.200D+01

Matrix Lagrange Multiplier Initial Values
0.100D+01
0.100D+01
0.100D+01
0.100D+01

Matrix Initial conditions vector x0
-.100D+01
0.100D+00
0.100D+01
-.500D+00

Subsystem no. 1 at 2nd level iteration no. 1

Subsystem no. 2 at 2nd level iteration no. 1

At second level iteration no. 1 interaction error = 0.347D+00

Subsystem no. 1 at 2nd level iteration no. 2

Subsystem no. 2 at 2nd level iteration no. 2

At second level iteration no. 2 interaction error = 0.771D−03

Subsystem no. 1 at 2nd level iteration no. 3

Subsystem no. 2 at 2nd level iteration no. 3

At second level iteration no. 3 interaction error = 0.507D−03

Subsystem no. 1 at 2nd level iteration no. 4

Subsystem no. 2 at 2nd level iteration no. 4

At second level iteration no. 4 interaction error = 0.323D−04

Subsystem no. 1 at 2nd level iteration no. 5

Subsystem no. 2 at 2nd level iteration no. 5

At second level iteration no. 5 interaction error = 0.310D−05
Figure 7.17 Interaction error versus iteration for CAD Example 7.6.

Figures 7.16 and 7.17 show the states, inputs, and the interaction error trajectories for this problem after the algorithm has converged.

7.4 DECENTRALIZED CONTROL

As mentioned in Sec. 7.1, in many large-scale systems the notion of “centrality” does not hold. Under such conditions, the system often remains physically in one location while its output information is shared among $N$ controllers, called decentralized, which collectively contribute to control of the system. Therefore, the basic difference between decentralized and hierarchical control is the following. In hierarchical control, as seen in Sec. 7.3, the system is decomposed into subsystems whose solutions are coordinated by a higher level controller, for example, a supervisory type control. In decentralized control, on the other hand, the system’s local output is used by local feedback controllers which collectively control the system. In other words, in a decentralized system, unlike a centralized system, no single controller exist which receives information from every component of the output vector.

The main motivation behind the study of decentralized control is that conventional methods of centralized control can not be easily extended to decentralized systems. Some fundamental problems such as pole placement, state feedback, optimal control, state estimation of centralized control theory require complete information
from all system sensors to a centralized controller. This scheme is clearly inadequate for feedback control of large complex systems. Because of the physical configuration and often high dimensionality of such systems, a centralized control is neither economically feasible nor even necessary. Therefore, in many applications of feedback control theory to linear large-scale systems, some degree of restriction is assumed to prevail on the transfer of information. In some cases, a total decentralization is assumed; that is, every local control $u_i$ is obtained from the local output $y_i$ and possible external input $v_i$. In others, an intermediate restriction is placed on the information flow.

In this section, we consider the problem of decentralized control by pole assignment method.

### 7.4.1 Stabilization and Pole Assignment by Decentralized Output Feedback

Consider a large-scale linear time-invariant system with $N$ local control stations (channels),

$$
\dot{x}(t) = Ax(t) + \sum_{i=1}^{N} B_i u_i(t) \tag{7.104}
$$

$$
y_i(t) = C_i x, \quad i = 1, 2, \ldots, N \tag{7.105}
$$

where $x$ is an $n \times 1$ state vector, and $u_i$ and $y_i$ are $m_i \times 1$ and $r_i \times 1$ control and output vectors associated with the $i$th control station, respectively. The original system control and output orders $m$ and $r$ are given by $m = \sum_{i=1}^{N} m_i$, $r = \sum_{i=1}^{N} r_i$. Consider decentralized output feedback control law

$$
u_i = K_i y_i + v_i, \quad i = 1, \ldots, N \tag{7.106}
$$

where $K_i$ is a constant $m_i \times r_i$ matrix and $v_i$ is the $i$th command vector. The decentralized output feedback stabilization or decentralized pole placement problem is to find a set of gain matrices $K_i$, $i = 1, \ldots, N$ such that the eigenvalues of the closed-loop matrix $\lambda(A - B K_D C)$ are appropriately assigned. Here, $B$, $C$, and $K_D$ matrices are given by

$$
B = [B_1 B_2 \cdots B_N], \quad C = [C_1^T C_2^T \cdots C_N^T]^T \tag{7.107}
$$

$$
K_D = \text{Block diag } \{K_1, K_2, \ldots, K_N\} \tag{7.108}
$$

Although decentralized stabilization using state feedback has been treated extensively in literature (see, for example, Jamshidi, 1983 for 18 references), the solution of the problem using output feedback is not available. The main difficulty with this problem unlike the centralized output feedback control (see Chap. 5), is that it reduces to a nonlinear system of equations in elements of $K_i$ matrices. Recently, a solution to decentralized output feedback pole placement was proposed by Tarokh (1987).
Before presenting the details of the pole assignment algorithm, the following definitions are stated.

**Definition 7.1.** For the system \((A, B, C)\) and a set of output feedback gains \(K\), the set of fixed modes of \((A, B, C)\) with respect to \(K\) is defined as the intersection of all possible sets of the eigenvalues of matrix \((A + BK_D C)\), that is,

\[
\Lambda(A, B, K_D, C) = \bigcap_{K_D \in K} \lambda(A + BK_D C)
\]  

(7.109a)

where \(\lambda(\cdot)\) denotes the set of eigenvalues of \((A + BK_D C)\). Note also that \(K_D\) can take on the null matrix; hence the set of "fixed modes" \(\Lambda(\cdot)\) is contained in \(\lambda(A)\). In short, the fixed modes of a system are the eigenvalues of the system that are invariant under decentralized output feedback.

The fixed modes of centralized system \((A, B, C, \hat{K})\), where \(\hat{K}\) is \(m \times r\), correspond to the uncontrollable and unobservable modes of the system. The following theorem provides the necessary and sufficient conditions for the stabilizability of a decentralized closed-loop system.

**Theorem 7.1.** For the system \((A, B, C)\) in Eqs. (7.104) to (7.107) and the class of block-diagonal matrices \(K\) in Eq. (7.109a), the local dynamic feedback laws

\[
u_i = K_{di} y_i + H_i z_i + L_i v_i
\]  

(7.109b)

\[
\dot{z}_i = F_i y_i + S_i z_i + G_i v_i
\]  

(7.109c)

where \(H_i, L_i, F_i, S_i, G_i\) are gain matrices of appropriate dimensions, and \(K_{di}\) are \(n_i \times n_j\) dimensional block-diagonal submatrices of \(K_D\) in (7.109a). Then the controller (7.109b) would asymptotically stabilize the system if, and only if, the set of fixed modes of \((A, B, C)\) with respect to \(K\) is contained in the open left half complex plane. Consult Wang and Davison (1972) for a proof of this theorem.

**CAD Example 7.7**

In this CAD example, a PC-MATLAB.M file is written to evaluate the fixed modes of a decentralized system.

```matlab
function [m]=fixmod(a,b,c)
%FIXMOD calculates the fix modes of a multivariable system
% dx/dt = Ax + Bu , y=Cx, [m] = FIXMOD(A,B,C)
% m is vector of modes - fixed or otherwise.
[n,mu]=size(b);[r,n]=size(c);
m=eig(a+b+rand(mu,r)*c);

>> a = [-1 0 2 1 ; -.2 2.0 .1 ; 0 0 -2.0 ; 0 0 .5 .75];
>> b = [.5 0 ; 0 1 ; 0 2.5 ; 1 0];
>> c = [1 0 0 0 ; 0 0 .25 1];
>> eig(a)
    2.0000
    -0.7505
    -2.2857
    1.6572
```
>> m1 = fixmod(a,b,c)

\[
\begin{align*}
\text{m1} &= \\
&= 2.0000 \\
&= -3.1756 \\
&= 2.6455 \\
&= -0.3579
\end{align*}
\]

>> m2 = fixmod(a,b,c)

\[
\begin{align*}
\text{m2} &= \\
&= 2.0000 \\
&= -2.9302 \\
&= 2.1886 \\
&= -0.4415
\end{align*}
\]

It is clear that \( \lambda = 2 \) is the fixed mode of this system. By virtue of Theorem 7.1, this system can not be stabelized via decentralized control.

### 7.4.2 An Iterative Solution to Decentralized Output Control

In this section, the iterative output feedback design of Sec. 5.2 is extended to the decentralized case. Considering the decentralized control law Eq. (7.106) and output feedback matrix \( K_D \) in Eq. (7.108), the closed-loop characteristic polynomial is given by

\[
\hat{p}(s) = \left| sI - A \right| = \left| sI - A - BK_DC \right|
\]

\[
= \left| sI - A \sum_{j=1}^{N} B_j K_j C_j \right|
\]

\[
= s^n + \hat{p}_1 s^{n-1} + \cdots + \hat{p}_n
\]  

Equation (7.110) is essentially the same as Eq. (5.16), and therefore, the development leading to Eq. (5.23) can be applied to system Eqs. (7.104) and (7.105). We note that some of the elements of \( \delta k \) are zero due to the decentralized structure of \( K_D \). Let \( k_D \) be the \( \mu \times 1 \) vector formed from the columns of \( K_D \) after removing the zero elements of \( K_D \), where \( \mu = \sum_{j=1}^{N} m_j \). Then, Eq. (5.23) can be written as

\[
\delta p = \hat{\hat{p}} \hat{E}_D \delta k_D + \hat{\Phi}(\delta k_D)
\]  

where

\[
\hat{E}_D = \begin{pmatrix}
\hat{e}_{OD} \\
\vdots \\
\hat{e}_{n-1D}
\end{pmatrix}
\]  

and \( \hat{e}_{iD} \) is a \( 1 \times \mu \) vector obtained from \( \hat{e}_i \) after removing the elements of \( \hat{e}_i \) corresponding to the zero elements of \( \delta k \). Note that \( \hat{e}_i \) is formed from the rows of the matrix \( C(A + BK_DC)B \), where \( \hat{K}_D = \text{block-diag} (\hat{K}_1, \hat{K}_2, \ldots, \hat{K}_N) \) and \( \hat{K}_j, j = \ldots \)}
1, 2, . . . , N are arbitrary \( m_j \times r_j \) matrices. The following result is deduced from this discussion.

**Theorem 7.2.** Consider the decentralized system Eqs. (7.104) and (7.105) and the set of desired poles \( \Lambda_d = (\lambda_1, \lambda_2, \ldots, \lambda_n) \). Then, for almost all \( \Lambda_d \), a decentralized constant output feedback matrix \( K_D = \text{block-diag}(K_1, K_2, \ldots, K_N) \) exist such that \( A + \sum_{j=1}^{N} B_j K_j C_j \) has \( \Lambda_d \) as its eigenvalues if \( \text{rank} \ E_D = n \) where \( E_D \) is defined in Eq. (7.112).

**Remark 7.1.** A counterpart to Theorem 5.3 can be stated for decentralized systems by defining the matrix \( E_D \) whose rows are obtained from \( CA'B \) after removing the elements of \( CA'B \) corresponding to zero elements of \( K^T \). Then, decentralized pole assignment is possible if \( \text{rank} \ E_D = n \). It is noted that both \( \text{rank} \ \hat{E}_D = n \) and \( \text{rank} \ E_D = n \) are sufficient conditions for decentralized pole assignability.

**Remark 7.2.** Since \( \hat{E}_D \) is an \( n \times \mu \) matrix, a necessary condition for decentralized pole assignment is \( \mu = \sum_{j=1}^{l} m_j \mu_j \geq n \). When \( \text{rank} \ \hat{E}_D = n \), it is possible to assign \( \min(n, \sum_{j=1}^{l} m_j \mu_j) \) poles by decentralized constant output feedback.

**Remark 7.3.** It can be shown that when the system has decentralized fixed modes, \( \text{rank} \ \hat{E}_D < n \) and decentralized pole assignment is not possible. The proof of this statement is through the connection between controllability-observability and fixed modes (Taroikh; 1985, 1986).

**Remark 7.4.** Theorem 7.2 can be generalized to systems using any structurally-constrained output feedback matrix \( K_{sc} \), where some of the elements of \( K_{sc} \) are fixed (not necessarily zero). In this case, the matrix \( \hat{E}_D \) is obtained from the rows of the matrix \( C(A + B\hat{K}_{sc} C)B \) after removing the elements corresponding to the fixed elements of \( K_{sc} \), where \( \hat{K}_{sc} \) is a matrix with the specified structure whose free elements are chosen arbitrarily.

With this development, one can use Algorithm 5.2 to design a decentralized output feedback controller.

**Example 7.6**

Consider the decentralized system (Jamshidi, 1983)

\[
\begin{bmatrix}
-0.4 & 0.2 & 0.6 & 0.1 & -0.2 \\
0 & -0.5 & 0 & 0 & 0.4 \\
0 & 0 & -2 & 0 & 0.2 \\
0.2 & 0.1 & 0.5 & -1.25 & 0 \\
0.25 & 0 & -0.2 & 0.5 & -1
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5
\end{bmatrix}
+ 
\begin{bmatrix}
1 & -1 \\
2 & 1 \\
0 & 0 \\
0 & 0 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2
\end{bmatrix}
\]

\[
y_1 = \begin{bmatrix}1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \end{bmatrix} x, \quad y_2 = (0 & 0 & 0 & 1 & 1) x
\]

Preferably, place the closed-loop poles at \((-1, -1, -2, -3, -4)\) by decentralized output feedback.
Solution  For this system, we have
\[
B = (B_1, B_2) = \begin{pmatrix}
1 & -1 & 0 \\
2 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & -2 \\
0 & 0 & 1
\end{pmatrix}
\]
\[
C = \begin{pmatrix}
C_1 \\
C_2
\end{pmatrix} = \begin{pmatrix}
1 & 1 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 1
\end{pmatrix}
\]

The decentralized output feedback matrix has the structure
\[
K_D = \begin{pmatrix}
k_1 & k_3 & 0 \\
k_2 & k_4 & 0 \\
0 & 0 & k_5
\end{pmatrix}
\]

Thus
\[
k^T = (k_1 \ k_2 \ 0 \ k_3 \ k_4 \ 0 \ 0 \ 0 \ k_5)
\]
\[
k_D^T = (k_1 \ k_2 \ k_3 \ k_4 \ k_5)
\]

Since the elements 3, 6, 7, and 8 in \(k^T\) are zero, we form \(e_{iD}\) from \(e_i\) by removing these elements. The resulting matrix \(E_D\) is
\[
E_D = \begin{pmatrix}
3 & 0 & -1 & -2 & 4 \\
-1 & 0.1 & 1 & 1.1 & -7 \\
0.39 & -0.15 & -0.81 & -0.45 & 11.81 \\
-0.136 & 0.1345 & 0.501 & 0.0045 & -19.7265 \\
-0.0268 & -0.0402 & -0.0968 & 0.2788 & 32.7965
\end{pmatrix}
\]

Now \(\text{rank } E_D = 5 = n\), and therefore, pole assignment is possible by decentralized output feedback. Algorithm 5.2 is used to compute \(K_D\). The result is
\[
K_D = \begin{pmatrix}
0.91008 & 0.47852 & 0 \\
7.09889 & 2.93333 & 0 \\
0 & 0 & -0.55876
\end{pmatrix}
\]

The local control laws are
\[
u_1 = \begin{pmatrix}
0.91008 \\
7.09889
\end{pmatrix} y_1
\]
\[
u_2 = -0.55876 y_2
\]

It is interesting to note that using different initial feedback matrices, it is possible to obtain six alternative solutions for this example.

Figure 7.18 shows the step responses of the system before (open-loop) and after (closed-loop) the design of the decentralized output feedback controller. Note that the closed-loop decentralized system provides stable output responses.
Figure 7.18 Impulse responses for Example 7.6.
(a) Output No. 1. (b) Output No. 2. (c) Output No. 3.
PROBLEMS

7.1 Consider a system
\[
\dot{x} = \begin{pmatrix}
-1 & 0 & -1 \\
0 & -2 & -1 \\
1 & 0 & -5
\end{pmatrix} x + \begin{pmatrix}
1 & 0 \\
0 & 1 \\
1 & 2
\end{pmatrix} u
\]
and find an aggregated model using general aggregation. Is this aggregation dynamically exact?

7.2 Repeat Prob. 7.1 for modal aggregation.

7.3 Show that the aggregation matrix which guarantees exact aggregation is given by
\[
R = [v_1^T; v_2^T; \cdots; v_k^T]
\]
where \( v_i, \ i = 1, \ldots, k \) are the left eigenvectors of matrix \( A \), that is, \( v_i A = \lambda_i v_i \).

7.4 CAD problem. Use program MODAL of LSSPAK/PC or your favorite CAD language (package) to solve Prob. 7.2. Repeat using MATRIXx or CONTROL.lab’s matrix analysis primitives.
7.5 CAD problem. Use primitive AGGR of CONTROL.lab or your favorite CAD language to find a reduced-order model for system of Prob. 7.1. Verify your result by using primitives PINV, INV, and elementary operations "*", "/", and so on.

7.6 Show that for a stable matrix $A$, Grammian matrices $G_c$ and $G_o$ satisfy Eqs. (7.38) and (7.39), respectively.

7.7 A system is described by

$$
\dot{x} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.5 & -1 & -10 & -3 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} u 
$$

$$
y = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} x
$$

Find a reduced-order model using the balanced approach.

7.8 Use definition of $\hat{G}_o - \hat{G}_c$ and transformation matrix $S$ in Eq. (7.50) to show that $\hat{G}_c = \hat{G}_o = \Sigma$.

7.9 CAD problem. Use CONTROL.lab's primitives or your own CACSD package other than BALN (or MATRIXx's BALANCE) to find a balancing reduced-order model for the seventh-order system of CAD Example 7.2.

7.10 Let the following system be dependent on two parameters,

$$
\dot{x} = \begin{pmatrix} -1 & 0 & a \\ 0 & 0 & -1 \\ b & 0 & -1 \end{pmatrix} x
$$

For what range on $a$ and $b$ is this system weakly coupled?

Hint: Find $r$ and $R$ the minimum and maximum values of $\lambda\{A\}$ and check the ratio $r/R << 1$. Let $\varepsilon_{12}$ and $\varepsilon_{21}$ be the maximum of moduli of $A_{12}$ and $A_{21}$ submatrices in

$$
A = \begin{bmatrix} A_1 & A_{12} \\ A_{21} & A_2 \end{bmatrix}
$$

and check for $\varepsilon_{12}\varepsilon_{21}/n_1/R^2$ ratio as well, where $\dim(A_{1}) = n_1 \times n_1$.

7.11 The balanced method of Sec. 7.2.3 can be extended to unstable linear time-invariant systems if one notes that their balancing Lyapunov equations would be, (Santiago and Jamshidi 1986),

$$
G_o(A - kI) + (A - kI)^T G_o + C^T C = 0
$$

$$
G_c(A - kI)^T + (A - kI)G_c + BB^T = 0
$$

where $k$ is greater than the real part of the positive-most eigenvalue of the $A$ matrix. Use this concept to devise a balancing algorithm for unstable systems.

7.12 Use approach of Prob. 7.11 to reduce the following unstable system using the balancing method.

$$(A, B, C) = \left( \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -60 & -8 & 9 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 50 & 15 & 3 \end{bmatrix} \right)$$
7.13 For the system

\[
\dot{x} = \begin{pmatrix} -1 & 0 & 1 \\ 1 & -2 & 0 \\ 0 & -1 & -10 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u \\
\begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} x
\]

find a reduced-order model using Algorithm 7.1.

7.14 Repeat Prob. 7.7 for the following unstable system,

\[
\dot{x} = \begin{pmatrix} -1 & 0 & 1 \\ 0 & -2 & 1 \\ 0 & 0 & 1 \end{pmatrix} x + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} u \\
y = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 2 & 1 \end{pmatrix} x
\]

7.15 CAD problem. Consider a two-subsystem problem

\[
\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{pmatrix} = \begin{pmatrix} -1.2 & 0 & 0 & 0.1 \\ -0.5 & -2 & -2 & 0.2 \\ 0.5 & 0.5 & -2.0 & 0 \\ 0 & 0 & -0.5 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + \begin{pmatrix} 0.2 \\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0.2 \\ 0.2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \\
x(0) = (-1 \ 0 \ -1 \ 0)^T
\]

Use the goal coordination algorithm to find a hierarchical controller using a quadratic cost function with \( Q_i = I_2, R_i = 1, i = 1, 2, \Delta t = 0.1 \) and \( t_f = 2 \). Use your favorite code.

7.16. CAD problem. Repeat Prob. 7.13 using the interaction prediction algorithm. You may use INTPRD of LSSPAK/PC or your favorite code.

7.17. CAD problem. Determine the fixed modes of the following system:

\[
\dot{x} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} x + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u \\
y = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} x
\]

Is the system stabilizable under decentralized control? Verify your conclusion using CONTROL.lab or MATRIXx. Now change \( a(2, 2) = 1 \) and \( b(2, 2) = 0 \) and repeat.

7.18. Find a decentralized stabilizing output controller for the system.

\[
\dot{x} = \begin{pmatrix} 0 & 1 \\ 2 & -1 \end{pmatrix} x + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u \\
y = \begin{pmatrix} 1 & 0 \\ 1 & 2 \end{pmatrix} x
\]

Place the poles at \(-2.5 \pm j 0.866\). Verify your results using PC-MATLAB.
Appendix A

Review of Linear Algebra

A.1 INTRODUCTION

In this appendix, we briefly discuss several basic concepts of linear algebra and matrix theory that are frequently used in various parts of this text. These include spaces, linear independence, inner products and norms, eigenvalues and eigenvectors, functions of a matrix, and quadratic forms. A number of exercises are provided at the end of the appendix.
A.2 LINEAR SPACES

In this section, definitions of fields, vectors, vector space, linear dependence and associated terminologies are presented.

A set $F$ of numbers containing two or more numbers is a number field if for every pair of members $\alpha$ and $\beta$ in $F$, the numbers $(\alpha + \beta)$, $(\alpha - \beta)$, $\alpha \beta$ and $\alpha/\beta$ ($\beta \neq 0$) are also members of $F$. For example the set of all real numbers constitutes a field, but the set of positive numbers is not a field. The notion of a field can be generalized to mathematical objects other than numbers, e.g. matrices (Chen, 1984).

Let $F$ denote the real or the complex number field. An $n$-dimensional vector $x$, over a field is an ordered set of $n$ numbers which belong to $F$. It is denoted by

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

The numbers $x_1$, $x_2$, $\ldots$, $x_n$ are called elements or components of $x$. The vector $x$ is called an $n \times 1$ vector or a column vector. The transpose of $x$ is a $1 \times n$ vector or a row vector and is denoted by

$$x^T = (x_1, x_2, \ldots, x_n)$$

The null vector denoted by $0$ is a vector whose elements are all zero.

A vector space $V$ over a field $F$ is defined as the set of all vectors over $F$ for which two operations called vector addition and scalar multiplication are defined and satisfy the following conditions:

1. To every pair of vectors $x$ and $y$ in $V$ the sum of $x$ and $y$, denoted by $x + y$, belongs to $V$. Furthermore, addition of vectors is commutative and associative, that is, $x + y = y + x$ and $(x + y) + z = x + (y + z)$.

2. To every pair of vectors $x$ and $y$ in $V$, there exists a unique vector $z$ in $V$ such that $x + z = y$. In particular, there exists a vector $0$ such that $x + 0 = x$ for all $x$ in $V$. In addition, to every vector $x$ in $V$, there exists a unique vector $-x$ such that $x + (-x) = 0$.

3. For each vector $x$ in $V$ and each scalar $\alpha$ in $F$ the product $\alpha x$ belongs to $V$. Furthermore, multiplication by scalars is associative and distributive, that is, $(\alpha_1 \alpha_2)x = \alpha_1 (\alpha_2 x)$, $\alpha(x + y) = \alpha x + \alpha y$, $(\alpha_1 + \alpha_2)x = \alpha_1 x + \alpha_2 x$.

4. For every $x$ in $V$, $1x = x$, where $1$ is the element $1$ in $F$.

If $F$ is the field of real (complex) numbers, $V$ is called a real (complex) vector space. A subset $W$ of vectors in $V$ is called a subspace of $V$ if for any pair of vectors $x$ and $y$ in $W$ and any scalar $\alpha$ both $x + y$ and $\alpha x$ also belong to $W$. Note that a subspace always contains $0x = 0$ and that a subspace of a vector space is itself a vector space.
As an example, consider the two dimensional real vector space $\mathbb{V}$ denoted by
\[
\begin{pmatrix}
x \\
y
\end{pmatrix}
\]. Then, every straight line passing through the origin is a subspace of $\mathbb{V}$, that is, \[
\begin{pmatrix}
x \\
\alpha x
\end{pmatrix}
\] for any fixed real $\alpha$ is a subspace of \[
\begin{pmatrix}
x \\
y
\end{pmatrix}
\].

A.3 LINEAR INDEPENDENCE AND BASIS

The $n$ vectors $x_1, x_2, \ldots, x_n$ are said to be linearly independent if
\[
\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = 0
\]
implies that $\alpha_1 = \alpha_2 = \cdots = \alpha_n = 0$, where $\alpha_1, \alpha_2, \ldots, \alpha_n$ are constants. On the other hand, $x_1, x_2, \ldots, x_n$ are said to be linearly dependent if, and only if, there exists a vector $x_i$ that can be expressed as a linear combination of other vectors $x_j, j = 1, 2, \ldots, n; i \neq j$.

The maximum number of linearly independent vectors in a linear space $\mathbb{V}$ is called the dimension of the linear space $\mathbb{V}$. In an $n$-dimensional real vector space there are, at most, $n$ linearly independent vectors over the field of real numbers.

A set of vectors $b_1, b_2, \ldots, b_n$ of a vector space $\mathbb{V}$ is called the basis of $\mathbb{V}$ if these $n$ vectors are linearly independent and every vector $x$ in $\mathbb{V}$ can be uniquely expressed as a linear combination of the vectors $b_1, b_2, \ldots, b_n$, that is,
\[
x = \sum_{i=1}^{n} \alpha_i b_i
\]
where $\alpha_i$ are the coordinates of the vector $x$ relative to the basis $b_1, b_2, \ldots, b_n$.

For a given vector space $\mathbb{V}$, a basis is not unique. In an $n$-dimensional vector space, any set of $n$ linearly independent vectors is a basis. For example in a three dimensional vector space, the set of vectors \[
\begin{pmatrix}
1 \\
0 \\
0
\end{pmatrix}, \begin{pmatrix}
0 \\
1 \\
0
\end{pmatrix}, \begin{pmatrix}
0 \\
0 \\
1
\end{pmatrix}
\]
or the set of vectors \[
\begin{pmatrix}
1 \\
1 \\
0
\end{pmatrix}, \begin{pmatrix}
0 \\
1 \\
1
\end{pmatrix}, \begin{pmatrix}
1 \\
0 \\
1
\end{pmatrix}
\],

\[
\begin{pmatrix}
1 \\
0 \\
1
\end{pmatrix}
\] form a basis.

A.4 INNER PRODUCT AND NORMS

Consider a pair of vectors $x$ and $y$ in a vector space $\mathbb{V}$. Then, any rule that assigns a scalar quantity $(x, y)$ to the pair is called an inner (scalar) product if the following conditions are satisfied:
1. \((x, y) = (\overline{y}, \overline{x})\), where the bar denotes the conjugate of a complex number
2. \((\alpha x, y) = \overline{\alpha}(x, y) = (x, \overline{\alpha}y)\), where \(\alpha\) is a complex number
3. \((x + w, y + z) = (x, y) + (x, z) + (w, y) + (w, z)\), where \(w\) and \(z\) are vectors in \(V\)
4. \((x, x) > 0\) for \(x \neq 0\)

There are many different definitions of inner products which satisfying these conditions. A commonly used definition of the inner product of a pair of \(n\)-dimensional vectors \(x\) and \(y\) is

\[(x, y) = \bar{x}_1 y_1 + \bar{x}_2 y_2 + \cdots + \bar{x}_n y_n = x^* y\]

where \(\bar{x}_i\) is the conjugate of the element \(x_i\), and \(x^*\) is conjugate transpose of the vector \(x\). When \(x\) and \(y\) are real, we have

\[(x, y) = x^T y\]

Note that

\[x^* x = \sum_{i=1}^{n} \bar{x}_i x_i = \sum_{i=1}^{n} |x_i|^2\]

is a nonnegative scalar and that

\[xx^* = \begin{pmatrix} x_1 \bar{x}_1 & \cdots & x_1 \bar{x}_n \\ \vdots & \ddots & \vdots \\ x_n \bar{x}_1 & \cdots & x_n \bar{x}_n \end{pmatrix}\]

is an \(n \times n\) matrix. It is easy to verify that for an \(n \times n\) matrix \(A\) and an \(n \times 1\) vector \(x\), the inner product of \(Ax\) and \(y\) has the property that \((Ax, y) = (x, A^*y)\).

Similarly, when \(A\) is a real \(n \times n\) matrix and \(x\) is a real \(n \times 1\) vector, we have \((Ax, y) = (x, A^T y)\), where \(A^T\) is the transpose of \(A\). Note that \(Ax\) is a linear transformation that transforms the vector \(x\) in a vector space into the vector \(Ax\) in another vector space. If \(A\) is a unitary matrix, that is, \(A^{-1} = A^*\), then the inner product \((y, y)\) is invariant under the linear transformation \(y = Ax\), since

\[(y, y) = (Ax, Ax) = (x, A^* A x) = (x, A^{-1} A x) = (x, x)\]

Such a transformation is called a **unitary transformation**. Similarly when \(A\) is an orthogonal matrix, that is, \(A^{-1} = A^T\), the inner product \((y, y)\) is invariant under the linear transformation \(y = Ax\). Such a transformation is called **orthogonal**.

Finally when the inner product of two vectors \(x\) and \(y\) is zero, that is, \((x, y) = 0\), then \(x\) and \(y\) are said to be orthogonal.

We can now use the inner products to define norms of a vector. A norm is a function that assigns to every vector \(x\) a real number denoted by \(\|x\|\) such that

1. \(\|x\| > 0\) for \(x \neq 0\) and \(\|x\| = 0\) for \(x = 0\)
2. \(\|\alpha x\| = |\alpha| \|x\|\) where \(\alpha\) is a scalar and \(|\alpha|\) is the absolute value of \(\alpha\)
3. \( \| x + y \| \leq \| x \| + \| y \| \) for all \( x \) and \( y \)
4. \( \| (x, y) \| \leq \| x \| \| y \| \) This is called Schwarz inequality.

Several definitions are available. A commonly used definition is

\[
\| x \| = (x, x)^{1/2} = [(x_1^2 + x_2^2 + \cdots + x_n^2)^{1/2}
\]

When \( x \) is a real vector, the quantity \( \| x \| \) is the length of the vector.

Among other used definitions of norms are \( \| x \| = \sum_{i=1}^{n} |x_i| \) and \( \| x \| = \max \{|x_i|\} \). Verify that all these norms satisfy the above conditions.

### A.5 Eigenvalues and Eigenvectors

Let \( A \) be an \( n \times n \) matrix with elements in the field of real numbers. Then a scalar \( \lambda \) in the field of complex numbers is called an eigenvalue of \( A \) if there exists a nonzero vector \( x \) such that

\[
Ax = \lambda x
\]  \hspace{1cm} (A.1)

Any nonzero vector \( x \) satisfying Eq. (A.1) is called an eigenvector of \( A \) associated with the eigenvalue \( \lambda \).

In order to determine the eigenvalues of \( A \), write (A.1) as

\[
(\lambda I - A)x = 0
\]  \hspace{1cm} (A.2)

where \( I \) is an identity matrix and \( (\lambda I - A) \) is an \( n \times n \) matrix. For Eq. (A.2) to have a nontrivial solution for \( x \) (i.e., \( x \neq 0 \)), the \( n \times n \) matrix \( (\lambda I - A) \) must be rank deficient which implies that

\[
p(\lambda) = \det (\lambda I - A) = 0
\]  \hspace{1cm} (A.3)

where \( p(\lambda) \) is a polynomial of degree \( n \) in \( \lambda \) and is called the characteristic polynomial of \( A \). Since \( p(\lambda) \) is of degree \( n \), the \( n \times n \) matrix \( A \) has \( n \) eigenvalues.

The eigenvalues can be real or complex, and distinct or repeated. When the eigenvalues are distinct, we can state the following result.

#### Theorem A.1

Let \( \lambda_i, i = 1, 2, \ldots, n \) be the distinct eigenvalues of \( A \) and let \( x_i \) be the eigenvectors associated with the eigenvalues \( \lambda_i \). Then the set of vectors \( x_1, x_2, \ldots, x_n \) are linearly independent.

#### Proof

The proof is by contradiction. Suppose that \( x_1, x_2, \ldots, x_n \) are linearly dependent. Then, there exists \( \alpha_1, \alpha_2, \ldots, \alpha_n \), not all zero, such that

\[
\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = 0
\]  \hspace{1cm} (A.4)

Assuming \( \alpha_1 \neq 0 \), Eq. (A.4) implies that
(\lambda_2 I - A)(\lambda_3 I - A) \cdots (\lambda_n I - A) \left( \sum_{i=1}^{n} \alpha_i x_i \right) = 0 \quad (A.5)

Now

(\lambda_j I - A)x_i = (\lambda_j - \lambda_i)x_i \quad \text{for } i \neq j

and

(\lambda_i I - A)x_i = 0

Thus Eq. (A.5) can be written as

\alpha_1(\lambda_2 - \lambda_1)(\lambda_3 - \lambda_1) \cdots (\lambda_n - \lambda_1)x_1 = 0 \quad (A.6)

Since \lambda_i, i = 1, 2, \ldots, n are distinct by assumption, Eq. (A.6) implies that \alpha_1 = 0, which is a contradiction. Thus, the set of vectors \( x_1, x_2, \ldots, x_n \) is linearly independent.

For illustration, consider \( A = \begin{pmatrix} 1 & -1 \\ 2 & -1 \end{pmatrix} \) whose characteristic equation is

\( p(\lambda) = \lambda^2 + 1 = 0 \) and hence \( \lambda_1 = j, \lambda_2 = -j \). The eigenvector \( x_1 \) associated with the eigenvalue \( \lambda_1 \) can be obtained from:

\[(\lambda_1 I - A)x_1 = \begin{pmatrix} j - 1 \\ -2 \\ j + 1 \end{pmatrix} x_1 = 0\]

Hence \( x_1 = \begin{pmatrix} 1 \\ -j \end{pmatrix} \) is a solution. Similarly, it is easy to verify that \( x_2 = \begin{pmatrix} 1 \\ 1 - j \end{pmatrix} \) is an eigenvector associated with the eigenvalue \( \lambda_2 = -j \). Note that \( x_1 \) and \( x_2 \) are linearly independent since

\[\det(x_1 \ x_2) = \det\left(\begin{pmatrix} 1 \\ -j \end{pmatrix}, \ \begin{pmatrix} 1 \\ 1 - j \end{pmatrix}\right) = 2j \neq 0.\]

Let us now consider the case of repeated eigenvalues. Assume that an eigenvalue of the \( n \times n \) matrix \( A \), say \( \lambda_1 \), is repeated and the remaining eigenvalues are distinct. In this case, only one eigenvector can be found for the repeated eigenvalues \( \lambda_1 \) using \( (\lambda_1 I - A) x_1 = 0 \), and the total number of eigenvectors will be less than \( n \). In order to circumvent this situation, we define a generalized eigenvector of rank \( k \) associated with eigenvalue \( \lambda_1 \) if a vector \( y \) and a number \( k > 1 \) can be found such that

\[(\lambda_1 I - A)^k y = 0\]
and

\[(\lambda I - A)^{k-1} y \neq 0\]

then the set of vectors \(x_k = y, x_{k-1} = (\lambda I - A)x_k, \ldots, x_1 = (\lambda I - A)x_2\)
are called the generalized eigenvectors associated with the eigenvalue \(\lambda_1\).

It is easy to show that the generalized eigenvectors \(x_1, x_2, \ldots, x_k\) are linearly
independent. Furthermore, when the \(n \times n\) matrix \(A\) has several repeated eigenvalues,
the total number of simple and generalized eigenvectors is \(n\) and form a set of \(n\)
linearly independent vectors.

### A.6 FUNCTIONS OF A MATRIX

In this section, we shall study functions of a square matrix. To this end, we define
power of the \(n \times n\) matrix \(A\) as

\[A^k = A A \cdots A\quad (k \text{ terms})\]

\[A^0 = I\]

where \(k\) is a positive integer. The following result, due to Cayley and Hamilton, is
very important in system theory.

**Cayley-Hamilton Theorem.** Let \(A\) be an \(n \times n\) matrix with the characteristic
equation

\[\det(\lambda I - A) = \lambda^n + p_1 \lambda^{n-1} + \cdots + p_{n-1} \lambda + p_n = 0\quad (A.7)\]

Then the matrix \(A\) satisfies its characteristic equation, that is,

\[A^n + p_1 A^{n-1} + \cdots + p_{n-1} A + p_n I = 0\quad (A.8)\]

**Proof.** The \(\text{adj} (\lambda I - A)\) is a polynomial in \(\lambda\) of degree \(n - 1\) (Reader can
show this as an exercise). Hence it can be expressed as

\[\text{adj} (\lambda I - A) = B_1 \lambda^{n-1} + \cdots + B_{n-1} \lambda + B_n\quad (A.9)\]

where \(B_1, \ldots, B_n\) are constant \(n \times n\) matrices with \(B_1 = I\).

Furthermore, \((\lambda I - A)\) and \(\text{adj} (\lambda I - A)\) are commutative, and

\[(\lambda I - A) \text{adj} (\lambda I - A) = \text{adj} (\lambda I - A) (\lambda I - A) = \det(A - \lambda I)\]

Thus, using Eqs. (A.7) and (A.9) we obtain

\[\det (\lambda I - A) I = (I \lambda^n + p_1 I \lambda^{n-1} + \cdots + p_n I)\]

\[= (\lambda I - A) \text{adj} (\lambda I - A)\]

\[= \text{adj} (\lambda I - A)(\lambda I - A)\]
Now \((\lambda I - A)\) and \(\text{adj} (\lambda I - A)\) commute, and their product becomes zero if \(\lambda\) is replaced by \(A\) which implies
\[
A^n + p_1 A^{n-1} + \cdots + p_n I = 0.
\]

The characteristic polynomial is not necessarily the least degree polynomial that \(A\) satisfies. For the \(n \times n\) matrix \(A\), a polynomial of degree \(m < n\) may exist such that
\[
A^m + q_1 A^{m-1} + \cdots + q_m I = 0
\]
where \(q_1, \ldots, q_m\) are constant scalars. A polynomial with least degree having \(A\) as its root is called the \textit{minimal polynomial}. It can be shown that the minimal polynomial \(\psi(\lambda)\) is
\[
\psi(\lambda) = \frac{\det(\lambda I - A)}{d(\lambda)} \tag{A.10}
\]
where \(d(\lambda)\) is the greatest common divisor of all the elements of \(\text{adj}(\lambda I - A)\).

Let us demonstrate these concepts by an example. Consider the matrix
\[
A = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 3 & 1 \end{pmatrix}
\]
The characteristic polynomial is
\[
p(\lambda) = \det(\lambda I - A) = (\lambda - 2)^2(\lambda - 1) = \lambda^3 - 5\lambda^2 + 8\lambda - 4
\]
According to the Cayley-Hamilton theorem, we have
\[
A^3 - 5A^2 + 8A - 4I = 0
\]
which can readily be verified. In order to find the minimal polynomial, we form
\[
\text{adj} (\lambda I - A) = \begin{pmatrix} (\lambda - 2)(\lambda - 1) & 0 & 0 \\ 0 & (\lambda - 2)(\lambda - 1) & 0 \\ 0 & -3(\lambda - 2) & (\lambda - 2)^2 \end{pmatrix}
\]
The greatest common divisor of all the elements of \(\text{adj} (\lambda I - A)\) is \((\lambda - 2)\) and hence the minimal polynomial is
\[
\psi(\lambda) = \frac{p(\lambda)}{\lambda - 2} = (\lambda - 2)(\lambda - 1) = \lambda^2 - 3\lambda + 2
\]
It is easy to verify that
\[
A^2 - 3A + 2I = 0
\]
Thus, the minimal polynomial is of degree \(m = 2\) which is less than \(n = 3\).
A.7 QUADRATIC FORMS

Consider a real symmetric \( n \times n \) matrix \( A \), (i.e., a matrix with real elements such that \( a_{ij} = a_{ji} \quad i, j = 1, 2, \ldots, n \)) and a real \( n \times 1 \) vector \( x \). Then

\[
x^T A x = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j
\]

is called a quadratic form in \( x \). Note that \( x^T A x \) is a real scalar quantity, and can be positive, zero, or negative. Depending on the sign of \( x^T A x \) the following definitions are made.

The quadratic form \( x^T A x \) is said to be positive definite if

\[
x^T A x > 0 \quad \text{for} \ x \neq 0 \quad \text{and} \quad x^T A x = 0 \quad \text{for} \ x = 0
\]

It is said to be positive semidefinite if

\[
x^T A x \geq 0 \quad \text{for} \ x \neq 0 \quad \text{and} \quad x^T A x = 0 \quad \text{for} \ x = 0
\]

Similarly \( x^T A x \) is said to be negative definite if

\[
x^T A x < 0 \quad \text{for} \ x \neq 0 \quad \text{and} \quad x^T A x = 0 \quad \text{for} \ x = 0
\]

It is said to be negative semidefinite if

\[
x^T A x \leq 0 \quad \text{for} \ x \neq 0 \quad \text{and} \quad x^T A x = 0 \quad \text{for} \ x = 0
\]

If \( x^T A x \) can take both positive and negative signs, it is said to be indefinite.

Sylvester criterion, given below, provides a test for determining the sign of \( x^T A x \) from the matrix \( A \).

**Sylvester Theorem.** A necessary and sufficient condition for positive definiteness of the quadratic form \( x^T A x \) is that all the principal minors of \( A \) be positive, namely

\[
\det (A_k) \equiv \det \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \cdots & a_{kk} \end{pmatrix} > 0 \quad \text{for} \ k = 1, 2, \ldots, n \quad (A.11)
\]

**Proof.** Consider the sufficiency part and assume that \( x^T A x \) is positive definite. Let us substitute \( x_{m+1} = x_{m+2} = \cdots = x_n = 0 \) in the quadratic form

\[
x^T A x = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j \quad \text{(A.12)}
\]

then

\[
x^T A x = \sum_{r=1}^{m} \sum_{s=1}^{n} a_{rs} x_r x_s \quad \text{(A.13)}
\]

is positive definite. The coefficient matrix is now given by
Since the quadratic form Eq. (A.13) is positive definite, we must have \( \det(A_m) > 0 \). Hence, we conclude that if Eq. (A.12) is positive definite, then all principle minors of \( A \), namely \( \det(A_k) \), \( k = 1, 2, \ldots, n \), must be positive.

For the necessity part, we must show that if \( \det(A_k) > 0 \) for \( k = 1, 2, \ldots, n \), then \( x^T A x \) is positive definite. This is accomplished by induction, namely, assume that the result holds for the \((k - 1) \times (k - 1)\) matrix \( A_{k-1} \) and express \( A_k \) in terms of \( A_{k-1} \). This gives

\[
A_k = \begin{pmatrix} A_{k-1} & b \\ b^T & a_{kk} \end{pmatrix}
\]

where

\[
b = \begin{pmatrix} a_k \\ a_{2k} \\ \vdots \\ a_{k-1,k} \end{pmatrix}
\]

Then show that \( x^T A_k x \) is positive definite. Details are left as an exercise.

It can be shown that a necessary and sufficient condition for the quadratic form \( x^T A x \) to be positive semidefinite is that \( \det A_k \geq 0 \) for \( k = 1, 2, \ldots, n - 1 \) and \( \det A = 0 \), where \( A_k \) is defined previously.

For illustration, consider the quadratic form

\[
Q = 2 x_1^2 + 2 x_2^2 + ax_3^2 + x_1 x_2 + x_1 x_3 + 3 x_2 x_3
\]

where \( a \) is a scalar. Determine a range(s) for a positive definiteness and positive semidefiniteness of \( Q \). We have

\[
2 > 0, \quad \det \begin{pmatrix} 2 & 0 \\ 1 & 3 \end{pmatrix} > 0, \quad \det \begin{pmatrix} 2 & 0 & 0 \\ 1 & 3 & 2 \\ 1 & 1 & a \end{pmatrix} = 2(3a - 2)
\]

Thus \( Q \) is positive definite for \( a > \frac{2}{3} \), and is positive semidefinite for \( a = \frac{2}{3} \).

### A.8 DETERMINANT AND INVERSE FORMULAS

In this section, we state a number of identities for determinant and inverse of a matrix. We recall that the determinant of the \( n \times n \) matrix \( A \) is
\[ \det A = \sum_{j} a_{ij} c_{ij}, \quad i = 1, \ldots, r \]  

(A.14)

where \( a_{ij} \) is a typical element of \( A \) and \( c_{ij} \) is the cofactor corresponding to \( a_{ij} \) and is given by \( c_{ij} = (-1)^{i+j} \det M_{ij} \), where \( M_{ij} \) is the \((n - 1) \times (n - 1)\) matrix obtained by deleting the \( i \)th row and \( j \)th column of \( A \). The scalar quantity \( \det M_{ij} \) is called the \( ij \)th minor of \( A \).

Using Eq. (A.14), several results can be readily obtained as follows:

- If the matrix \( \hat{A} \) is obtained by interchanging any two rows (or columns) of \( A \), then \( \det \hat{A} = -\det A \).
- If any row (or column) of \( A \) is multiplied by a scalar \( \alpha \), the resulting matrix has the property that \( \det \hat{A} = \alpha \det A \). Furthermore, it is easy to verify that \( \det (\alpha A) = \alpha^n \det A \).
- If \( \hat{A} \) is obtained from \( A \) by adding a multiple of any one row (or column) to another, then \( \det \hat{A} = \det A \).

These operations performed on \( A \) to obtain \( \hat{A} \) are called elementary operations.

The rank of a matrix is the number of linearly independent rows (or columns) of the matrix. In the \( n \times n \) matrix \( A \), \( \det(A) = 0 \) if there are less than \( n \) independent rows or columns in \( A \). (See Probs. A.29 and A.30 for formulas concerning rank of product and sum of two matrices \( A \) and \( B \).)

Now consider the case where two matrices \( A \) and \( B \) of dimensions \( n \times n \) and \( m \times p \) are given. Let \( |A|_{i,j} \) be the \( r \times r \) minor formed by selecting \( r \) rows, say \( i_1, \ldots, i_r \), and \( r \) columns, say \( j_1, \ldots, j_r \), of \( A \), where \( r \leq \min(n, m) \). Similarly define \( |B|_{i,j} \) as the \( r \times r \) minors formed by selecting appropriate rows and columns of \( B \). Then, the Binet-Cauchy formula states that

\[ |AB|_{i,j} = \sum_k |A|_{i,k} |B|_{k,j} \]  

(A.15)

where \( k \) ranges over all integers \( r, 1 \leq r \leq \min(n, m, p) \). Equation (A.15) can be used to show that when \( m = n = p \), that is, \( A \) and \( B \) are both square matrices, we have \( \det(AB) = \det(BA) \). As an exercise, use Binet-Cauchy formula to determine \( \det(AB) \) where

\[
A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}
\]

In some cases, the elements of a matrix are themselves matrices. The Binet-Cauchy formula can be used to find the determinant of such block matrices. For example, the determinant of \( \begin{pmatrix} A & 0 \\ C & B \end{pmatrix} \) is evaluated in terms of determinants of the square matrices \( A \) and \( B \) as follows:
\[
\begin{pmatrix}
A & 0 \\
C & B
\end{pmatrix} = 
\begin{pmatrix}
A & 0 \\
0 & I
\end{pmatrix} 
\begin{pmatrix}
I & 0 \\
0 & B^{-1}C
\end{pmatrix} 
\begin{pmatrix}
I & 0 \\
B^{-1}C & I
\end{pmatrix}
\]

Hence

\[
\det \begin{pmatrix}
A & 0 \\
C & B
\end{pmatrix} = \det A \det B
\]

where it is assumed that \(B^{-1}\) exists.

Similarly, it can be shown that when \(A\) is nonsingular,

\[
\det \begin{pmatrix}
A & B \\
C & D
\end{pmatrix} = \det A \det (D - CA^{-1}B)
\]

This is called Schur determinant identity. As an exercise, derive a formula similar to Eq. (A.16) for the case where \(D\) is nonsingular.

Another useful determinant identity which can be obtained using Binet-Cauchy formula is

\[
\det (I_n + AB) = \det (I_m + BA)
\]

where \(A\) and \(B\) are \(n \times m\) and \(m \times m\) matrices, respectively. An interesting special case of Eq. (A.17) is when \(m = 1\) in which case \(A\) is the \(n \times 1\) column vector \(a\) and \(B\) is the \(1 \times n\) row vector \(b\), and Eq. (A.17) reduces to

\[
\det (I_n + ab) = 1 + ba
\]

Equation (A.18) and the definition of inverse of a matrix can be used to show that for the nonsingular \(n \times n\) matrix \(C\), and the column and row vectors \(a\) and \(b\), we can write

\[
(C + ab)^{-1} = C^{-1} - \frac{(C^{-1}a)(b C^{-1})}{1 + b C^{-1}a}
\]

Identity Eq. (A.19) is useful when \(C^{-1}\) is known and the inverse \((C + ab)^{-1}\) is to be computed. A generalization of Eq. (A.19) is

\[
(A + BCD)^{-1} = A^{-1} - A^{-1}B (DA^{-1}B + C^{-1})^{-1} DA^{-1}
\]

where \(A, B, C,\) and \(D\) are \(n \times n, n \times m, m \times m,\) and \(m \times n\) matrices, respectively. Identities Eqs. (A.17) and (A.20) have many useful applications in feedback control systems. For example, the dynamical system \(\dot{x} = Ax + Bu, y = Cx\) with the feedback control low \(u = v - Ky\), where \(v\) is the command input and \(K\) is the feedback matrix, has a closed-loop transfer function matrix \([I + C(sl - A)^{-1}B K]^{-1}\).

\([C(sl - A)^{-1}B]\) and a closed-loop characteristic polynomial \(\det (sl - A) \det[I + C (sl - A)^{-1}B K]\).

The inverse of a block matrix can be expressed in terms of the submatrices as

\[
\begin{pmatrix}
A & B \\
C & D
\end{pmatrix}^{-1} = 
\begin{pmatrix}
A^{-1} + EG^{-1}F & -EG^{-1} \\
-G^{-1}F & G^{-1}
\end{pmatrix}
\]

(A.21)
where \( G = (D - CA^{-1}B) \), \( E = A^{-1}B \), and \( F = CA^{-1} \), assuming that \( A^{-1} \) exists.

This identity can be verified by multiplying both sides of Eq. (A.21) by \( \begin{pmatrix} A & B \\ C & D \end{pmatrix} \)
and equating the corresponding expressions. The matrix \( G \) is known as the Schur complement of \( A \).

**A.9 SINGULAR VALUE DECOMPOSITION**

A matrix can be decomposed into several useful forms. A very important decomposition is the so-called *singular value decomposition* which is used extensively for matrix computation. Recall from Sect. A.4 that an orthogonal matrix \( U \) is one that has the property that \( U^T U = I \). We now state the following theorem.

**Singular Value Decomposition Theorem.** Let \( A \) be an \( n \times m \) matrix, then there exist an orthogonal \( n \times n \) matrix \( U = (u_1, \ldots, u_n) \) and an orthogonal \( m \times m \) matrix \( V = (v_1, \ldots, v_m) \), where \( u_1, \ldots, u_n \) and \( v_1, \ldots, v_m \) are column vectors, such that

\[
U^T A V = \text{diag} (\sigma_1, \ldots, \sigma_p)
\]

(A.22)

where \( p = \min (n, m) \) and \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0 \).

**Proof.** Let \( x \) and \( y \) be \( m \times 1 \) and \( n \times 1 \) vectors, respectively, such that \( \|x\| = \|y\| = 1 \) and \( Ax = \sigma y \) with \( \sigma = \|A\| \), where \( \|x\| = (x_1^2 + \cdots + x_m^2)^{1/2} \), \( \|y\| = (y_1^2 + \cdots + y_n^2)^{1/2} \) and \( \|A\| = \left( \sum_{i=1}^n \sum_{j=1}^m a_{ij} \right)^{1/2} \). Let \( V = (x, V_1) \) and \( U = (y, U_1) \), where \( V_1 \) and \( U_1 \) are \( m \times (m - 1) \) and \( n \times (n - 1) \) matrices, respectively. It follows that

\[
U^T A V = \begin{pmatrix} \sigma & b^T \\ 0 & B \end{pmatrix} \equiv A_1
\]

where \( b \) is a vector and \( B \) is a matrix of appropriate dimension. Since

\[
\left\| A_1 \begin{pmatrix} \sigma \\ b \end{pmatrix} \right\|^2 \geq (\sigma^2 + b^T b)^2
\]

it follows that \( \|A_1\|^2 \geq (\sigma^2 + b^T b)^2 \). Now since \( \sigma^2 = \|A\|^2 \), we must have \( b = 0 \). The proof then follows by induction.

The scalars \( \sigma_i, i = 1, 2, \ldots, p \) are called the singular values of \( A \), and the vectors \( u_i \) and \( v_i \) are called the \( i \)-th left and \( i \)-th right singular vectors, respectively. Now consider (A-22) and premultiply both sides by \( U \) to obtain
where we have substituted $U^T U = I$. Similarly postmultiplying both sides of (A.22) by $V^T$ and noting that $VV^T = I$, we have

$$A^T u_i = \sigma_i v_i$$  \hspace{1cm} (A.24)

Substituting $v_i$ from (A.24) into (A.23), we obtain

$$(AA^T) u_i = \sigma_i^2 u_i$$  \hspace{1cm} (A.25)

It is seen that $\sigma_i^2$ and $u_i$ are, respectively, the eigenvalues and eigenvectors of the matrix $AA^T$. Similarly, (A.23) and (A.24) yield

$$(A^T A) v_i = \sigma_i^2 v_i$$  \hspace{1cm} (A.26)

Thus $\sigma_i^2$ and $v_i$ are the eigenvalues and eigenvectors of $A^T A$.

It is to be noted that one or more singular values of $A$ can be zero. In this case we can write

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > \sigma_{r+1} = \cdots = \sigma_p = 0$$  \hspace{1cm} (A.27)

and we conclude from (A.22) that rank $A = r$.

Singular values are very useful for determining the rank of a matrix in situations where a near rank deficiency occurs. In such situations, numerical difficulties due to rounding errors can make the rank determination a very difficult task, if computations are not based on singular value decomposition.

**PROBLEMS**

A.1 Which of the following sets constitute a field:

a) the set of integers  

b) the set of all $3 \times 3$ matrices  

c) the set of odd numbers

A.2 Which of the following sets are linearly independent in the field of real numbers?

\begin{align*}
\text{a.} & \begin{pmatrix} 2 \\ 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 3 \end{pmatrix} \\
\text{b.} & \begin{pmatrix} 1 \\ 2 \\ a \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \\ a \end{pmatrix}, \begin{pmatrix} -2 \\ 2 \\ 1 \end{pmatrix}
\end{align*}

where $a$ is a constant.

A.3 Show that the set of all $n \times n$ matrices with real elements forms a linear space over the field of real numbers with dimension $n^2$.

A.4 Prove that $(\text{adj } A) (\text{adj } B) = \text{adj } (BA)$, where $A$ and $B$ are $n \times n$ matrices.

A.5 Prove that for the $n \times n$ matrices $A$ and $B$
\[
\det \begin{pmatrix} A & B \\ B & A \end{pmatrix} = \det (A + B) \det(A - B)
\]

**A.6** Show that for \( n \times n \) matrices \( A \) and \( B \)
\[
\text{tr} (AB) = \text{tr} (BA)
\]

**A.7** Let the eigenvalues of \( A \) be denoted by \( \lambda_1, \lambda_2, \ldots, \lambda_n \). Show that
\[
\lambda_1 \lambda_2 \cdots \lambda_n = \det (A)
\]
and that
\[
\lambda_1 + \lambda_2 + \cdots + \lambda_n = \text{tr} A
\]

**A.8** Show that every eigenvalue \( \lambda_i, i = 1, 2, \ldots, n \), of the orthogonal matrix \( A \) has magnitude equal to unity.

**A.9** Prove that if \( \det A \neq 0 \), then \( A^T A \) is positive definite.

**A.10** Show that if \( x^T A x \) is positive definite, then \( x^T A^{-1} x \) is also positive definite.

**A.11** Show that the eigenvalues of \( AB \) and \( BA \) are identical, where \( A \) and \( B \) are \( n \times n \) matrices.

**A.12** Consider the positive definite quadratic form \( x^T A x \). Show that every eigenvalue of \( A \) is positive.

**A.13** Prove that
\[
\| x + y \|^2 + \| x - y \|^2 = 2(\| x \|^2 + \| y \|^2)
\]

**A.14** Prove that if \( \lambda \) is an eigenvalue of the orthogonal matrix \( A \), then \( 1/\lambda \) is also an eigenvalue of \( A \).

**A.15** Let
\[
A = P \begin{pmatrix} \lambda_1 & & 0 \\ & \lambda_2 & \\ & & \ddots \\ 0 & & & \lambda_n \end{pmatrix} P^{-1}
\]
show that \( \text{tr} A^k = \sum_{i=1}^{n} \lambda_i^k \) for any integer \( k \).

**A.16** Let the minimal polynomial of the \( n \times n \) matrix \( A = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} \) be \( \psi(\lambda) \). Suppose that the minimal polynomial of \( A_1 \) and \( A_2 \) are \( \psi_1(\lambda) \) and \( \psi_2(\lambda) \), respectively. Show that \( \psi(\lambda) \) is the minimum common factor of \( \psi_1(\lambda) \) and \( \psi_2(\lambda) \).

**A.17** a. Show that the gradient of the scalar quantity \( f = x^T A x \), where \( A \) is an \( n \times n \) real symmetric matrix, is
\[
\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} = 2 A x
\]
b. The kinetic energy of a system is given by

\[ E = \frac{1}{2} \dot{q}^T A \dot{q} \]

where \( \dot{q} \) is the \( n \times 1 \) vector of system velocities. Using the result of part (a) show that if \( p_i \) is defined by

\[ p_i = \frac{\partial E}{\partial \dot{q}_i} \quad i = 1, 2, \ldots, n \]

then

\[ \dot{q}_i = \frac{\partial E}{\partial p_i} \quad i = 1, 2, \ldots, n \]

where \( q_i \) is the \( i \)th element of \( q \).

A.18 Show that functions of a matrix commute, that is,

\[ f(A) g(A) = g(A) f(A) \]

and consequently,

\[ Ae^{\lambda} = e^{\lambda} A \]

A.19 Show that if \( \lambda \) is an eigenvalue of \( A \), then \( f(\lambda) \) is an eigenvalue of the matrix function \( f(A) \). Consequently, \( 1/\lambda \) is an eigenvalue of \( A^{-1} \).

A.20 Show that \( A = \begin{pmatrix} a & b \\ 0 & c \end{pmatrix} \) is positive definite if \( a > 0, c > 0 \), and \( |b| < 2\sqrt{ac} \), where \( a, b, \) and \( c \) are scalars.

A.21 Show that the \( n \times n \) matrix \( A \) is positive definite if, and only if, \( (A + A^T) \) is positive definite.

A.22 Let \( B = A^T A \), where \( A \) is an \( m \times n \) matrix. Show that \( B \) is nonnegative definite. Show also that \( B \) is positive definite if, and only if, rank \( A = n \).

A.23 Consider the \( n \times n \) matrix \( \Lambda = \begin{pmatrix} \Lambda_1 \\ \Lambda_2 \\ \vdots \\ \Lambda_n \end{pmatrix} \), where \( \Lambda_i = (1, \lambda_i, \ldots, \lambda_i^{n-1}) \) are \( 1 \times n \) vectors, and \( \Lambda \) is called the Vandermonde matrix. Show that \( \det \Lambda = \prod_{i<j} (\lambda_j - \lambda_i), 1 \leq i < j \leq n \).

A.24 Suppose that \( D \) is nonsingular, determine \( \det \begin{pmatrix} A & B \\ C & D \end{pmatrix} \).

A.25 Use Binet-Cauchy formula to express \( \det(I_n + AB) \) in terms of the minors of \( n \times m \) matrix \( A \) and \( m \times n \) matrix \( B \).

A.26 Let \( G(s) = D + C(sI_m - A)^{-1} B, H(\alpha) = A + B(\alpha I_m - D)^{-1} C \) be \( m \times m \) and \( n \times n \) matrix functions, and define \( \delta(s, \alpha) = \det(\alpha I_m - G(s)), \gamma(s, \alpha) = \det(sI_n - H(\alpha)) \) show that
\[
\begin{vmatrix}
\begin{pmatrix}
sI_n - A & -B \\
-C & \alpha I_m - D
\end{pmatrix}
\end{vmatrix} = \det(sI_n - A) \delta(s, \alpha)
\]

and
\[
\det(sI_n - A) \delta(s, \alpha) = \det(\alpha I_m - D) \gamma(s, \alpha)
\]

A.27 Show that for a square matrix A and any matrices B and C
\[
[I + C(sI - A)^{-1}B]^{-1} = I - C(sI - A + BC)^{-1}B
\]

A.28 Show that if \(A^{-1}\) and \(B^{-1}\) exist
\[
\begin{pmatrix}
A & B \\
C & D
\end{pmatrix}^{-1} = \begin{pmatrix}
(A - BD^{-1}C)^{-1} & -E G^{-1} \\
-G^{-1}F & G^{-1}
\end{pmatrix}
\]
where \(E = A^{-1}B, F = CA^{-1}\) and \(G = D - CA^{-1}B\)

A.29 Prove that \(\text{rank}(AB) \leq \min \{\text{rank}(A), \text{rank}(B)\}\).
Also show that for the \(n \times m\) matrix A and the \(m \times p\) matrix B, we have
\[
\text{rank}(A) + \text{rank}(B) - m \leq \text{rank}(AB) \leq \min\{\text{rank}(A), \text{rank}(B)\}
\]
This is called Sylvester's inequality.

A.30 Show that in general for \(m \times n\) matrices A and B
\[
\text{rank}(A + B) \leq \text{rank}(A) + \text{rank}(B)
\]

A.31 Find eigenvalues and singular values of a square matrix with ones on the first super-diagonal elements and zeros elsewhere.

A.32 Consider the lower triangular \(n \times n\) matrix A with \(-1\)s on the diagonal and \(+1\)s everywhere else. Show that \(\sigma_{\min}(A) = 2^{-n}\), so that for large \(n\) the matrix is almost singular even though all its eigenvalues are nonzero.

A.33 Show that
\[
\sigma_{\min}(A) \leq |\min \lambda(A)| \leq |\max \lambda(A)| \leq \sigma_{\max}(A)
\]
where \(|\min \lambda(A)|\) and \(|\max \lambda(A)|\) are the smallest and largest eigenvalues of A in magnitude.

A.34 Let \(\sigma_1 \geq \cdots \geq \sigma_r \geq \sigma_{r+1} = \cdots = \sigma_p = 0\) be the singular values of A show that
\[
a. \ A = \sum_{i=1}^{r} \sigma_i u_i v_i^T = U_r \Sigma_r V_r^T
\]
where \(U_r = (u_1, \ldots, u_r), V_r = (v_1, \ldots, v_r)\) and \(\Sigma_r = \text{diag} (\sigma_1, \ldots, \sigma_r)\).
\[
b. \ ||A||^2 = \sigma_1^2 + \cdots + \sigma_p^2
\]
\[
c. \ \text{rank} A = r
\]

A.35 Consider the linear system of equations \(Ax = b\), where \(A\) is an \(n \times m\) matrix. Show that
\[
x = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i
\]
where \(u_i\) and \(v_i\) are, respectively, left and right singular vectors, and \(\sigma_i\) are singular values of A.
Appendix B

Computer-Aided Control Systems Design—

Packages and Languages
B.1 INTRODUCTION

One of the basic themes of this text has been the "design" of control systems whose models are expressed either by state equation (time domain) or by transfer function (frequency domain). The other main theme has been the use of computer and computer-aided design programs as "design tools" for modern control systems. The object of this appendix is many fold. After a brief historical note of computer-aided control systems design (CACSD), various techniques and tools will be reviewed. A detailed description of several popular CACSD languages and packages will be given and a brief survey will follow. Finally, an outlook and some conclusions will end our discussions of the subject. The present discussion is, in part, given by various papers in Jamshidi and Herget (1985) and Cellier and Rimvall (1988).

Long before computers were used to design control systems, they were being used to simulate them. One possible definition of simulation, according to Korn and Wait (1987), is that they represent experimentation with models. Each simulation program consists of two primary segments—one is written to represent the mathematical model of the system, while the other is written to do the experiment itself. One of the earliest simulation programs may be CSMP (Continuous Systems Modeling Program) which was a popular simulation environment in the 1960s on various IBM main frame machines. For a long while, CSMP was one of only a few computer programs available to engineers to test the dynamic behavior of physical systems. The basic experiment performed in most simulation runs have been to determine the behavior of the system's trajectory under the influence of known signals such as a "step," "sinusoidal," or similar. This experiment still constitutes the main feature of many simulation packages of today. In other words, a true simulation program should be capable of investigating all model-related aspects of the system such as the errors, uncertainties, tolerances, and so forth. This is not to say that such programs do not exist. In fact, programs such as IBM's DSL (Digital Simulation Language) and ASTAP (Wai and Dost, 1980; IBM, 1984) constitute two notable exceptions. These simulation packages would allow the user to experiment some modeling aspects of physical systems and electronic circuits. Another notable example is the celebrated package SPICE which has been used extensively throughout the world for electronic/electrical network problems. Still, other programs, such as ACSL (Mitchell and Gauthier 1986) offer possibilities to linearize and perform steady-state responses for a nonlinear system. It is, however, a generally accepted opinion that, while model representation techniques have become more powerful over the past years, relatively less has been done to enhance the capabilities of the simulation experiment description.

The main difficulty with many of those packages has been the data structure offered with it. The data structure in many simulation packages is much the same as it was in 1967, when the CSSL specifications (Augustin et al., 1967) were first introduced.

When it comes to CACSD software, the simulation aspect of the system is no longer the main issue. Rather, it is the enhanced experimental description of the system that represents the important issue. In other words, simulation is only one of
the many software modules (tools) available within the CACSD software environment. In sequel, we refer to computational algorithms used for specific design purpose as CAD techniques and the software developed to realize them as CAD tools.

However, the truth is that until recently, the data structure of CACSD packages was just as poor as those among simulation languages. Improper data structure is often time consuming when going through inflexible question-and-answer protocols of earlier CAD packages. This trend followed until Moler (1980) introduced a matrix manipulation laboratory software program he called MATLAB. The only data structure in MATLAB is a double-precision complex matrix. MATLAB offers a set of natural and consistent operators to be performed on matrices. Within MATLAB, a matrix is defined by

\[
A = <3, 4, 5; 6, 7, 8; 9, 0, 1>
\]

or

\[
A = \begin{bmatrix}
3 & 4 & 5 \\
6 & 7 & 8 \\
9 & 0 & 1
\end{bmatrix}
\]

or

\[
A = \begin{bmatrix}
3 & 4 & 5 \\
6 & 7 & 8 \\
9 & 0 & 1
\end{bmatrix}
\]

As seen, elements on the same row are separated by either commas or blanks, and entire rows are separated by either semicolons or a carriage return (CR). The matrices can also be defined by other matrices within the two broken brackets (<···>), that is,

\[
A = <2 * \text{ONES} (4,1), \text{EYE} (4); < -1 -2 -3 -4 -5>>
\]

where ONES (4,1) stands for a 4 × 1 dimensional matrix of all ones as elements, 2 * ONES (4,1) is, therefore, a 4 × 1 matrix whose elements are all 2s. EYE (4) represents a 4 × 4 unity matrix which is concatenated from right to the 4 × 1 matrix of 2s. Therefore, up to the semicolon in this expression of A we have defined a 4 × 5 matrix. The last portion of this expression, a 1 × 5 row matrix < -1 -2 -3 -4 -5>, is concatenated from below; thus, completing the definition of a 5 × 5 matrix called A, that is,

\[
A =
\begin{bmatrix}
2 & 1 & 0 & 0 & 0 \\
2 & 0 & 1 & 0 & 0 \\
2 & 0 & 0 & 1 & 0 \\
2 & 0 & 0 & 0 & 1 \\
-1 & -2 & -3 & -4 & -5
\end{bmatrix}
\]
Assume that one wishes to solve the linear system of equations $Ax = b$ whose solution for a nonsingular $A$ is $x = A^{-1}b$. In MATLAB, this system can be solved in two ways:

$$x = \text{INV}(a) \ast b$$

or

$$x = a \backslash b.$$  

The latter expression indicates that $b$ is divided at left by $a$. While, the former represents a full matrix inversion, in the latter expression a Gaussian elimination is performed. As it may already be noted, MATLAB provides a convenient environment for linear algebra and matrix analysis. A tool like MATLAB did exist in the early 1970s. It was APL (A Programming Language) by IBM. With APL, one could do all the operations that MATLAB can do. However, the biggest difficulty with APL has been its extreme cryptic nature. APL is sometimes called "write only" language. User of APL needed to think the exact way that the computer would execute the APL program. User of MATLAB, on the other hand, would have the computer "think" like the human operator.

It should be noted, however, that MATLAB was not designed for CACSD problems. MATLAB, to quote its creator Cleve Moler, is a "software laboratory for matrix analysis." Nevertheless, when it comes to analysis and synthesis of linear time-invariant systems, MATLAB is exactly what a control engineer would need. To illustrate how MATLAB, in its original form can be used, consider it solving a linear quadratic state regulator problem (Cellier and Rimvall 1988). Consider a linear time-invariant system

$$\dot{x} = Ax + Bu$$  \hspace{1cm} (B.1)

it is desired to obtain a control which would satisfy this state equation while minimizing a cost function,

$$J = \frac{1}{2} \int_0^\infty (x^TQx + u^Tru) \, dt$$  \hspace{1cm} (B.2)

where $u^T$ denotes the transpose of vector $u$. This problem was solved in great detail in Sec. 6.2.2. Associated with this problem is the solution of an algebraic matrix Riccati equation (AMRE) (see Chap. 6).

$$0 = A^TK + KA - KSK + Q$$  \hspace{1cm} (B.3)

where $S = BR^{-1}B^T$. The optimal control is given by

$$u^* = -R^{-1}B^TKx.$$  \hspace{1cm} (B.4)

One of the earliest solutions of Eq. (B.3) was based on the eigenvalues/eigenvectors of the Hamiltonian matrix (Jamshidi, 1980)
\[ H = \begin{bmatrix} A & -S \\ -Q & -A^T \end{bmatrix} \]  \tag{B.5}

This method is incorporated in the following algorithm for the solution of this LQ problem.

1. Check for system's controllability and stop with a message if system is uncontrollable.
2. Compute the Hamiltonian matrix Eq. (B.5).
3. Compute the \(2n\) eigenvalues and right eigenvectors of \(H\). The eigenvalues will be symmetrical with respect to both real and imaginary axes, as long as the system is controllable.
4. Concatenate those eigenvectors with negative eigenvalues into a reduced \(2n \times n\) modal matrix and split them into two \(n \times n\) upper and lower submatrices, that is,

\[ U = \begin{bmatrix} U_1 \\ \cdots \\ U_2 \end{bmatrix} \]

5. The Riccati matrix is given by \(K = Re (U^*_2 U_1^{-1})\), where the asterisk represents conjugate transpose.

The following MATLAB file (called LQ.mti) can be written and executed to realize this algorithm

```
EXEC ('LQ.mti')
IF ans <> n, Show ('system not controllable'), RETURN, END
<v,e> = EIG(<a, -b*(r*b') ; -q, -a'>);
e = DIAG(e); i=0;
FOR j = 1, 2+n, IF e(j) < 0, i = i + 1; v(:,i) = v(:,j); END
u1 = v(1:n,1:i); u2 = v(n+1:2*n, 1 : i);
k = REAL(u2/u1);
p = - r*b' * k
RETURN
```

where the global variable \(ans\) in second line corresponds to the rank of the usual controllability matrix, \(R_c = [B AB A^2B \cdots A^{n-1}B]\). This set of codes is clearly very readable as soon as one is familiar with the MATLAB commands EIG, DIAG, REAL, and so on.

From MATLAB, several CACSD packages and languages were derived in a span of five years. Table B.1 shows a list of six CACSD software programs which have been based on the original MATLAB. These programs are called the "spiritual children" of MATLAB by Rimvall and Cellier (1988).
TABLE B.1 Some MATLAB—Driven CACSD Software

<table>
<thead>
<tr>
<th>Software Name</th>
<th>Location Developed, Year</th>
<th>Principle Developer(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTRL-C</td>
<td>Systems Control, Inc., USA, 1984</td>
<td>J. N. Little, et al.</td>
</tr>
<tr>
<td>IMPACT</td>
<td>SWISS Federal Institute of Technology, Switzerland, 1983</td>
<td>M. Rimvall, et al.</td>
</tr>
<tr>
<td>CONTROL.lab</td>
<td>University of New Mexico, USA, 1985</td>
<td>M. Jamshidi, et al.</td>
</tr>
<tr>
<td>MATLAB-SC</td>
<td>Philips Research Laboratories, Germany, 1985</td>
<td>M. Vanbegin and P. Van Dooren</td>
</tr>
</tbody>
</table>

These CACSD programs will be compared from a number of comparative points of views later on.

Parallel to and even before the discovery of MATLAB by control systems engineering community, a number of non-MATLAB software programs were created both in North America and Western Europe. A list of eight non-MATLAB CACSD programs are summarized in Table B.2. It must be noted that there are still a good number of CACSD and simulation packages which have been developed. These software programs are either not relevant to the main theme of this chapter and are specialized in nature, or are less publicized by their authors. In any event, some of the software programs not listed in Tables B.1 and B.2 will be briefly discussed later on.

TABLE B.2 Some non-MATLAB CACSD Software Packages

<table>
<thead>
<tr>
<th>Software Name</th>
<th>Location Developed, Year</th>
<th>Principle Developer(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEDDC (CADACS)</td>
<td>University of Bochum, Germany, 1979</td>
<td>H. Unbehauen and Chr. Schmid</td>
</tr>
<tr>
<td>TIMDOM</td>
<td>University of New Mexico, 1983</td>
<td>M. Jamshidi, et al.</td>
</tr>
<tr>
<td>CC</td>
<td>California Inst. Technology, USA, 1984</td>
<td>P. M. Thompson</td>
</tr>
<tr>
<td>TRIP</td>
<td>Delft University of Technology, the Netherlands, 1985</td>
<td>P. P. J. Van den Bosch</td>
</tr>
<tr>
<td>WCDS</td>
<td>University of Waterloo, Canada, 1986</td>
<td>J. D. Aplevich</td>
</tr>
<tr>
<td>CATPAC</td>
<td>Philips Laboratories, Germany, 1986</td>
<td>D. Buenz</td>
</tr>
</tbody>
</table>
B.2 DEVELOPMENT OF CAD METHODS AND PROGRAMS

B.2.1 CAD Methods

The development of computer-aided design methods and programs have traditionally been dependent on the theoretical schemes available to the designer. The design methods, by and large, had been graphics in nature and were limited to single-input single-output (SISO) systems described in frequency domain. The major developments in the design of these control systems were the advents of Nyquist stability criterion, Bode Plots, Root locus plots and Nichols charts (D’Azzo and Houpis, 1966). During the decades of 1930s through 1960s, the use of computers as a design tool was either nonexistent or very much limited to writing a subroutine, most likely in FORTRAN, and a main program to call it, all in a batch mode. For example, in the late sixties or early seventies, finding the eigenvalues of an $n \times n$ ($n$, say 5) matrix would take about half a day, even with reliable eigenvalue/eigenvector routines.

As the control systems become more complex, that is, with multi-input multi-output (MIMO), new design techniques and subsequently CAD tools were needed. Clearly, for MIMO systems, the graphical techniques of SISO systems were not appropriate and could not provide sufficient insight. Kalman, among many others, led the way to a new domain of system analysis and design—**time domain or state-space methods**. Here, the system is described by a set of first-order ordinary differential equations. Refer to Chap. 2 for details of the state-space representation of MIMO systems. This modern approach seemed more appropriate for many algorithmic design methods. These MIMO design algorithms, for example, linear quadratic state regulator (see Chap. 6) are applicable equally to both SISO and MIMO systems.

The modern control design methods often resulted numbers, parameters, values, gain factors, and so forth which often lacked an "intuitive feel" for what was being achieved by applying these new techniques. However, without adequate insight into the controlled plant, it was often very difficult to determine an adequate controller structure.

Because of these reasons, many control scientists concentrated again on frequency domain methods in an attempt to find new design schemes for MIMO systems. Notable examples of these efforts were the generalization of Nyquist diagram by Rosenbrock (1969), matrix polynomial representation (Wolovich, 1974; Wonham, 1979), and robust controller design (Ackerman, 1985). These techniques, however, are very sensitive to large number crunching.

Another area of development in control system design techniques has been **large-scale systems** (LSS). These systems, as seen in Chap. 7, are associated with high dimensions among other complex attributes such as nonlinearities, delays, and so on. The advent of large-scale systems has confronted the control system designer with two new challenges. One is the need for new control philosophies such as hierarchical and decentralized control (see Chap. 7) and the other is the need of special CAD algorithms to handle this class of systems.

An area where relatively little design algorithms exist is nonlinear systems.
Most attempts in the design of nonlinear control systems have had to specialize in either narrowing down the class of systems treated or using linear systems' linearization techniques for otherwise nonlinear problems. The robustness of these refined algorithms is very poor, that is, they can not easily handle the design of controllers for systems with either unmodeled dynamics or plants whose parameters vary. A prime example of such systems is robot manipulators whose models are highly nonlinear and plant parameters change. A popular scheme to treat nonlinear systems has been adaptive control (Landau, 1979, Chalam, 1987). The controller gains adapt themselves in accordance with the plant parameter variations. Another potential scheme is robust control (Dorato, 1987, Dorato and Yedavali, 1990) where the controller can handle the plant whose parameters undergo certain variations.

CAD techniques algorithms can be categorized into three groups: (a) SISO systems techniques, (b) MIMO systems techniques and (c) large-scale systems techniques. Corresponding to this categorization, as indicated by Cellier and Rimvall (1988), three classes of algorithms can be distinguished: (a) SISO algorithms which can effectively handle low-order systems, (b) MIMO algorithms which can efficiently handle high-order systems, and (c) LSS algorithms which are dedicated for very high-order systems.

In the theory of linear control systems, most design schemes are based on the systems’ canonical form (see Chaps. 2, 4, and 5). As seen in the previous chapters, these forms are based on "minimum parameter data representation." To describe this notion, let us consider an \( n \)th order SISO system represented by the following transfer function

\[
\frac{c_n s^{n-1} + \cdots + c_2 s^2 + c_1 s + c_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}
\]

where the denominator is the characteristic polynomial. In this system representation there are \( 2n \) degrees of freedom, that is, \( (a_0, a_1, \ldots, a_{n-1}, c_0, c_1, \ldots, c_{n-1}) \). The controllable canonical representation of this system is given by,

\[
\dot{x} = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
0 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1
\end{bmatrix} x + \begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
1
\end{bmatrix} u
\]

\[y = [c_0, c_1, c_2, \ldots, c_{n-1}] x\]

If the system has \( n \) distinct eigenvalues, \( \lambda_1, \lambda_2, \ldots, \lambda_n \), the Jordan-canonical representation of the system is given by
\[ \dot{x} = \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_n \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 1 \end{bmatrix} u \]

where \( b_i, i = 1, 2, \ldots, n \) are constant coefficients.

These two representations are valid only for SISO (low-order) algorithms. This indicates that for systems of higher order, the controllable canonical forms are not the most desirable representation. The biggest advantage of this representation is its simplicity. If one is interested in obtaining high-order algorithms, this "simple" representation must be modified by introducing some redundancy among its \( 2n \) degrees of freedom. Under this condition, the system would have more than \( 2n \) degrees of freedom with linear dependencies existing between them. It is through this redundancy that one can optimize the numerical behavior of controller design algorithms by balancing the parameter's sensitivities (Laub, 1980, Cellier and Rimvall, 1988). More efficient high-order algorithms based on the Hessenberg representation can also be obtained (Patel and Misra, 1984).

For large-scale systems, as mentioned before, one can decompose (reduce) it into a finite number of small-scale subsystems and use lower order algorithms for each of them. On the other hand, one can use the sparse matrix techniques for the control of the original large-scale system. These techniques keep track of the indices of nonzero elements and are computationally cost effective only for very high-order systems and not for low-order or high-order algorithms.

In summarizing the development of CAD techniques for control systems design, one can classify them into several schemes: SISO versus MIMO systems versus LS systems; continuous-time versus discrete-time; time-domain versus frequency-domain; linear versus nonlinear systems, user-friendly versus non-user-friendly, just to name a few. The last category refers to the algorithms in which the user concentrates on the physical parameters of the problem being solved and not the computational or numerical attributes of the algorithms themselves. An example of this situation is a variable-order and variable-step-size numerical integration algorithm (such as Runge-Kutta 45 method) in which only the required accuracy, a physical parameter needs to be specified by the user.

### B.2.2 CAD Programs

As mentioned earlier, up to 1970 no tangible development had taken place for CAD programs used in the design of control systems—multivariable or otherwise. The only available computer programs were some lists of subroutines programs collected by users of a particular line of computers. An example of this was IBM’s Scientific Subroutine Package (SSP) in the 1960s which consisted of a score of routines for various problems in linear algebra and numerical mathematics.
In the earlier days, the use of computer as a design tool for control systems boiled down to writing a main program and call one or a set of subroutines. The book by Melsa and Jones (1973) was one of the first published works in this area. In the early to mid-1970s a gradual interest was developed both in Europe and North America to coordinate efforts to put subroutine libraries together. In fact, colleagues at Swiss Federal Institute of Technology in Zürich led by Cellier and Rimvall (1988) helped develop a “program information center” (PIC) by mid-1970s in Europe.

The initial efforts in CACSD programs began in late 1970s by creating interactive “interface” to existing subroutine library such as the one gathered in Europe to reduce the turnaround time. Towards this goal, Agathoklis et al. (1979) at the Swiss Federal Institute of Technology came up with the first generation of an interactive CACSD package they called INTOPS. This package proved to be a surprisingly useful new educational tool, but it could not be an effective research tool.

The next step in the development of CACSD turned out to be actually a giant step in the right direction. In the fall of 1980, a conference on numerical and computational aspect of control systems was organized at Lund Institute of Technology in which MATLAB (Moler, 1980) was formally introduced. In matter of months after that meeting, MATLAB was installed at many locations all over the world. With its one-line “help” facility, MATLAB became a very simple new tool for students of mathematics and system theory. MATLAB resembled APL, as mentioned before, but without the latter's highly cryptic nature.

In spite of all its fine features, MATLAB was not really designed as CACSD program or tool. In fact, as Cellier and Rimvall (1983) noted, from the viewpoint of a control engineer, MATLAB had the following shortcomings:

1. MATLAB cannot handle frequency domain design techniques easily,
2. MATLAB cannot handle nonlinear systems,
3. MATLAB did not have any plotting capabilities which is a major factor in CACSD,
4. MATLAB's programming capabilities through EXEC files is not sufficient. EXEC files cannot be called as functions, but only as subroutines. Their input/output capabilities are limited.
5. There is no GOTO statement in MATLAB and, furthermore, WHILE, IF, and FOR cannot be properly nested.
6. Using SAVE/LOAD facility of MATLAB would require handling a large number of files which is difficult to maintain. A genuine data base would have been desirable.

These points were, in part, taken into account by a number of subsequent CACSD packages which will be discussed in Sec. B.3.

To summarize the development of CAD programs for control system design, we would go through a number of classification of CACSD programs. Figure B.1
shows one possible classification. As seen, on top, the earlier categorization of CACSD packages were either a "subroutine library" or an "integrated" set of programs (modules). The early development of these packages were of the former, while the more recent packages are of the latter type. Leaving the earlier "subroutine library" types on the side, the "integrated" CAD packages may be further divided into two types. One is those in which an open-ended operator set of design "shells" would allow the user to code its own algorithm using the CACSD software environment. All the MATLAB-based packages (Sec. B.3) are of this type. On the other hand, some packages have gone through a fairly long history of maturity throughout years so that we can refer to them as "comprehensive." A good example of this class of packages is KEDDC (or CADACS) developed in Germany since the early 1970s (Schmid, 1979, 1985). More details on this package are given in Sec. B.4.

From this point, both design "shells" or "comprehensive" packages can be further categorize in several ways and directions, depending on various input/output data-handling, or actual implementation of the design algorithms. As an example, a "comprehensive" CACSD package can be subdivided into a completely interactive, batch-operated, or both. When dealing with nonlinear or even linear time-varying systems, one often has to do a substantial amount of number crunching in "batch" mode. On the other hand, many CACSD packages dealing with linear time-invariant systems operate in a fully "interactive" mode for a quick analysis and design of such
systems. A good example of a package which can be used in both interactive and batch modes is TIMDOM/PC (see Sec. B.4).

Another possible classification of CAD programs for control systems is the style in which algorithms are coded in. This categorization is either "code-driven" or "data-driven." The packages in which algorithms and operators are implemented as data statements are called "data-driven." These packages are usually written in interpretive language (e.g., BASIC) and are useful for experimentation. However, they are not terribly efficient in execution. On the other hand, "code-driven" packages are based on compiled programs whose algorithms and operators are implemented in program code. The algorithms and operators of "code-driven" packages are usually much faster in execution and represent a more stabilized state of a package’s development.

A final possible categorization of CACSD packages is through the styles of "driving through" it, that is, moving on from beginning to the end of a given CAD session. As it is shown on the bottom of Figure B.1, six possible styles can be mentioned. In the "question and answer" style, the user is asked numerous questions to determine the flow of the program. This style of user interface, although very simple to implement, is not useful for research-oriented problem solving. "Menu-driven" packages are those in which the user is always given a choice list of various categories of CACSD problems. The user can then choose one of the choices on the list and that specific program is loaded and made available to the user. The user can make its choice by either a pointer or a mouse on the screen. Many non-MATLAB packages available today are menu-driven (see Sec. B.4). Some believe that this type of user interface shares too much information leading to a slower package. However, we believe that carefully designed menu-driven interfaces would have more educational use rather than research.

The other popular interface is "command-driven." Here, the user would need to have the initiative to figure out what to do next. The CACSD program would "prompt" the screen with a symbol to indicate to the user that it is ready for the next "command." All MATLAB-based (Sec. B.3) packages are "command-driven." One disadvantage of this style of interface is that the user has to remember what commands are available and how they work. The HELP facility in MATLAB-based programs would come in handy here.

A "form-driven" interface is most useful during the setup stage of a problem solution. The screen is often divided into separate alphanumeric fields, each representing one parameter of the algorithm. By moving over fields, the user can override default parameter values. This type of interface is heavily dependent on the hardware and very few CACSD programs utilize it.

The early utility of "graphic-driven" interface in CACSD applications was to display system's time and frequency responses. Being highly dependent on the terminal and hardware, graphics still remains the most difficult issue in any CACSD package. The basic approach in alleviating this problem has been to utilize a graphics library which supports a wide variety of terminal drivers to put between the package and terminal itself. However, these libraries have been too extensive, and their full utility has been questionable. The recent acceptance of the "graphic kernel system" (GKS)
as standard (ANSI, 1985) is clearly a positive step in the right direction toward eliminating hardware dependencies on terminals. As indicated by Cellier and Rimvall (1988), fancy graphics often call for high-speed communication links, which would drive costs up again. GKS standardization would, of course, help drive cost down. The appearance of special-purpose graphics workstations such as APPOLO Domain and SUN would provide high-speed enhanced graphics capabilities at reasonable prices.

Another graphics-related outcome in CACSD package developments has been graphics-input feature of some programs. Examples of this are SYSTEM-BUILD capability of MATRIXx (Shah et al, 1985) and BUILD features of CC (Thompson and Wolf, 1983), and PRO-MATLAB (Little and Moler, 1986). A powerful upcoming simulation package for nonlinear systems, suggested by Robinson and Kisner (1988), McClung and Jamshidi (1992), uses artificial intelligence and object-oriented programming to develop a truly graphics-driven package. In this package, as well as in MATRIXx, control system blocks are automatically translated by a graphics compiler into a coded model format.

A last interface is that of “window-driven” variety (see Fig. B.1). Here, the screen is divided into a number of logical windows, each representing one logical unit—similar to the physical devices in a computer system used to be. Within each window, one may employ any one of the described interfaces, that is, question-and-answer, menu-driven, command-driven, or form-driven. In a recently developed robot simulation package ROBOT_S by O’Neill and Jamshidi (1989), a number of windows are utilized (on a SUN workstation), each with its own menu-driven interface. Another CACSD package which allows mixture of two interfaces is IMPACT (Rimvall and Cellier, 1985), a MATLAB-based package, which has extended MATLAB’s HELP facility by an extensive QUERY facility. Using QUERY, the user can receive “help” at either command level or at an entire session level. In this way, the user can choose an almost surely question-and-answer extreme all the way to a solely command-driven feature on IMPACT. More details can be obtained in the authors’ recent survey (Cellier and Rimvall, 1988). Further advanced topics and development on CACSD will be forthcoming in another volume by Jamshidi and Herget (1993).

### B.3 MATLAB-BASED CACSD PACKAGES

As discussed earlier, there are at least six CACSD packages which have been designated as the “spiritual children” of MATLAB (see Table B.1), including PROMATLAB—a modern version of MATLAB. In this section, an attempt is made to highlight the common and distinguished features of some of these packages. Because of the lack of the availability of all of MATLAB-based packages, only four of these packages will be covered in a brief manner. These are CTRL-C (Little et al., 1985), MATRIXx (Shah et al., 1985), CONTROL.lab (Jamshidi et al., 1986) and PC-Matlab (Little and Moler, 1985).

The advent and impact of MATLAB on CACSD environment in the 1980s has already been brought up in some detail. Furthermore, aside from MATLAB’s con-
venient matrix data structure, from a numerical analysis point of view, numerically stable algorithms have been used throughout this "laboratory" of matrix analysis which have stemmed from previous successful packages such as EISPAK (Smith, et al., 1973) and LINPAK (Dongarra, et al., 1979). Figure B.2 shows a generic structure of a typical MATLAB-based package. These packages typically have a graphics interface to either externally-supported plotting language or its own internal plotting package, and a number of other modules which we have, generically, designated as PAK1, PAK2, ..., PAKN. Therefore, the common feature of all these packages are MATLAB's EISPAK and LINPAK algorithms, HELP facility as well as their command-driven nature. For this reason, linear algebra capabilities of these packages are very much the same.

However, each of these packages do possess some settled differences which we will try to point out in the next subsections.

### B.3.1 CTRL_C

One of the earliest MATLAB-based programs which came in existence in 1984 was CTRL_C by Systems Control Technology, Inc. whose principle authors were J. Little, A. Emami-Naeini, and S. N. Bangert (Little et al., 1985). In CTRL_C, like all MATLAB-based packages, all variables are stored in a large stack which resides in either semiconductor or virtual memory, or both. There is, however, a natural data communications between this stack and disk files. All MATLAB-based packages inform the user of readiness to accept the next command. In CTRL_C, the combinations of "[]>" appears on the screen, while in MATRIXx and CONTROL.lab, MATLAB's original facing broken bracket "<>" are used. In PC-MATLAB, the symbols ">>" are utilized to prompt the user.

**Matrix analysis.** To enter a matrix in CTRL_C, the same as all such packages, a simple list is used inside two brackets whose rows are distinguished by a semicolon ";". For example, the input line


\[
[> \ a = [1 \ 5 \ 9 \ 3; 2 \ 6 \ 1 \ 4; 3 \ 7 \ 1 \ 6; 4 \ 8 \ 2 \ 8]
\]

in CTRL_C would result in the output

\[
A =
\begin{align*}
1. & \quad 5. & \quad 9. & \quad 3. \\
2. & \quad 6. & \quad 1. & \quad 4. \\
3. & \quad 7. & \quad 1. & \quad 6. \\
4. & \quad 8. & \quad 2. & \quad 8.
\end{align*}
\]

[>]

From this point on, this \(4 \times 4\) matrix, denoted by \(A\), will be stored for later use in the same terminal session. The matrix analysis in CTRL_C, stemming from MATLAB, is very simple. For example, the relation (or command) \(b = a'\) would result in the transpose of matrix \(A\) as defined. Matrix multiplication, inversion, determinant, rank, condition number, and so on, similarly can be determined by

\[
[> \ c = a * b; \\
[> \ \text{ain} = \text{inv}(a); \\
[> \ d = \text{det}(a); \\
[> \ r = \text{rank}(a); \\
[> \ \text{cn} = \text{cond}(a);
\]

where the semicolon at the end of each statement, the same as all MATLAB-based programs, signals to the package not to print out the results, rather save them in appropriate newly-defined variables. The original MATLAB (Moler, 1980) came with 77 commands, key words or symbols such as "::", FOR, DO, IF, and so on. From those, 16 were purely dedicated linear algebra (matrix analysis) commands. Some of the common matrix functions in CTRL_C, inherited from MATLAB, are

- \(\text{eig}(a)\) - eigenvalue and eigenvectors
- \(\text{exp}(a)\) - matrix exponential
- \(\text{inv}(a)\) - inverse
- \(\text{svd}(a)\) - singular-value decomposition
- \(\text{Schur}(a)\) - Schur decomposition

However, commands such as \(\text{geig}(a, b)\) which finds the generalized eigenvalues and eigenvectors of \(a\) is a dedicated CTRL_C command. Polynomials are represented by a vector of its coefficients ordered by their descending power. In CTRL_C, \(\text{conv}(a, b)\) can be used to find the product of polynomials \(a\) and \(b\), that is

\[
[> \ a = [1 \ 2 \ 1]; \ b = [1 \ 2]; \\
[> \ c = \text{conv}(a, b)
\]

yields
\[ c = 1. \quad 4. \quad 5. \quad 2. \]

Several other desirable operations such as polynomial division, root finding, characteristic polynomial, and other polynomial operations are similarly supported.

**Engineering graphics.** In CTRL\_C, like many well-developed MATLAB-based packages, a score of graphics commands and features such as Plot, log-log, 3d Plots, labels, title, axes, and so on, are supported. As an example, the following four CTRL\_C commands would create a simple sine function

\[
[> \quad t = 0 : .05 : 4 * \pi; \quad y = \sin (t); \\
[> \quad \text{Plot}(t,y), \text{title('sin(t}')
\]

which is shown in Fig. B.3. The first statement would define a vector consisting of elements from 0 to 4\pi in increments of 0.05. The second statement (on first line) would create a vector of sine values of time vector \( t \). The third statement would plot vector \( y \) versus vector \( t \) and provide a title for it.

CTRL\_C would allow one to use 3D plots for a better understanding of the structure of large system matrices. For example, "to look" at a fifty-ninth-order aircraft system matrix, one can use command P3d(a) to produce a 3-dimensional plot of Fig. B.4, where the height \( h \) represents the value of an element above the X-Y plane. This plot would give a perspective to the designs which would not be evident by looking at 3600 numbers.

![Figure B.3 A simple sine function plotted in CTRL-C.](image-url)

Control systems. It is clear that for linear time-invariant systems in state-space form, matrix environments are particularly useful. Within CTRL_C, systems may be represented in discrete-time, continuous-time, in polynomial notation, as a Laplace transfer function, or a z-transform transfer function. Within CTRL_C, one can move from time-domain to frequency-domain and vice versa. In sequel, a simple example illustrates some of the features of CTRL_C.

CAD Example B.1
Consider a third-order SISO system described in Figure B.5. In this example various features of CTRL_C will be illustrated. To begin, the numerator and denominator coefficients for the first block are entered:

\[
\text{num} = \begin{bmatrix} 1 & 2 \end{bmatrix}; \quad \text{den1} = \begin{bmatrix} 1 & .4 & 1 \end{bmatrix};
\]

To determine the open-loop poles of the first block,

\[
\text{dr} = \text{root(den)}
\]

\[
\text{DR} = \\
-0.2000 + 0.9798i \\
-0.2000 - 0.9798i
\]
The natural frequency and damping ratio of this block are easily determined by

\[
\text{wn} = \text{abs}(	ext{dr})
\]

\[
\text{WN} =
\begin{align*}
1.0000 \\
1.0000
\end{align*}
\]

\[
\text{zeta} = \cos(\text{imag}((\log)(\text{dr})))
\]

\[
\text{ZETA} =
\begin{align*}
0.2000 \\
0.2000
\end{align*}
\]

To cascade the second block of the system in Fig. B.5, a new vector of denominator of this block is introduced,

\[
\text{den2} = [1 \ 1.96];
\]

Through the convolution command, the series connection denominator can be obtained:

\[
\text{den} = \text{conv}(	ext{den1,den2})
\]

\[
\text{DEN} =
\begin{align*}
1.0000 & \quad 2.3600 & \quad 1.7840 & \quad 1.9600
\end{align*}
\]

This equivalent cascaded system can be realized in state space form by the following statement:

\[
[a,b,c,d] = \text{tf2ss}(	ext{num,den})
\]

\[
D =
\begin{align*}
0
\end{align*}
\]
C =
  0.  1.  2.

B =
  1.
  0.
  0.

A =
-2.3600  -1.7840  -1.9600
  1.0000   0.0000   0.0000
  0.0000   1.0000   0.0000

With the system now in state-space format, one can determine time and frequency responses. To do that in CTRL_C, like all other MATLAB-based packages, using colon "::" command, one can first define a time base, that is, the statement

\[ t = 0 : 0.1 : 10; \]

creates a vector of 101 time points spanning from 0 to 10 in steps of 0.1 or 100 ms. The following two statements would compute, respectively, the impulse and step responses of the system using the newly created time base

\[ y_i = \text{impulse}(a,b,c,1,t); \]
\[ y_s = \text{step}(a,b,c,d,1,t); \]

The 101 \times 1 vectors \( y_i \) and \( y_s \) represent the system’s output time responses under the influence of an impulse and a step, respectively. To plot these responses, one can simply type in,

\[ \text{Plot}(t,y_i,t,y_s) \]

which results in the plots of Fig. B.6.

In a similar fashion, one can plot frequency responses. This is achieved by using the function LOGSPACE to create a vector of frequency points equally spaced between two decades, that is,

\[ w = \text{logspace}(-1,1); \]
\[ [\text{mag, phas}] = \text{bode}(a,b,c,d,1,w); \]

where matrices \( \text{mag} \) and \( \text{phas} \) contain the magnitude and phase responses at the frequencies in vector \( w \). The magnitude and phase are plotted on log-log scales and titled in the upper right corner of the screen with the commands.
Figure B.6  Time responses for system of CAD Example B.1.

\[
\begin{align*}
[&> \text{window('222')} ] \\
[&> \text{Plot(w,mag,'loglog')} ] \\
[&> \text{title('magnitude')} ]
\end{align*}
\]

Similar commands are used to plot the phase and Nichols chart, for example, as shown in Fig. B.7.

Finally, one can use pole placement algorithm of Ackerman (1985) coded in command PLACE to design a state feedback for this open-loop system. Assume that the desired poles are \(-3, -3 \pm j3\) which are stored away in a vector \(P\). Then the feedback gain \(k\) is obtained by

\[
[&> p = 3 \ast [-1; (-1 + i); (-1 - i)]; ] \\
[&> k = \text{Place}(a,b,p) ] \\
K = \\
6.6400 \quad 34.2160 \quad 52.0400
\]

which can be checked to provide the desired poles by

\[
[&> e = \text{eig}(a - b \ast k) ] \\
E = \\
\begin{align*}
&-3.0000 + 3.0000i \\
&-3.0000 - 3.0000i \\
&-3.0000 + 0.0000i
\end{align*}
\]

which does check.
The reference feedforward matrix \( M \) is calculated to provide unity DC gain,

\[
|> m = 1/(d - (c - d * k)/(a - b * k) * b)
\]

\[M = \]

27.0000

To determine the impulse and step responses of the closed-loop system, the following statements would perform that

\[
|> ac = a - b * k; bc = b * m; cc = c - d * k; dc = - d * k;
\]

\[
|> yi = impulse(ac,bc,cc,1,t);
|> ys = step(ac,bc,cc,dc,1,t);
|> Plot(t,yi,t,ys)
\]

which results in Fig. B.8.

**Figure B.7** Frequency responses for system of CAD Example B.1.
CTRL\_C has numerous other capabilities that space would not allow to cover. In the area of signal processing, for example, CTRL\_C has fast Fourier transform, filter design, and so on. Moreover, within CTRL\_C, the user can use its "procedure" capability to create new commands. This capability is available on all MATLAB-based packages and discussion of this type will be discussed later.

### B.3.2 MATRIXx

MATRIXx is generally considered as the first commercially available CACSD package based on MATLAB. The structure of this package can be summarized through the following broad categories.

- Matrix, vector and scalar operations
- Graphics
- Control design
- System identification and signal processing
- Interactive model building (SYSTEM BUILD)
- Simulation and evaluation

The structure of MATRIXx is shown in Fig. B.9. The details of each segment is beyond the scope of this section, but instead greater detail will be given to SYSTEM\_BUILD which is one of the features that distinguishes MATRIXx from many other
programs. However, before dealing with that feature, we can summarize the capabilities of MATRIXx (Version 6.0, May, 1986) through Tables B.3 to B.8 to show a summary of the available tools. As can be seen from these tables, the capabilities of MATRIXx is quite vast. This package has been used in Chapter 6 before. Covering the full capabilities of MATRIXx would be a book in itself. Instead of going into details of this CACSD package, we would first introduce a number of CAD examples to illustrate its features and then get into details of SYSTEM_BUILD and follow it with another CAD example.

![Diagram of MATRIXx structure](image)

Figure B.9 Structure of MATRIXx.

<table>
<thead>
<tr>
<th>TABLE B.3</th>
<th>Examples of MATRIXx Capabilities: Matrix Arithmetic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Entry, Display and Editing</td>
<td></td>
</tr>
<tr>
<td>Addition, Subtraction, Multiplication and Division</td>
<td></td>
</tr>
<tr>
<td>Absolute Value, Real Part, Imaginary Part and Complex Conjugate of a Matrix</td>
<td></td>
</tr>
<tr>
<td>Sum and Product of Matrix Elements</td>
<td></td>
</tr>
<tr>
<td>Element-by-Element Multiply and Divide</td>
<td></td>
</tr>
<tr>
<td>SIN, COS, ATAN, SORT, LOG, EXP of Matrix Elements</td>
<td></td>
</tr>
<tr>
<td>Eigenvalue, Singular Value, Principal Values, Schur, LU, Cholesky, OR, Orthogonal, Hessenberg, and Reduced row echelon</td>
<td></td>
</tr>
<tr>
<td>Decomposition of Matrices</td>
<td></td>
</tr>
<tr>
<td>Random Vector and Matrix Generation and Manipulation</td>
<td></td>
</tr>
<tr>
<td>Eigenvectors, Generalized eigenvectors, Pseudo inverse, condition number, rank, norm, inverse Hilbert matrices, and rational numbers</td>
<td></td>
</tr>
</tbody>
</table>
TABLE B.4 MATRlXX Capabilities: Graphics

Flexible Commands
Multiple Plots
Axis Labels and Plot Title
Symbols, Lines and Styles
Tics and Grids
Log Scales
Bar Charts
Labeling and Scales
Polar Plots
Plot Location and Size
Personalization
Report Quality
3-D Graphics
  Parallel & Perspective Projections
  Surfaces
  Curves
  Viewing Transformation
Font size, style and changes
Handcopy, redrawing

CAD Example B.2

The definition of a surface is a family of points defined by vectors \( x \) and \( y \), and a matrix \( Z \) so that \( Z = \text{dim}(x) \times \text{dim}(y) \). The surface is then \( z = f(x, y) \). Individual points are linearly connected.

As an example, consider the following MATRlXX statements:

\[
\begin{align*}
\langle \rangle & \ X = \lfloor -2 \pi : .35 : 2 \pi \rfloor \ ; \ Y = X; \\
\langle \rangle & \ Z = \sin(X)/X*(\sin(Y)/Y); \\
\langle \rangle & \ PLOT(X,Y,Z)
\end{align*}
\]

which produces the surface plot of Figure B.10.

![Figure B.10](image)

**Figure B.10** A three-dimensional plot produced by MATRlXX.
### TABLE B.5 MATLABx Capabilities: System Descriptions and Simulations

<table>
<thead>
<tr>
<th>System Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous-time Systems</td>
</tr>
<tr>
<td>Transfer-Function Descriptions</td>
</tr>
<tr>
<td>State-Space Descriptions</td>
</tr>
<tr>
<td>Conversion Between State-Space and Transfer-Function Forms</td>
</tr>
<tr>
<td>Discrete-Time Systems</td>
</tr>
<tr>
<td>Transfer-Function Descriptions</td>
</tr>
<tr>
<td>State-Space Descriptions</td>
</tr>
<tr>
<td>Conversion Between State-Space and Transfer-Function Form</td>
</tr>
<tr>
<td>Conversions from Continuous-Time to Discrete-Time Descriptions</td>
</tr>
<tr>
<td>Systems in Modal Form</td>
</tr>
<tr>
<td>Appending System Dynamics</td>
</tr>
<tr>
<td>Series Connections</td>
</tr>
<tr>
<td>Parallel Connections</td>
</tr>
<tr>
<td>Feedback Connections</td>
</tr>
<tr>
<td>General Interconnections</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous-time Systems</td>
</tr>
<tr>
<td>Impulse Response</td>
</tr>
<tr>
<td>Step Response</td>
</tr>
<tr>
<td>Initial-value Response</td>
</tr>
<tr>
<td>Response to General Inputs</td>
</tr>
<tr>
<td>Discrete-time Systems</td>
</tr>
<tr>
<td>Pulse Response</td>
</tr>
<tr>
<td>Step Response</td>
</tr>
<tr>
<td>Initial-value Response</td>
</tr>
<tr>
<td>Response to General Inputs</td>
</tr>
<tr>
<td>Nonlinear Simulations</td>
</tr>
<tr>
<td>Simulate general nonlinear continuous models</td>
</tr>
<tr>
<td>Simulate general nonlinear discrete models</td>
</tr>
<tr>
<td>Simulate general nonlinear multirate models</td>
</tr>
<tr>
<td>Simulate general nonlinear hybrid multirate models</td>
</tr>
</tbody>
</table>

In MATLABx, a MIMO linear time-invariant system \( \dot{x} = Ax + Bu, \ y = Cx + Du \) is represented by a system matrix

\[
S = \begin{pmatrix} A & B \\ C & D \end{pmatrix}
\]

which can be realized by

\[
<> S = [a, b; c, d];
\]

On the other hand, to extract the system's matrices, one can use command "SPLIT"

\[
[A,B,C,D] = \text{SPLIT}(S,NS);
\]
TABLE B.6 MATRIXx Capabilities: Control Design and System Analysis Capabilities (Applicable to Continuous, Discrete and Hybrid Systems)

<table>
<thead>
<tr>
<th>Classical Tools</th>
<th>Modern Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Locus</td>
<td>Pole Placement</td>
</tr>
<tr>
<td>Bode Plots</td>
<td>Optimal Control Design, Discrete and Continuous</td>
</tr>
<tr>
<td>Nyquist Plots</td>
<td>Optimal Filter Design, Discrete and Continuous</td>
</tr>
<tr>
<td>Nichols Plots</td>
<td>Frequency-Shaped LQG Design</td>
</tr>
<tr>
<td>Gain and Phase Margins</td>
<td>Singular-Value Decomposition of the Return-Difference</td>
</tr>
<tr>
<td></td>
<td>Eigensystem Decompositions Including the Jordan Canonical Form</td>
</tr>
<tr>
<td></td>
<td>Model Following Control</td>
</tr>
<tr>
<td></td>
<td>Linearization of Nonlinear Systems</td>
</tr>
<tr>
<td></td>
<td>Minimal Realization and Kalman Decomposition</td>
</tr>
<tr>
<td></td>
<td>Geometric Control Algorithms</td>
</tr>
<tr>
<td></td>
<td>Multivariable Nyquist Plots</td>
</tr>
<tr>
<td></td>
<td>Transmission Zeros</td>
</tr>
<tr>
<td></td>
<td>Algebraic Riccati Solutions</td>
</tr>
<tr>
<td></td>
<td>Model Transformation and Reduction</td>
</tr>
<tr>
<td></td>
<td>Controllability, Observability and Minimal Realization</td>
</tr>
<tr>
<td></td>
<td>Controllable Realization</td>
</tr>
<tr>
<td></td>
<td>Observable Realization</td>
</tr>
<tr>
<td></td>
<td>Minimal Realization</td>
</tr>
<tr>
<td></td>
<td>Stair-Case Form</td>
</tr>
<tr>
<td></td>
<td>Eigenvalue Test</td>
</tr>
<tr>
<td></td>
<td>Grammian Test</td>
</tr>
<tr>
<td></td>
<td>Transforming to Internally Balanced Form</td>
</tr>
<tr>
<td></td>
<td>Transforming to Modal Form</td>
</tr>
<tr>
<td></td>
<td>Evaluating Modal Residues</td>
</tr>
<tr>
<td></td>
<td>Model Reduction</td>
</tr>
</tbody>
</table>

or

\[
A = \text{SPLIT}(S,N); \\
\]

where \(N\) is a variable representing the number of states. The same format is used for discrete-time systems.

One of the more powerful sets of commands in MATRIXx are tools for connecting linear systems in series, parallel, or feedback configurations. SERIES, PARALLEL, FEEDBACK, AFEEDBACK, APPEND, and CONNECT commands. A sequence of these commands can reduce connected systems to a single equivalent system. This is useful for input-output control design, state-space control design, and system response analysis.
### TABLE B.7  MATRIXx Capabilities: System Identification, Signal Processing and Data Analysis Capabilities

<table>
<thead>
<tr>
<th>Data Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time-History Plots</td>
</tr>
<tr>
<td>Multichannel Cross-Plots</td>
</tr>
<tr>
<td>Scatter Plots</td>
</tr>
<tr>
<td>Frequency Plots</td>
</tr>
<tr>
<td>Histograms</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Transformations and Spectral Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplexing/Demultiplexing</td>
</tr>
<tr>
<td>Detrending</td>
</tr>
<tr>
<td>Censoring</td>
</tr>
<tr>
<td>Digital Filtering</td>
</tr>
<tr>
<td>Discrete Fourier Transform</td>
</tr>
<tr>
<td>Inverse Fourier Transform</td>
</tr>
<tr>
<td>Autocorrelation</td>
</tr>
<tr>
<td>Cross Correlation</td>
</tr>
<tr>
<td>Autospectrum</td>
</tr>
<tr>
<td>Cross Spectrum</td>
</tr>
<tr>
<td>Decimation and Interpolation</td>
</tr>
<tr>
<td>Maximum Entropy Spectrum Estimation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step-Wise Regression and Model Building</td>
</tr>
<tr>
<td>Maximum Likelihood Identification of State-Space Models and Nonlinear Models</td>
</tr>
<tr>
<td>(generated by SYSTEM_BUILD)</td>
</tr>
<tr>
<td>Recursive Maximum Likelihood Identification</td>
</tr>
<tr>
<td>Extended Kalman Filter Algorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Filter Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window-Based Methods</td>
</tr>
<tr>
<td>Wiener Filter</td>
</tr>
<tr>
<td>REMEZ Exchange Algorithm for Finite Impulse Response Filters</td>
</tr>
<tr>
<td>Elliptic, Chebyshev, Butterworth Infinite Impulse Response Design</td>
</tr>
</tbody>
</table>

### TABLE B.8  Integration Algorithms Available in MATRIXx

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler</td>
<td>Euler</td>
</tr>
<tr>
<td>Rk2</td>
<td>Runge-Kutta (2nd order)</td>
</tr>
<tr>
<td>Rk4</td>
<td>Runge-Kutta (4th order)</td>
</tr>
<tr>
<td>Kutta-Merson</td>
<td>(fixed step)</td>
</tr>
<tr>
<td>Kutta-Merson</td>
<td>(variable step)</td>
</tr>
<tr>
<td>DASSL</td>
<td>implicit stiff predictor-corrector</td>
</tr>
</tbody>
</table>
Figure B.11 shows a brief summary of these commands. These commands can be used in succession for a complex system, and once it is reduced to a simple form prior to analysis, simulation or design. Consider the following example:

**CAD Example B.3**

Consider a complex system of Figure B.12.

\[
\begin{align*}
\text{<> } S1 &= \{-1 \ 1; 1 0\}; \text{NS1} = 1 & \text{Define SISO system } 1 = \frac{1}{s+1} \\
\text{<> } S2 &= \{0 \ 1 0; -1 \ -0.2 \ 1; 1 0 \ 0\}; \text{NS2} = 2 & \text{Define MIMO system 2 (Oscillator – 1 input, 2 outputs)} \\
\text{<> } [S,NS] &= \text{APPEND}(S1,NS1,S2,NS2); & \text{Append systems 1 and 2} \\
\text{<> } \text{FEEDBK} &= \{-1 0 0; 0 \ -3 \ -2\}; & \text{Define feedback gain matrix} \\
\text{<> } [S,NS] &= \text{CONNECT}(S,NS,\text{FEEDBK},[\text{GAIN}]): & \text{Connect FEEDBACK around system.} \\
\text{<> } [A,B,C,D] &= \text{SPLIT}(S,NS); & \text{Extract new system components.}
\end{align*}
\]

The next example illustrates the use of LSIM to obtain time response of a system to a general input. The basic format is given by:

\[
[T,Y] = \text{LSIM}(S,NS,U,\text{DELTAT},X0)
\]

where \(U\) is the input array to be provided by the user, \(X0\) is the initial condition vector, \(\text{DELTAT}\) is the time increment between points in the \(U\) array, \(T\), and \(Y\) are the time vector and output response matrix, that is, if the system has \(r\) outputs, then \(Y\) is described by:

\[
y = \begin{bmatrix} y_1 & y_2 & \cdots & y_r \end{bmatrix} \quad \text{1st time point} \\
\vdots & \vdots & \ddots & \vdots \\
\text{last time point} 
\]

**CAD Example B.4**

\[
\begin{align*}
\text{<> } S &= \{0,1,0; -1,-.2,1; 1,0,0\}; \text{NS} = 2; & \text{Define second order system.} \\
\text{<> } \text{RAND}('\text{NORMAL}'); \\
\text{<> } U &= \text{RAND}(100,1); & \text{Generate white noise input.} \\
\text{<> } DT &= 0.2; \\
\text{<> } [T,Y] &= \text{LSIM}(S,NS,U,DT); & \text{Calculate response to input } U. \\
\text{<> } \text{PLOT}(T,\{U,Y\}) & \text{Plot the input and the output.}
\end{align*}
\]

Figure B.13 shows the plots of input and output.

The next example utilizes the inverted pendulum problem of CAD Ex. B.5 and places its closed-loop poles using the command POLEPLACE for SISO systems.

**CAD Example B.5**

\[
\begin{align*}
\text{<> } & // \text{Pole placement for an Inverted pendulum problem} \\
\text{<> } & // \text{Kwakernaak, H. and Sivan, R. "Linear Optimal Control Systems". Wiley, New York, 1972.}
\end{align*}
\]
**Matrix $x$'s block connection commands for multivariable systems.**

**Series**

\[ [S, NS] = \text{series}(S_1, NS_1, S_2, NS_2) \]

**Parallel**

\[ [S, NS] = \text{parallel}(S_1, NS_1, S_2, NS_2) \]

**Feedback**

\[ [S, NS] = \text{feedback}(S_1, NS_1, S_2, NS_2) \]

1. General
2. Constant-gain feedback $S_2$
3. Unity feedback

\[
\begin{pmatrix}
A_1 & 0 & B_1 \\
B_2 + C_1 & A_2 & B_2 + D_1 \\
D_2 + C_1 & C_2 & D_2 + D_1
\end{pmatrix}
\]

\[
\begin{pmatrix}
A_1 & 0 & 0 \\
0 & A_2 & 0 \\
C_1 & C_2 & D_1 & D_2
\end{pmatrix}
\]
**Command**

**FORMAT**

\[
\text{AFeedback} \quad [S,NS] = \text{AFeedback}(S1,NS1,S2,NS2)
\]

\[
[S,NS] = \text{AFeedback}(S1,NS1,S2)
\]

\[
[S,NS] = \text{AFeedback}(S1,NS1)
\]

**Diagram**

1. General
2. Constant-gain feedback (S2)
3. Unity feedback

**Function**

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
y_1 \\
y_2
\end{pmatrix} =
\begin{pmatrix}
S
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
y_1 \\
y_2
\end{pmatrix}
\]

(d) Feedback—augmented input/output.

The AFeedback command proceeds a closed-loop configuration with augmented inputs and outputs as shown in this diagram. It is identical to the Feedback command in both format and function.

**APPEND**

\[
[S,NS] = \text{APPEND}(S1,NS1,S2,NS2)
\]

(e) Parallel—append of two systems.

The APPEND command appends two state-space systems \([S1,NS1]\) and \([S2,NS2]\) in a form suitable for interconnecting with the CONNECT command. A number of systems can be appended by appending two at a time.
CONNECT \[ [S, NS] = \text{CONNECT}(S1, NS1, K, M, N) \]
\[ [S, NS] = \text{CONNECT}(S1, NS1, K, M) \]
\[ [S, NS] = \text{CONNECT}(S1, NS1, K) \]

1. General
2. unity output gain
3. unity input and output gains

\[ S = \begin{pmatrix} A1 + B1 * V * K * C1 & B1 * V * M \\ N * W * C1 & N * D1 * V * M \end{pmatrix} \]

where \[ V = (I - K * D1)^{-1} \]
\[ W = (I - D1 * K)^{-1} \]

(f) Feedback with unity gains.
Figure B.12  Block diagram of system of CAD Example B.3.

\[
\begin{bmatrix}
\mathbf{A} = [0 & 1 & 0 & 0; & 0 & -1 & 0 & 0; & 0 & 0 & 1; & -11.65 & 0 & 11.65 & 0]; \\
\mathbf{B} = [0 & 1 & 0 & 0]; \\
\end{bmatrix}
\]

// Desired poles are \( -5 \pm j8.66 \) and \( -8.66 \pm j5 \)

\[
\text{POLES} = [-5 + 8.88*\text{J}, -5 - 8.88*\text{J}];
\]

\[
\text{KC} = \text{POLEPLACE} (\mathbf{A}, \mathbf{B}, \text{POLES})
\]

\[
\begin{bmatrix}
\text{KC} =
1.0D+03 * \\
0.3848 & 0.0263 & -1.2431 & -0.2618
\end{bmatrix}
\]

The next example uses the inverted pendulum problem of CAD Ex. B.7 and designs a Kalman filter for it.

**CAD Example B.6**

Consider the inverted pendulum system discussed and assume that the measurement equation is

\[
\begin{bmatrix}
\text{u} \\
\text{y}
\end{bmatrix}
\]

Figure B.13  Input/output response for CAD Example B.4.
\[ y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1.188 & 0 & 1.188 & 0 \end{pmatrix} x = Cx \]

We further assume that the only input disturbance is a white-noise force applied to the cart of intensity $0.5 N^2 \cdot \text{sec}$ and that there is white noise on the displacement and angle sensors of intensity $(2 \times 10^{-6} m^2 \cdot \text{sec})$ and $(6 \times 10^{-6} \text{ rad}^2 \cdot \text{sec})$, respectively. The noise matrices for the Kalman filter design are thus entered as $Q_{xx}$. The $C$ matrix is entered as

\[
\begin{eqnarray*}
&\text{l} & \text{0} & \text{0} & \text{0} \\
-1.188 & \text{0} & \text{1.188} & \text{0} \\
\end{eqnarray*}
\]

\[
\begin{array}{ccc}
\text{C} &=& \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1.188 & 0 & 1.188 & 0 \end{bmatrix} \\
\text{QXX} &=& \text{DIAG}([0 \ 0.5 \ 0 \ 0]) \\
\text{QYY} &=& \text{DIAG}([2E-6 \ 6E-6]) \\
\text{EVAL,KE} &=& \text{ESTIMATOR}(A, C, QXX, QYY) \\
\text{KE} &=& \\
24.8747 & -7.5703 \\
395.3388 & -148.8699 \\
5.7578 & 5.0467 \\
9.2047 & 18.0480 \\
\text{EVAL} &=& \\
-17.4711 & +17.3498i \\
-17.4711 & -17.3498i \\
-2.9607 & +0.9178i \\
-2.9607 & -0.9178i \\
\end{array}
\]

**System build—Interactive modeling.** The interactive model building facility called SYSTEM_BUILD is a tool for building models of complex systems for use in simulation, control design, and trade-off studies. The user can develop multi-input/multi-output (MIMO) system models of individual parts of the system. Transfer function descriptions can be combined with nonlinear functions and state-space models. It is also possible to connect an externally defined FORTRAN module to models defined in SYSTEM_BUILD which can be linearized and simulated with arbitrary inputs. Modules or parts can be changed or replaced without recompiling and relinking FORTRAN code.

A hierarchical structure allows models to be developed "top-down" or "bottom-up." In the top-down approach, the designer specifies an overall system in terms of its major subsystems. Each major subsystem can be defined as an interacting interconnection of lower level subsystems. The lowest level subsystems are finally specified using basic elements, which might consist of nonlinearities, table look-ups, transfer functions, state-space models, and summing junctions. Nonlinearities can include saturation, absolute values, hysteresis, general piecewise linear functions—quantization, and general algebraic nonlinearities. Transfer functions can be written as numerator/denominator polynomial coefficients, zeros/poles, or natural frequencies and damping ratios.

In the bottom-up building approach, the lowest subsystem models are developed first. Major subsystem and complete system models may then be assembled from
lower-level system models. The basic building blocks available in SYSTEM_BUILD are shown in Table B.9.

An example, taken from Version 6.0 of MATRIXx’s User’s Guide would give a good walk-through approach for a nonlinear system. Figure B.14 shows a simple plant model of a nonlinear system. This system serves to illustrate SYSTEM_BUILD for the construction of the simple plant model shown in this figure. The plant consists of a linear state-space system preceded by an actuator. The actuator consists of a first-order lag followed by a deadband nonlinearity. The plant will be represented in SYSTEM_BUILD by a super-block containing two blocks:

1. A super-block ACT which contains the model of the actuator,
2. A dynamic system block S which contains the state-space system.

Figure B.15 shows the SYSTEM_BUILD block diagram for the plant.

The state-space to be modeled is given by the following equations:

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -4 & -0.04 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -16 & -0.16 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} u \quad y$$

$$= \begin{pmatrix} 3.2 & 0 & 3.2 & 0 \\ 0 & 3.2 & 0 & 3.2 \end{pmatrix} x$$

This system consists of two lightly damped, second-order poles with a single input and two outputs. The first output is position; the second, velocity. Recall that
MATRX uses a "system" matrix to store the $A$, $B$, $C$, and $D$ matrices that make up a linear state-space system, that is,

$$\dot{x} = Ax + Bu \quad \text{and} \quad y = Cx + Du$$

the system matrix is defined as

$$S = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

Now, we formally present the SYSTEM_BUILD treatment of this system.

**CAD Example B.7**

$$\langle\rangle S = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -4 & -0.04 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -16 & -0.16 & 1 \\ 3.2 & 0 & 3.2 & 0 & 0 \\ 0 & 3.2 & 0 & 3.2 & 0 \end{pmatrix};$$

$$\langle\rangle \text{ BUILD}$$

$$\text{<<< SYSTEM_BUILD >>>}$$

<table>
<thead>
<tr>
<th>1</th>
<th>Catalog</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Delete Super-Block</td>
</tr>
<tr>
<td>7</td>
<td>Real-Time Tools</td>
</tr>
</tbody>
</table>

Super-Block name: PLANT

Editing new super-block PLANT

Sample Period (0=Cont.): 0

$$\text{<<< DESCRIBE BLOCKS >>>}$$

<table>
<thead>
<tr>
<th>1</th>
<th>Define Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Copy - Paste Block</td>
</tr>
</tbody>
</table>

Location: 3

Inserting new block

The plant is continuous so 0 is entered. To define a block in the top right of the screen, use location 3 by selecting the "Define Block" option and enter 3 at the location prompt.

$$\text{<<< DEFINE BLOCKS >>>}$$

<table>
<thead>
<tr>
<th>2</th>
<th>Remove Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Connect Blocks</td>
</tr>
</tbody>
</table>

When the $$\text{<<< TYPE OF BLOCK >>>}$$ menu appears, select "Dynamic Systems," bringing you to the menu where "State-Space System" can be selected.
### Type of Block

<table>
<thead>
<tr>
<th>1</th>
<th>Next Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Gain Block</td>
</tr>
<tr>
<td>3</td>
<td>Super-Block</td>
</tr>
<tr>
<td>4</td>
<td>Algebraic Equations</td>
</tr>
<tr>
<td>5</td>
<td>Piece-Wise Linear</td>
</tr>
<tr>
<td>6</td>
<td>Dynamic Systems</td>
</tr>
<tr>
<td>7</td>
<td>Trig Functions</td>
</tr>
<tr>
<td>8</td>
<td>User Code Block</td>
</tr>
</tbody>
</table>

### Dynamic Systems

<table>
<thead>
<tr>
<th>1</th>
<th>Next Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Nth Order Integrator</td>
</tr>
<tr>
<td>3</td>
<td>State-Space System</td>
</tr>
<tr>
<td>4</td>
<td>Num - Den Coeffs.</td>
</tr>
<tr>
<td>5</td>
<td>Gain Zeros - Poles</td>
</tr>
<tr>
<td>6</td>
<td>Gain Damps - Freqs</td>
</tr>
<tr>
<td>7</td>
<td>Hysteresis Block</td>
</tr>
<tr>
<td>8</td>
<td>Time Delay: exp(-kTs)</td>
</tr>
</tbody>
</table>

When the **<<<< SPECIFICATION >>>** menu appears, select “Block Name.” At the prompt, enter S, as the name (no relationship to the stack variable S). Only super-blocks are required to be named. All other blocks default to blank.

### Specifications

<table>
<thead>
<tr>
<th>1</th>
<th>Block Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Number of Inputs</td>
</tr>
<tr>
<td>3</td>
<td>Number of Outputs</td>
</tr>
<tr>
<td>4</td>
<td>Number of States</td>
</tr>
<tr>
<td>5</td>
<td>Parameter Entry</td>
</tr>
</tbody>
</table>

**Block name:** S

The state-space system being entered has 1 input, 2 outputs, and 4 states, therefore the number of outputs and the number of states must be specified. Select “Number of Outputs” and respond with 2. Then select “Number of States” and answer the prompt with 4.

### Specifications

<table>
<thead>
<tr>
<th>1</th>
<th>Block Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Number of Inputs</td>
</tr>
<tr>
<td>3</td>
<td>Number of Outputs</td>
</tr>
<tr>
<td>4</td>
<td>Number of States</td>
</tr>
<tr>
<td>5</td>
<td>Parameter Entry</td>
</tr>
</tbody>
</table>

**Number of outputs:** 2

### Specifications

<table>
<thead>
<tr>
<th>1</th>
<th>Block Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Number of Inputs</td>
</tr>
<tr>
<td>3</td>
<td>Number of Outputs</td>
</tr>
<tr>
<td>4</td>
<td>Number of States</td>
</tr>
<tr>
<td>5</td>
<td>Parameter Entry</td>
</tr>
</tbody>
</table>

**Number of States:** 4

Leave the **<<<< SPECIFICATIONS >>>** menu by selecting “Parameter Entry.” SYSTEM_BUILD expects a system matrix for state-space systems. Respond to the prompt with the stack variable name, S. For dynamic systems, the initial states are zero or not. Answering N to the prompt causes SYSTEM_BUILD to prompt for initial states.

### Specifications

<table>
<thead>
<tr>
<th>1</th>
<th>Block Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Number of Inputs</td>
</tr>
<tr>
<td>3</td>
<td>Number of Outputs</td>
</tr>
<tr>
<td>4</td>
<td>Number of States</td>
</tr>
<tr>
<td>5</td>
<td>Parameter Entry</td>
</tr>
</tbody>
</table>
State-space matrix: S
Zero initial states (Y/N)? Y
You are now returned to the <<< DESCRIBE BLOCKS >>> menu. To define another block in location 2, select “Define Block.”

<<< TYPE OF BLOCK >>>
1 Define Block
4 Copy - Paste Block

Location : 2
When the <<< TYPE OF BLOCK >>> menu appears, select “Super-Block.”

<<< SPECIFICATIONS >>>
1 Block Name
4 Number of States

This brings us to the <<< SPECIFICATIONS >>> menu. All Super-Blocks must be named. Select “Block Name,” and enter ACT.

Block name : ACT

Until they are analyzed, super-blocks have zero number of states associated with them. However, they do have an associated number of inputs and outputs. These must be specified if different than the default of one (1). Because models can be built from bottom-up or top-down, ACT need not exist to be included in PLANT. The existence of ACT will only be resolved when PLANT, or a super-block containing PLANT, is analyzed. The number of inputs and outputs must be specified even if ACT already exists, as SYSTEM_BUILD does not check the catalog when a super-block is nested with the editor. Selecting “Parameter Entry” completes the inclusion of the super-block, and returns you to the <<< DESCRIBE BLOCKS >>> menu.

<<< SPECIFICATIONS >>>
1 Block Name
4 Number of States

Selecting the “Connect Blocks” option brings up the <<< CONNECT BLOCKS >>> menu.

<<< DESCRIBE BLOCKS >>>
1 Define Block
4 Copy - Paste Block

<<< CONNECT
1 Define Block
4 Copy - Paste Block

You are now returned to the <<< CONNECT BLOCKS >>> menu. To define another block in location 2, select “Define Block.”
A single external input is connected to the ACT block by selecting "External Input," specifying the length of the input vector as one (1), and the destination of the input as location 2.

Ext input vector dim (0) : 1
To block location : 2
Connection complete

The output of ACT is connected to the state-space system block, S, by selecting "Internal Path," and specifying the origin as block 2 and the destination as block 3.

From block location : 2
To block location : 3
Connection complete

The last connection required is to the output. Select "External Output." The state-space system has two outputs, both of which need to be external outputs. The length of the output vector is 2. Because the super-block has two outputs, a straight parallel connection is possible. When asked if the connection is simple, respond affirmatively (Y).

Ext output vector dim (0) : 2
From block location : 3
Is this connection simple (Y/N) ? Y
Connection complete

The definition of the super-block plant is now complete. To save the super-block and return to the top menu, type TOP.
The super-block, ACT, must now be defined. Select "Edit Super-Block," enter the name ACT, and define the sampling period as 0.

Super-block name : ACT
Editing new super-block ACT
Sample Period (0=Cont.) : 0

A block is defined at location 2.

This block is a dynamic system:

in NUmerator-DENominator form:

The block name is specified as LAG

The block is SISO (single-input/single-output), with a single state so all defaults apply. You can proceed directly to "Parameter Entry." SYSTEM_BUILD prompts for the order of the numerator (highest power), which is zero, and then prompts for the denominator coefficients, entered in order of decreasing powers of s. The initial states are again specified as zero (0).
A piece-wise linear block is then defined at location 3.

A deadband location is chosen:

Defaults apply (The number of states is not relevant, as there are no dynamics). Proceed directly to "Parameter Entry." The deadband is specified as 0.2 (symmetric about 0).
The blocks can now be connected. The super-block will have a single-input and a single-output. There are only three connections required, each of which is 1-to-1.

<<< DESCRIBE BLOCKS >>>
1 Define Block
4 Copy - Paste Block

<<< CONNECT BLOCKS >>>
1 Internal Path
4 Examine Connection

<<< INPUT >>>
1 Internal Path
2 External Input
4 Examine Connection
5 Disconnect Blocks
6 Describe Blocks

Ext input vector dim (0) : 1
To block location : 2
Connection complete

<<< CONNECT BLOCKS >>>
1 Internal Path
4 Examine Connection

<<< OUTPUT >>>
1 Internal Path
2 External Input
4 Examine Connection
5 Disconnect Blocks
6 Describe Blocks

Ext output vector dim (0) : 1
From block location : 3
Connection complete

<<< CONNECT BLOCKS >>>
1 Internal Path
4 Examine Connection

<<< INTERNAL >>>
1 Internal Path
2 External Input
4 Examine Connection
5 Disconnect Blocks
6 Describe Blocks

From block location : 2
To block location : 3
Connection complete

You can return to the top menu. All piece in PLANT have been defined, so it is ready for analysis.
"Analyze" is selected for the super-block PLANT.
Super-block name: PLANT
Super-Block Reference Map
   PLANT
   ACT
All super-blocks identified
System Built with 0 error(s) and 0 warning(s).
Use SIN('IALG') to set the integration algorithm.

SYSTEM_BUILD creates the connected model, displaying a reference map in the process. Errors and inconsistencies in the model are flagged. You are returned to the MATRIX command level.

<> [SL,NSL] = LIN(1)

NSL =
5.

SL =

\[
\begin{bmatrix}
-5.000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 \\
0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0000 \\
4.0000 & -4.0000 & -0.0400 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 \\
4.9000 & 0.0000 & 0.0000 & -16.0000 & -0.1600 & 0.0000 \\
0.0000 & 3.2000 & 0.0000 & 3.2000 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 3.2000 & 0.0000 & 3.2000 & 0.0000 \\
\end{bmatrix}
\]

SL is the linearized system matrix, NSL is the number of states in the linear model, and the argument on LIN is the state and control perturbation to be used in the numerical linearization.

The Bode diagram for the system is found by typing:

<> BODE(SL,NSL,0,1,10)

Here the frequency range is from 0.1 to 10 rads/sec.

The system response to a 1 Hz. sinusoid can be simulated and plotted using the following commands:
<< TIME = [0:.1:10]';US = SIN(2*PI*TIME);
<< Y = SIM(TIME,US);
// Continuous outputs are solved explicitly.
// Continuous states are solved explicitly.
// Using Variable Step Kutta-Merson

<< PLOT(TIME,Y)

Figure B.16 shows the Bode plot of the linearized system.

Shah, et al. (1985) have presented several other examples of both MATRIXx and SYSTEM-BUILD. Other features of MATRIXx such as adding user's own "macros" or "real-time" interface are possible, however, will not be covered for lack of space.

Within MATRIXx, the user can program its own code via four programming tools. These are (a) macros, (b) command files, (c) user routines, and (d) functions. Each scheme has its advantages and disadvantages from programming efficiency point of view. Each scheme will be briefly defined with more details on functions followed by an example.
Macros. Macro is a sequence of MATRIXx operations which can be executed with a single command. A macro is stored as a text storing within MATRIXx. Note that macros should be saved once they have been created, say through MATRIXx’s editor, before leaving MATRIXx.

Command File. Command files contain sequences of MATRIXx commands that are usually longer than those used in macros. Command files are saved as data files and can be used in different sessions. More important, command files can be accessed by many users, while macros are defined only for the user who defined it.

User Routines. In MATRIXx, as in its original predecessor MATLAB, a "user" subroutine can be written in FORTRAN to perform a specific task which is not supported by MATRIXx.

Functions. Functions have three main advantages for the first two schemes.

1. Functions allow parameter passing, hence can be used like any other MATRIXx command.
2. Functions use local variables, that is, a function defined by a user can be utilized by another by simply changing the local variables.
3. Functions will remain an integral part of MATRIXx, just like "m" files in PRO-MATLAB.

The general form of MATRIXx function is given by

```c
//{output1, output2, ...} = FUNNAME(Input1, Input2, ...)
List of MATRIXx Commands
RETF
```

Next, a CAD example uses MATRIXx’ function which was created to generate an \( N \times N \) Hilbert matrix.

CAD Example B.8

Below is a function to generate an \( N \times N \) Hilbert matrix.

```c
//A = HILBM(N)
For I = 1 : N,

A(I,J) = 1/(I + J - 1);

RETF
```

...
<> N = 3;
<> b = HILBM(N);
<> b

B =

\[
\begin{pmatrix}
1.0000 & 0.5000 & 0.3333 \\
0.5000 & 0.3333 & 0.2500 \\
0.3333 & 0.2500 & 0.2000
\end{pmatrix}
\]

Before leaving our discussions on MATRIXx, a reminder that MATRIXx constitutes one of the more mature CAD environments for control systems and the full details on it can be best obtained by consulting the manual (ISI, 1988) and/or actually using it on a terminal.

**B.3.3 CONTROL.lab**

CONTROL.lab is another MATLAB-based CACSD package which was developed at CAD Laboratory for Systems and Robotics, University of New Mexico by Jamshidi et al. (1986). In many respects, CONTROL.lab is similar to CTRL-C and MATRIXx, therefore, the language will be briefly described and a few CAD examples will be given instead.

CONTROL.lab is an interactive computer-aided language that serves as a convenient media for computations involving linear multivariable systems and in some cases nonlinear systems. Aside from the matrix analysis capabilities of CONTROL.lab, its system analysis capabilities range from standard stability tests (Routh, Jury, Lyapunov, etc.), and analytical solutions of linear systems \([e^{AT}, A^k, (sI - A)^{-1}, C(sI - A)^{-1}B + D]\), controllability/observability tests, to simulation and time response of continuous-time and discrete-time systems. The design capabilities of CONTROL.lab range from standard pole placement schemes through state and output feedback through P, PD, PI, and PID, the solutions of algebraic and differential matrix Riccati equations on to the optimal linear quadratic problems. One of the stronger capabilities of CONTROL.lab is in the area of estimation and filtering. In this area, several primitives exist to design Kalman filters and state estimators (observers).

The package has a dedicated graphics interface to Tektronix’s PLOT 10 IGL (Tektronix, 1982) and would allow the use of more popular terminals such as DEC’s VT240 and Tektronix’s TEK 4010, to name two. A pictorial categorization of CONTROL.lab is shown in Fig. B.17. These various classes of primitives are described in its User’s Guide (Jamshidi and Schotik, 1987).
CAD Example B.9

In this example, we consider the numerical inversion of a transfer function,

\[ X(s) = \frac{b_n s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \]

where \( a_i \) and \( b_i \) are real constant coefficients and \( n \) is a positive integer. The differential equation corresponding to this transfer function is

\[ x^{(n)}(t) + a_{n-1}x^{(n-1)}(t) + \cdots + a_1 \dot{x}(t) + a_0 x(t) = 0 \]

with initial conditions

\[
\begin{align*}
x(0) &= b_{n-1} \\
\dot{x}(0) &= b_{n-2} - a_{n-1}x(0) \\
\ddot{x}(0) &= b_{n-3} - a_{n-1}\dot{x}(0) - a_{n-2}x(0) \\
&\vdots \\
x^{(n-1)}(0) &= b_o - a_{n-1}x^{(n-2)}(0) - \cdots - a_1 x(0)
\end{align*}
\]
In state form, the transfer function can be written by defining \( x = (x_1, x_2, \cdots, x_n)^T = (x, \dot{x}, \ddot{x}, \cdots, \dddot{x}^{(n-1)})^T \), that is,

\[
\dot{x} = Ax = \begin{bmatrix}
0 & 1 & 0 & \cdots & \cdots & 0 \\
0 & 0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 0 & 1 & \cdots & 0 \\
- a_0 & - a_1 & \cdots & \cdots & \cdots & - a_{n-1}
\end{bmatrix} x
\]

with initial state \( x(0) \) given by Eq. (B.1). As an example, consider a third-order transfer function,

\[
X(s) = \frac{s^2 + s}{s^3 + 5s^2 + 5.25s + 5}
\]

with initial state

\[
\begin{align*}
    x_1(0) &= x(0) = b_{n-2} = b_2 = 1 \\
    x_2(0) &= \dot{x}(0) = b_1 - a_2 x(0) = -4 \\
    x_3(0) &= \ddot{x}(0) = b_0 - a_2 \dot{x}(0) - a_1 x(0) = 14.75
\end{align*}
\]

and state form

\[
\dot{x} = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-5 & -5.25 & -5
\end{bmatrix} x \quad \text{(B.2)}
\]

To find the inverse Laplace transform of Eq. (B.2) on CONTROL.lab, one can use CSCS for the following system,

\[
A = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
-5 & -5.25 & -5
\end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad C = (1 \ 0 \ 0), \quad D = (0).
\]

The CONTROL.lab session is given by

\[
A = <0 \ 1 \ 0; \ 0 \ 0 \ 1; \ -5 \ -5.25 \ -5>;
\]

\[
<> b = <0 \ 0 \ 0>^t; \\
<> c = <1 \ 0 \ 0>^t; \\
<> d = 0; \\
<> e = <0 \ 5.1 \ 0.001>^t; \\
<> xo = <1 \ -4 \ 14.75>^t; \\
<> CSCS {a,b,c,d,xo};
\]

The resulting function \( x(t) \) versus time is given in Fig. B.18.
CAD Example B.10

In this example, a nonlinear system, shown in Fig. B.19, is used to simulate using CONTROL. lab. For the system shown, let \( r(t) = 3.5 \ u(t) \), select final time to be 4.0 with a step size of 0.08 second and obtain the print-plot of the output \( c(t) \).

Let the nonlinear block be represented by \( N \) and proceed to find an observable companion form for the system such that the desired output is one of state variables instead of a combination of several state variables.

\[
\frac{C(s)}{R(s)} = \frac{6N}{s(s + 2)(s + 10)} = \frac{6N}{s + 2 + \frac{6N}{s + 10}} + 1
\]

\[
\frac{6N}{s + 2 + \frac{6N}{s + 10}} = \frac{6N}{s^3 + 12s^2 + 20s + 6N}
\]

\[ C(s)(s^3 + 12s^2 + 20s + 6N) = R(s)(6N) \]
\[ \ddot{c} + 12\dot{c} + 20c + 6Nc = 6Nr \]
\[ \ddot{c} = 6Nr - 6Nc - 12\dot{c} - 20c \]
\[ c = \int \{ -12c + \int [-20c + \int (6Nr - 6Nc)] \} \ dt \ dt \ dt \]

The simulation diagram is shown in Fig. B.20. From this figure, it follows that:

\[
\begin{align*}
\dot{x}_1 &= 6Nr - 6Nx_3 \\
\dot{x}_2 &= x_1 - 20x_3 \\
\dot{x}_3 &= x_2 - 12x_3
\end{align*}
\]

\[
\therefore \quad A = \begin{bmatrix} 0 & 0 & -6N \\ 1 & 0 & -20 \\ 0 & 1 & -12 \end{bmatrix}, \quad B = \begin{bmatrix} 6N \\ 0 \\ 0 \end{bmatrix}, \quad C = [0 \ 0 \ 1]
\]

Using the primitive STEQ, the observable companion state equation derived can be checked.
Figure B.19  A nonlinear system for CAD Example B.10.

Figure B.20  Simulation diagram of a nonlinear system for CAD Example B.10.
Note: 99 is entered in place of the variable 6N.

\[
\begin{align*}
\langle \rangle & \quad p = \langle 99 & 20 & 12 & 1 \rangle; \\
\langle \rangle & \quad q = \langle 99 & 0 & 0 & 0 \rangle; \\
\langle \rangle & \quad r = \langle 3 & 2 \rangle; \\
\langle \rangle & \quad \langle a,b,c \rangle = \text{step}(p,q,r) \\
C & = \\
& \begin{bmatrix}
0 & 0 & 1 \\
\end{bmatrix}

B & = \\
& \begin{bmatrix}
99 \\
0 \\
0 \\
\end{bmatrix}

A & = \\
& \begin{bmatrix}
0 & 0 & -99 \\
1 & 0 & -20 \\
0 & 1 & -12 \\
\end{bmatrix}
\]

Next the routine describing the system is entered:

% cat nl.f
SUBROUTINE G(T,Y,YP)
DIMENSION Y(3), YP(3)
DOUBLE PRECISION T,GAIN,ERROR,R,Y,YP
R = 3.5D0
ERROR = R - Y(3)
IF (ERROR .LT. -1.5) THEN
   GAIN = -45.0D0
ELSE IF (ERROR .GT. 1.5) THEN
   GAIN = 45.0D0
ENDIF
YP(1) = 6.0D0 * GAIN
YP(2) = Y(1) - 20.0D0 * Y(3)
YP(3) = Y(2) - 12.0D0 * Y(3)
RETURN
END

SUBROUTINE G1
RETURN
END

SUBROUTINE G2
RETURN
END

Then the subroutines are compiled and linked with CONTROLAB.

% 77 -o controlab nl.f - lconstr lplot10
Then CONTROLAB is called

```
% controlab

Enter the system order, initial conditions, and solution range and step size

<> n = 3;
<> y = [0 0 0];
<> index = [0 4.0 0.08];
```

Finally, solve the system using the nonlinear simulation primitive "NLSM;"

```
<> <time,vars> = nlsm(n,h,index)
TIME     VARS =
  0.0800   0.     0.     0.     0.1600   21.6000  0.8563  0.0183
  0.2400   43.2000  3.3525  0.1187
  0.3200   64.8000  12.6028  0.3304
  0.4000   86.4000  12.6028  0.6553
  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...
  3.7600   87.6844  31.4213  2.4579
  3.8400  101.3566  34.9213  2.6579
  3.9200  115.5737  39.0002  2.9334
  4.0000  117.4516  43.2453  3.2554
```

A quick plot of $c(t)$ can be obtained by noting that $c(t) = \text{var}(i,3)$, $i = 1, 2, \ldots$. Hence, use a do loop

```
FOR I = 1:51, CT(I) = VARS(I, 3);
```

then plot(time, ct) would give a quick plot shown in Fig. B.21. A more illustrative plot can be obtained using GPLT.

CONTROLab is only available on a VAX environment under VMS. Its biggest shortcoming is the lack of macro editing and extendibility beyond what the original MATLAB can do. Efforts are underway, however, to refine the feature of this CACSD software program.

**B.3.4 PC_MATLAB**

PC_MATLAB represents a complete reprogrammed enhanced version of the original MATLAB written in "C" language. This task was achieved by Little and Moler (1985) of Math Works, Inc. This new software program is designed in a very modular fashion, somewhat even more convenient than extensive packages such as CTRL-C or MATRIXx. It is, therefore, an optimized, second generation MATLAB for MS-DOS personal computers. Its most useful features, in our opinion, are its programmable macros and the fact that most of its macros are transparent to the users as so-called .m files.

PC_MATLAB can be augmented with several other so-called "tool boxes" in
such areas as control, signal processing, identification, and so on. These tool boxes can be obtained afterwards and be "integrated" into a more comprehensive CACSD package. Table B.10 shows a summary of control tool box of PC_MATLAB. It is noted that this MATLAB-based package has essentially all the features of larger CACSD packages. In addition to these commands, PC_MATLAB has four other commands which are helpful in model building process.

These are

- **append**: Append the dynamics of two subsystems
- **connect**: System interconnection
- **parallel**: Form the parallel connection of two systems
- **series**: Cascade two system in series
<table>
<thead>
<tr>
<th>TABLE B.10</th>
<th>Primitives of PC_MATLAB’s Control Tool Box</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MODELING AND CONVERSIONS</strong></td>
<td></td>
</tr>
<tr>
<td>ss2tf</td>
<td>state-space to transfer function conversion</td>
</tr>
<tr>
<td>ss2zp</td>
<td>state-space to zero-pole conversion</td>
</tr>
<tr>
<td>tf2ss</td>
<td>transfer function to state-space conversion</td>
</tr>
<tr>
<td>tf2zp</td>
<td>transfer function to zero-pole conversion</td>
</tr>
<tr>
<td>zp2tf</td>
<td>zero-pole to transfer function conversion</td>
</tr>
<tr>
<td>zp2ss</td>
<td>zero-pole to state-space conversion</td>
</tr>
<tr>
<td>c2d</td>
<td>continuous to discrete-time conversion</td>
</tr>
<tr>
<td>d2c</td>
<td>discrete to continuous-time conversion</td>
</tr>
<tr>
<td>append</td>
<td>append system dynamics</td>
</tr>
<tr>
<td>connect</td>
<td>block diagram modeling</td>
</tr>
<tr>
<td>parallel</td>
<td>parallel system connection</td>
</tr>
<tr>
<td>series</td>
<td>series system connection</td>
</tr>
<tr>
<td><strong>TIME RESPONSE</strong></td>
<td></td>
</tr>
<tr>
<td>impulse</td>
<td>impulse response</td>
</tr>
<tr>
<td>step</td>
<td>step response</td>
</tr>
<tr>
<td>lsim</td>
<td>continuous simulation to arbitrary inputs</td>
</tr>
<tr>
<td>dimpulse</td>
<td>discrete-time unit sample response</td>
</tr>
<tr>
<td>dstep</td>
<td>discrete-time step response</td>
</tr>
<tr>
<td>dlism</td>
<td>discrete simulation to arbitrary inputs</td>
</tr>
<tr>
<td>filter</td>
<td>SISO z-transform simulation</td>
</tr>
<tr>
<td><strong>GAIN SELECTION</strong></td>
<td></td>
</tr>
<tr>
<td>lqr</td>
<td>linear-quadratic regulator design</td>
</tr>
<tr>
<td>lqe</td>
<td>linear-quadratic estimator design</td>
</tr>
<tr>
<td>dlqr</td>
<td>discrete linear-quadratic regulator design</td>
</tr>
<tr>
<td>dlqe</td>
<td>discrete linear-quadratic estimator design</td>
</tr>
<tr>
<td>place</td>
<td>pole placement</td>
</tr>
<tr>
<td>rlocus</td>
<td>root-locus</td>
</tr>
<tr>
<td><strong>FREQUENCY RESPONSE</strong></td>
<td></td>
</tr>
<tr>
<td>bode</td>
<td>Bode plots</td>
</tr>
<tr>
<td>nyquist</td>
<td>Nyquist plots</td>
</tr>
<tr>
<td>dbode</td>
<td>discrete Bode plots</td>
</tr>
<tr>
<td>freq</td>
<td>SISO z-transform frequency response</td>
</tr>
<tr>
<td><strong>UTILITY</strong></td>
<td></td>
</tr>
<tr>
<td>damp</td>
<td>damping factors and natural frequencies</td>
</tr>
<tr>
<td>margin</td>
<td>gain and phase margins</td>
</tr>
<tr>
<td>ctrb</td>
<td>controllability matrix</td>
</tr>
<tr>
<td>obsv</td>
<td>observability matrix</td>
</tr>
<tr>
<td>tzero</td>
<td>transmission zeros</td>
</tr>
<tr>
<td>fixphase</td>
<td>unwrap phase for Bode plots</td>
</tr>
<tr>
<td>ord2</td>
<td>generate A, B, C, D for a second-order system</td>
</tr>
<tr>
<td>ric</td>
<td>continuous Riccati equation residuals</td>
</tr>
<tr>
<td>dric</td>
<td>discrete Riccati equation residuals</td>
</tr>
<tr>
<td>abcdcheck</td>
<td>check the consistency of an (A, B, C, D) set</td>
</tr>
<tr>
<td>nargcheck</td>
<td>check number of .m file arguments</td>
</tr>
</tbody>
</table>
which can be used, in a similar fashion to SYSTEM_BUILD, to simulate systems described in block diagram form.

PC-MATLAB was used extensively in Chaps. 6 and 7. Here, a few other examples will be given to further illustrate the features of PC-MATLAB.

**CAD Example B.11**

In this example, two discrete-time systems will be simulated in PC-MATLAB. The first is a third-order system whose step response is sought. The second is a second-order system whose response to a white noise is determined.

```matlab
>> % First System Simulation
>> Phi = [ 0 1 1 ; -.5 -.5 1 ; 0 0 -.5 ];
>> gama = [.5 .5 ; 0.05 .05 0 ];
>> c = [ 1 0 0 ; 0 1 0 ]; d = 0 * ones (2,2);
>> yd1 = dstep (Phi, gama, c,d,1,20);
>> yd2 = dstop (Phi, gama, c,d,2,20);
>> kt = [ 0 : 1 : 19 ];
>> Plot (kt, yd1, kt, yd2)
>> title ('Step responses of a Discrete - Time Systems y1 & y2'), xlab ('sampling interval')
>> ylabel ('y1 and y2')
>> % Second System Simulation
>> a = [ 0 1 ; -1 -1 ]; b = [ 0 ; 1 ] c = [ 1 0 ];
>> d = 0 ; xo = [ 0 ; 0 ];
>> % Form an arbitrary random input
>> rand ('normal')
>> u = rand (50,1);
>> yn = dssim (a,b,c,d,u,xo);
>> Plot (yn)
>> title ('White Noise Response 2nd Order System')
>> xlabel ('Time'), ylabel ('y(k)')
```

Figure B.22 shows the resulting responses created.

**CAD Example B.12**

In this example, the EXEC file LQ.mtl presented in Sec. B.1 will be re-examined within the framework of PC-MATLAB. As it was mentioned before, PC-MATLAB would come with numerous .m files which utilize the capabilities of the software to solve a new problem. Following is a list of PC-MATLAB’s lqr.m file which effectively performs the same task as the LQ.mtl file.

```matlab
function [k, s] = lqr(a, b, q, r)
%LQR    Linear quadratic regulator design for continuous-time systems.
%       [K,S] = LQR(A,B,Q,R) calculates the optimal feedback gain matrix K
%       such that the feedback law:
%       u = Kx
```
Figure B.22 Two discrete-time systems responses, for CAD Example B.11.
(a) Step responses of a discrete-time system $y_1$ and $y_2$.
(b) White noise response second-order system.
minimizes the cost function:
\[ J = \text{Integral} \{ x'Qx + u'Ru \} \, dt \]
subject to the constraint equation:
\[ \dot{x} = Ax + Bu \]
Also returned is \( S \), the steady-state solution to the associated algebraic Riccati equation:
\[ 0 = SA + A'S - SBR^{-1}B'S + Q \]

```matlab
if nargcheck(4,4,nargin)
    return
endif

if abcdcheck(a,b)
    return
endif

[m,n] = size(a);
[mb,nb] = size(b);
[mq,nq] = size(q);
if (m == mq) | (nb == nq)
    errmsg('A and Q must be the same size')
    return
endif

[mr,nr] = size(r);
if (m == nr) | (nb == nr)
    errmsg('B and R must be consistent')
    return
endif

% Check if q and r are positive definite
chol(q);
chol(r);

[v,d] = eig([a b*r*b';q, -a']); % eigenvectors of Hamiltonian
d = diag(d);
[d,index] = sort(real(c)); % sort on real part of eigenvalue
if ~( (d(n)[0] & (d(n+1)>0 ))
    errmsg('Can’t order eigenvalues')
    return
endif

chi = v(1:n,index(1:n)); % select vectors with negative eigenvalues
lambda = v((n+1) : (2*n),index (1:n));
s = - real (lambda/chi);
k = r*b'*s;
```

In "lqr", functions "abcdcheck" and "nargcheck" check the dimensions of \( (A, B, C, D) \) matrices and check the number of input arguments, respectively. A careful look at the last ten statements of lqr reveals that it is effectively the same as lq.mtl. Now, suppose that one would wish to write a new .m file to solve the linear state regulator problem, find step responses of the open-loop as well as the optimum closed-loop responses, and plot the outputs. This function, called "Design" is given below:
function design (a,b,c,d,q,r,to,dt,tf)
% Function to design a linear regulator
% problem and plot the step responses
[f,k] = lqr(a,b,q,r);
ac = a - b*f;
time = [to : dt : tf];
% open-loop step response
yo = step (a,b,c,d,l,time);
% optimum closed-loop step response
yc = step (ac,b,c,d,l,time);
Plot (time,yo,time,yc)
end

This function is used below for a third-order system.

```matlab
>> a = [0 1 0 ; 0 0 1 ; .4 .5 .8]; b = [0 ; 1 ; 1];
>> c = [0 1 1 ]; q = eye (3); r = 1; d = 0;
>> design (a,b,c,d,q,r,0,0.1,10.)
```

One can now print some values of the intermediate matrices:

```matlab
>> k

k =

1.7643  0.9498  0.0828
0.9498  2.1145 -0.5971
0.0828 -0.5971  2.5808

>> f

f =

1.0326  1.5174  1.9837

>> eig(ac)
an =

-1.9078
-0.3966 + 0.3965i
-0.3966 - 0.3965i

>> eig (a)
an =

-0.2876 + 0.4563i
-0.2876 - 0.4563i
1.3751
```

Since the open-loop system is unstable, the open-loop response was scaled down by $10^{-12}$ to plot the two responses together. The resulting plots are shown in Figure B.23.

### B.3.5 MATLAB Toolboxes and Extensions

Ever since PC-Matlab and PRO-Matlab have been in the CACSD market, many satellite packages to this package have appeared. These packages have been created through a sequence of .m files which have been made either with close collaboration
with the Mathworks, Inc. known as "tool boxes" or by independent teams at universities and research institutions. In this section, a brief introduction is provided for this group of CACSD packages.

**System-ID tool box.** One of the first significant extensions to MATLAB has been the System Identification Tool Box developed by Ljung (1988). This tool box is a collection of .m files which implement the most common and useful parametric and nonparametric system identification methods. The tool box follows closely the theory developed by the author's book (Ljung, 1987). The first edition of the tool box contains eight functions which can create mathematical models based on input/output data. These functions are

1. **ARMAX** Auto regression, moving average with extra input approach
2. **ARX** Autoregressive with extra input
3. **BJ** Box-Jenkins approach
4. **IV** Instrumental variable method for an ARX-model structure
5. **IV4** Estimation of the parameters of an ARX-model structure using a near-optimal four-stage instrumental variable procedure
6. **OE** The output-error model identification method
7. **PEM** Predictor-error method—Estimation of the parameters of a general MISO linear model Structure
8. **SPA** Spectral Analysis of linear discrete-time systems

This tool box has proven to be a very successful addition to the original MATLAB. For example, Jamshidi, et al. (1989) have used it for the identification
of the models for a very complex three-channel multiple mirror telescope testbed system using experimental data gathering for input and output of the system components. Input/output data files were created experimentally and then loaded into MATLAB environment for component model identification.

**Robust control tool box.** Another useful extension to MATLAB has been the Robust Control Tool Box developed by Chiang and Safonov (1988). The tool box would allow the user to design a “robust” multivariable feedback control system based on the concept of the singular-value Bode plot. The following “tools” are included in this MATLAB extension:

1. LQG-based optimal control synthesis including loop transfer recovery and frequency-weighted LQG
2. H₂ and H∞ optimal control synthesis
4. Multivariable digital control system design.

The toolbox contains a number of demonstration examples including the elevon and canard actuator control of a fighter aircraft system and modal control of a large-space structure. This extension of MATLAB has certainly satisfied a need in robust control system design which is a popular research topic in today’s control theory.

The basic commands featured in this toolbox are:

```
H2QG        H₂ optimal control synthesis
HINF and LINF H∞ optimal control synthesis
LQG         Linear quadratic Gaussian optimal control synthesis
LTRU and LTRY LQG loop-transfer recovery optimal control synthesis
YOUULA      LQG control synthesis using Youla Parametrization
```

There are perhaps a few more toolboxes and MATLAB extensions which may be missing here. One such toolbox is the “State-space identification tool” by Milne (1988) which provides MATLAB-like commands to create continuous-time linear-time-invariant models using the maximum likelihood scheme. In the next three sections, three more toolboxes are discussed very briefly.

**Large-scale system tool box.** One of the attempts in creating design-specific tool boxes is underway at the University of New Mexico’s CAD Laboratory for Systems/Robotics for analysis, simulation, and design of large-scale systems which is based on the book by Jamshidi (1983). In this software environment, various issues of large-scale systems such as model reduction, analysis, structural properties, control, and design are incorporated. The basic features of this CACSD environment are similar to LSSPAK/PC which is discussed in some detail in Section B.4. Details regarding this tool box can be obtained from the first author.
The control kit. Another good example of the extendability of MATLAB is the "Control Kit" which has been created by the research team at the University of Sussex through the efforts of Atherton, et al. (Xue, 1991).

The main objective of the Control Kit is to create a menu interface for a classical control course to accompany PC-Matlab for students without the knowledge of this package. Although any menu-driven package is bound to have some inevitable restrictions, the Kit exploits the total flexibility of MATLAB. The theme structure of the control system in the Kit is a SISO system with a plant $G(s)$, controller compensator $C(s)$, and a feedback block $H(s)$. Through the various layers of menu, the users can build their own structure, realize it in both frequency and time domains, analyze it, and try a number of different controller structures. Clearly, every feature of linear system analysis and design of SISO systems that MATLAB supports is also supported by the Control Kit. The Kit can be obtained by contacting the authors (Xue, et al. 1991). Atherton's group has also created a MATLAB-based series of .m files for nonlinear 3180 systems analysis including the describing function method.

Robotics toolbox. Extension of MATLAB is not restricted just to control and identification. Once again at CAD Laboratory for Systems/Robotics of the University of New Mexico, in collaboration with Tampere University of Technology, Finland, a new effort has started to create an environment for robot manipulators. The environment contains a library of robots such as PUMA 560, Adept II, Rhino XR-2, and so on. It handles all the basic problems for robotics such as kinematics (forward and inverse), dynamics (nonlinear and linear), trajectory planning, simulation, analysis, and control. Both simulation as well as real-time facilities will be available. The initial report on this project can be found in Honey (1991) as well as Honey and Jamshidi (1991).

B.4 OTHER DESIGN PACKAGES

In this section, a few CACSD packages that have been developed independently of MATLAB are presented. It should be noted that due to limited space, one can not cover all such packages.

B.4.1 FREDOM—TIMDOM—LSSPAK

The past eight years have been an especially active period for computer-aided design of control systems in the United States and Europe. The use of the computer as a design tool in integrated electronic circuits has become very common in both industry and academic institutions. However, in the area of control system theory, the past years have been very critical. Today, there is hardly any university which does not have access to CAD software for control systems. Toward this goal, Computer-Aided Design Laboratory for Systems/Robotics (CAD LAB) in Electrical and Computer Engineering Department of the University of New Mexico has been active in developing and implementing various CAD software environments for design and analysis.
of linear control systems, robot manipulator control, and simulation. One such CAD software package CONTROL.lab/VAX, was already described briefly. In this section, three other software packages—FREDOM, TIMDOM, and LSSPAK on the HP 9800 series as well as on the IBM PC and its compatibles will be briefly described. These packages are

1. FREDOM, a CAD package for linear classical control systems
2. TIMDOM, a CAD package for linear modern control systems
3. LSSPAK, a CAD package for linear large-scale control systems

**FREDOM.** FREDOM is a FREquency-DOMain CAD package for SISO (single-input single-output) systems described by a pair of transfer functions described by a feed-forward transfer function \( G(s) \), described by (Morel, 1984).

\[
G(s) = \frac{Af(s)}{Bf(s)} = \frac{s^n + a_{m-1}s^{m-1} + \cdots + a_1s + a_0}{s^n + b_{n-1}s^{n-1} + \cdots + b_1s + b_0}
\]

and a feedback transfer function,

\[
H(s) = \frac{Cf(s)}{Df(s)} = \frac{s^p + c_{p-1}s^{p-1} + \cdots + c_1s + c_0}{s^q + d_{q-1}s^{q-1} + \cdots + d_1s + d_0}
\]

or they may be represented in the z-transform domain for discrete-time systems as shown by

\[
G(z) = \frac{Af(z)}{Bf(z)} = \frac{z^n + a_{m-1}z^{m-1} + \cdots + a_1z + a_0}{z^n + b_{n-1}z^{n-1} + \cdots + b_1z + b_0}
\]

\[
H(z) = \frac{Cf(z)}{Df(z)} = \frac{z^p + c_{p-1}z^{p-1} + \cdots + c_1z + c_0}{z^q + d_{q-1}z^{q-1} + \cdots + d_1z + d_0}
\]

where \( m \leq n \) and \( p \leq q \). There is no loss in generality by assuming,

\[a_m = b_n = d_p = d_q = 1\]

since any of these coefficients can be normalized to one.

In sequel, a brief description of the capabilities of FREDOM will be presented.
Figure B.24 A tree structure for FREDOM—TIMDOM/45.

Analysis. The analysis of linear control systems in FREDOM can be performed with several techniques. Figure B.24 shows a tree structure of FREDOM—TIMDOM/45—the version on the HP9845 computer. The structure of FREDOM and TIMDOM on other computers such as the IBM PC/XT may differ to some extent. However, the essential elements of both versions of the packages are the same.

When ANALYS is selected FREDOM/45, a menu with the following options will appear,

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROUTH</td>
<td>Routh-Hurwitz stability criterion</td>
</tr>
<tr>
<td>JURY</td>
<td>Jury-Blanchard stability</td>
</tr>
<tr>
<td>BODENY</td>
<td>Bode-Nyquist plots</td>
</tr>
<tr>
<td>ROOTLC</td>
<td>Root-Locus plot</td>
</tr>
<tr>
<td>SOL</td>
<td>Time domain solution of transfer function</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave frequency analysis</td>
</tr>
</tbody>
</table>
The first command, ROUTH performs a Routh-Hurwitz stability criterion for a SISO system. The second menu item is JURY which will construct the Jury-Blanchard table and, if necessary, the Raible table, for discrete-time system stability analysis. The remaining items are similarly self explanatory.

The final analysis technique is SOL which provides the complete response of the system. This is done by converting $G(s)$ and $H(s)$ into a state equation and appropriate integration and plotting routines are used to simulate the system. These menu items will appear in other versions of FREDOM, for example, FREDOM/PC/XT. However, the name(s) of some commands may be different on the PC version. For example, SOL in FREDOM/45 is called COMSCS in FREDOM/PC.

**Design.** The design of systems in the frequency domain involves the classical approaches to design of SISO systems. When DESIGN in the frequency domain is selected the following menu will appear.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEADLC</td>
<td>Lead, Lag or Lead-Lag Compensation</td>
</tr>
<tr>
<td>ROOTLC</td>
<td>Root-locus Plots</td>
</tr>
<tr>
<td>PARAOP</td>
<td>Parameter Optimization</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave frequency design</td>
</tr>
</tbody>
</table>

Once again, these items are self explanatory.

**Model Reduction.** The methods of model reduction in the frequency domain are restricted to SISO systems except for the matrix continued fraction method. When the user selects MODRED in the frequency domain the following menu will appear.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADE</td>
<td>Pade Approximation</td>
</tr>
<tr>
<td>MOMENT</td>
<td>Moment Matching</td>
</tr>
<tr>
<td>MATCF</td>
<td>Matrix Continued Fraction</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave frequency domain model reduction</td>
</tr>
</tbody>
</table>

The problem of model reduction was described in great detail in Sec. 7.2. It should be noted that both FREDOM/45 and TIMDOM/45 as depicted in Fig. B.24 in extended BASIC source are completely listed in a linear systems book by Jamshidi and Malek-Zavarei (1986).

**TIMDOM.** TIMDOM is a TIMe-DOMain CAD package for MIMO systems described by a quadruple of matrices $(A, B, C, D)$ representing a linear system in state-space form

$$\dot{x} = Ax + Bu, \quad x(0) = x_0$$

$$y = Cx + Du$$
where \( A \) is \( n \times n \) system matrix, \( B \) is \( n \times m \) input matrix, \( C \) is \( r \times n \) output matrix, \( D \) is \( r \times m \) input-output matrix, and \( x, u, \) and \( y \) are \( n \times 1 \) state, \( m \times 1 \) control and \( r \times 1 \) output vectors, respectively and \( x_0 \) is the initial state. In sequel, a brief description of the capabilities of TIMDOM will be presented.

**Analysis.** The analysis of linear control systems in TIMDOM can also be performed with several techniques. When ANALYS command of TIMDOM is selected, the following options would appear.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOL1</td>
<td>Analytical solution of state transition matrix</td>
</tr>
<tr>
<td>SOL2</td>
<td>Numerical solution of the state equation</td>
</tr>
<tr>
<td>LYAP</td>
<td>Lyapunov equation solution</td>
</tr>
<tr>
<td>CON</td>
<td>Controllability check</td>
</tr>
<tr>
<td>OBS</td>
<td>Observability check</td>
</tr>
<tr>
<td>RESMAT</td>
<td>Resolvent-matrix in ( Q(s)/P(s) ) form</td>
</tr>
<tr>
<td>STAB</td>
<td>Stability of the origin for continuous time and discrete-time systems</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave time domain analysis</td>
</tr>
</tbody>
</table>

These commands are fairly self explanatory.

**Design.** On selection of "DESIGN" in the time domain, the user is presented with the following menu:

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POLPLA</td>
<td>Pole placement</td>
</tr>
<tr>
<td>RICATI</td>
<td>Riccati equation solution</td>
</tr>
<tr>
<td>TPBVP</td>
<td>Two-point boundary-value problem</td>
</tr>
<tr>
<td>TRACK</td>
<td>Tracking problem</td>
</tr>
<tr>
<td>LISTRG</td>
<td>Linear state regulator problem</td>
</tr>
<tr>
<td>PARA</td>
<td>Parameter optimization</td>
</tr>
<tr>
<td>NEAROP</td>
<td>Near-optimum design</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave time domain design</td>
</tr>
</tbody>
</table>

The first item POLPLA is concerned with the placement of poles for a closed-loop system, which provides a state feedback controller for a linear time-invariant SISO system. The remaining five commands are self-explanatory. The command NEAROP refers to the case where a controller (say state feedback) is designed for a reduced-order (aggregated) model. This type of design technique was discussed in Chap. 7.

**Model Reduction.** The model reduction schemes of TIMDOM fall along the lines of large-scale systems order reduction schemes (Jamshidi, 1983) "aggregation" and "perturbation."
Some of these techniques were discussed in Chap. 7. When provoking the MODRED command, the following subcommands would typically appear in TIMDOM.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXACT</td>
<td>Exact aggregation</td>
</tr>
<tr>
<td>MODAL</td>
<td>Modal aggregation</td>
</tr>
<tr>
<td>CFAGGR</td>
<td>Continued fraction aggregation</td>
</tr>
<tr>
<td>SEPTIM</td>
<td>Separation of time scales</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave time domain model reduction</td>
</tr>
</tbody>
</table>

Here the first three commands are typical time-domain aggregation schemes, while the last one, SEPTIM corresponds to a perturbation method, whereby a system is checked whether its variables' TIME scales can be SEParated into a "slow" and a "fast" subclass of variables.

Estimation/Filtering. TIMDOM provides two techniques for state estimation and one for filtering. In a typical use of estimation/filtering submenu, called ESTIM, the following options would appear.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FULOBS</td>
<td>Full order observer</td>
</tr>
<tr>
<td>REDOBS</td>
<td>Reduced order observer</td>
</tr>
<tr>
<td>KALMAN</td>
<td>Standard Kalman filter</td>
</tr>
<tr>
<td>EXIT</td>
<td>Leave time domain estimation/filtering</td>
</tr>
</tbody>
</table>

The first item is essentially the design of a state estimator of the Luenberger-type for a linear time-invariant SISO system. The second one is the reduced-order version of it whereby only \((n - r)\), where \(n\) is the number of states and \(r\) is the number of outputs, of the system's state variables are estimated. The last item is the design of a Kalman filter with a zero-mean white noise driven linear time-invariant discrete-time system.

LSSPAK. LSSPAK is a CAD package for modeling and control of linear large-scale systems. The name LSSPAK stems from Large-Scale Systems PAckage. It consists of four main submenus: Linear Algebra (LINALG), Model reduction (MODRED), Analysis (ANALYS), and Design (DESIGN). The theory of large-scale systems was discussed in Chap. 7. A brief description of LSSPAK's structure follows.

Linear Algebra (LINALG). The linear algebra programs supported by LSSPAK are essentially the same as TIMDOM or FREDOM with exceptions of: (a) Cheby—a Chebyshev polynomial curve fitting program, and (b) Genrank—a program to calculate the generic rank of a structured matrix for use in checking the structural controllability and observability of a large-scale linear interconnected system.
Model Reduction (MODRED). Under model reduction, LSSPAK supports both frequency-domain techniques of FREDOM and time-domain methods of TIMDOM. In addition it provides

PADMOD  Pade-Modal method
PADROUT  Pade-Routh method
CHAIN    Chained aggregation
BALANC    Balanced realization

Analysis (ANALYS). In the analysis submenu of LSSPAK, the following options are offered

COMMAND  Description
STAB      Stability of a large-scale system via the Lyapunove method
STRCON    Structural controllability
STROBS    Structural observability
SIMUL     Simulation of a large-scale system

Design (DESIGN). The design of a large-scale system, as seen in Chap. 7, can take on two basic forms: "hierarchical" control which is associated with the decomposed form, i.e. Eq. (7.82) of a large-scale system and "decentralized" control which is associated with the decentralized form, i.e. Eq. (7.104). Here the following options are offered.

COMMAND  Description
GOALCR    Goal coordination algorithm of hierarchical control
INTPRD    Interaction prediction algorithm of hierarchical control
DYNCOM    Dynamic Compensation of decentralized control
ROBDEC    Robust decentralized controller design
DECSTM    Decentralized stabilization via the multilevel method

Throughout this text, several CAD examples on the use of TIMDOM/PC and LSSPAK/PC have been presented. We conclude this section by presenting one more CAD example.

CAD Example B.13

In this example, a third-order system with two inputs and one output is simulated on TIMDOM/PC.

<<COMSCS>> Finds a COMplete Solution of a linear Continuous-time System via 4th order Runge-Kutta with graphical plots
ORDER of the system n = 3 No. of system INPUTS m = 2
No. of system OUTPUTS r = 1
Initial time to = 0 Final time tf = 5 Step size dt = .1
Matrix A
\[
\begin{array}{ccc}
0.000E+00 & 0.100E+01 & 0.100E+01 \\
-0.200E+01 & -0.200E+01 & 0.100E+01 \\
0.000E+00 & 0.000E+00 & -0.200E+01 \\
\end{array}
\]

Matrix B
\[
\begin{array}{ccc}
0.100E+01 & 0.100E+01 \\
0.000E+00 & 0.100E+01 \\
0.100E+01 & 0.000E+00 \\
\end{array}
\]

Matrix C
\[
\begin{array}{ccc}
0.100E+01 & 0.500E+00 & 0.250E+00 \\
\end{array}
\]

Matrix D
\[
\begin{array}{c}
0.000E+00 \\
0.000E+00 \\
\end{array}
\]

Initial STATES:
1   0.000E+00
2   0.000E+00
3   0.000E+00

The output responses of the system for inputs $u = (1 \ 0)^T$ and $u = (0 \ 1)^T$ are shown in Fig. B.25. The output responses were saved in an ASCII file within TIMDOM/PC under COMSCS command. The resulting files were then loaded into PC-MATLAB to be plotted.

**B.4.2 CADACS (KEDDC)**

CADACS is a comprehensive CAD package designed to cover a wide range of control system engineering tasks and related areas. It contains modules for process identification, system analysis, controller design, simulation, and controller implementation based on a broad variety of modern as well as classical approaches for SISO and MIMO systems. This package is regarded as having a user-friendly interface, graphic output and state-of-the-art numerical algorithms. This package is based on the KEDDC framework, developed at Ruhr–University Bochum, Germany (Schmid, 1985). CADACS is available in different configurations and versions. It can easily be tailored and adapted by the user, as it is written in FORTRAN, CADACS can be used in the form of complete main programs or in the form of a comprehensive control engineering library. It provides a system which minimizes engineering and programming resources required for the complete cycle of system identification, control design and design verification.

Table B.11 shows the features of CADACS. The various systems' descriptions, forms, and transformations supported by CADACS are shown in Figure B.26. Various classes of techniques supported by CADACS are briefly described.

**IDENTIFICATION METHODS**

- Approximation in time domain (6 methods)
- Approximation in frequency domain (2 methods)
Figure B.25  Output response of CAD Example B.13.
(a) $u = [1 \ 0]$ and
(b) $u = [0 \ 1]$.

- Approximation by exponential functions
- Approximation by moment methods
- Recursive least squares algorithms
- Correlation and spectral analysis
TABLE B.11 Features of CADACS (KEDDC)

- Simple command-driven dialog combined with questions and answers
- Hierarchy of independent program modules
- Control engineering library
- Reliable numerical algorithms drawn from recent research in numerical analysis
- Device independent graphics output
- Transformation of system representation forms using alternative sets of strategies or numerical algorithms
- Signal analysis and filtering
- Off-line and on-line identification using deterministic and stochastic approach (20 methods)
- Model reduction
- Controller design (15 approaches)
- Parameter optimization
- Observer design (pole placement, Kalman filter)
- Simulation (general and special purpose simulators)
- Adaptive controller design (5 approaches)
- Controller implementation (simulation, real time)
- Standard tasks centralized in 'Manager' programs

- Maximum-likelihood parameter estimation
- Various parameter estimation methods using different numerical methods

CONTROL SYSTEM DESIGN METHODS

- Continuous and discrete time compensators
- Finite settling time methods
- Various pseudocompensator methods
- Optimization of PID-type controllers
- State-feedback controllers with or without PI-action
- Inverse-Nyquist-array technique
- Reduced and full-order observers
- Model reduction in open and closed-loop configurations
- Parameter optimization using sets of performance indices and design criteria
- Kalman filter design

ADAPTIVE CONTROL SYSTEM DESIGN METHODS

- Adaptive PI-controller using periodic test signals
- Adaptive compensator using various parameter estimation methods
- Self-tuners
- Various model reference adaptive control approaches
- Adaptive observers
1. series of deterministic or stochastic input and/or output signals,
2. series of auto- or crosscorrelation values,
3. discrete values of the pulse response,
4. discrete values of the step response,
5. transfer function (matrix) in s-domain,
6. transfer function (matrix) in z-domain,
7. discrete values of frequency responses,
8. matrices for continuous state space,
9. matrices for discrete state space,
10. polynomial matrices.

- Multivariable adaptive controller design
- Adaptive controller simulation

**Simulation.** Standard structures for linear control systems are simulated by a special simulator. The overall system to be simulated may be composed of different subsystems of different representation forms. Each block in this structure may be form 5, 6, 8, 9, or 10. A second type of simulator performs block-oriented continuous-time simulation using elementary CSMP-like blocks. For direct digital control operation (DDC) different special simulators are available. They simulate the DDC environment for testing DDC algorithms in real-time applications. In all cases graphic output is controlled by a Graphics Manager prepared menu.

**Centralized tasks.** Frequently used tasks such as transformations to different representation forms, data base definitions or basic calculations, are separated from
the other level-3 programs and are concentrated in main interactive system handlers, the so-called Managers.

- Signal Manager for handling of signal sequences
- System Manager for handling of systems described by transfer functions or transfer matrices in $s$- or $z$-domain.
- Frequency Manager for handling of frequency responses or spectra in tabular form
- Matrix Manager for handling of coefficient matrices or systems in state space
- Polynomial Matrix Manager for handling of polynomial matrices and related system representations
- Graphics Manager for all graphics operation
- Documentation Manager for real-time signal presentation
- Monitor for central organization problems such as task scheduling

CADACS program structure is shown in Fig. B.27. As seen, it is organized in a four-level hierarchy of independent program modules. All basic functions are contained within the first level. These include all numerical algorithms (e.g., for matrix or polynomial operations). The second level represents the control engineering library containing all relevant specialized algorithms as subroutines which make extensive

![Diagram]

**Figure B.27** Program structure of CADACS (KEDDC).
use of modules of proven numerical software. Level-1 and level-2 subroutines together form the complete library system. This is one form in which CADACS can be used. Modules of the third level are designed for dedicated main tasks, in which a set or a class of methods is grouped together. They are complete interactive or real-time programs which can run as stand-alone programs or which can run supervised by a central monitor which resides in the top level. It manages the housekeeping services for all internal resources and forms the friendly user interface. Any choice of level-3 programs is possible, but with a proper choice from a large set of programs (some are prerequisite) a CADACS system can be tailored according to the user’s requests. Communication between interactive programs is performed by a data base with standard data formats.

CADACS provides a high degree of portability, which results from the interface of CADACS to the operating system and to hardware-dependent functions. The strict separation of interactive and application code makes this possible. For the implementation in another computer environment only these level-1 modules have to be modified. That level contains a number of device dependent functions supporting the user interface. For a minimal configuration, dummy routines are available to replace nonessential functions.

**Interactive operation.** A unified command-driven dialog combined with question-and-answers allows a high degree of user interaction. Each level-3 interactive program has its own set of commands, which can be enabled by supervisory scheduling of the programs using the monitor. An active program prompts with its name. A two-character command initiates a subtask or a local question-and-answer dialog, where values can be entered in free format. Commands are natural, simple, and powerful. The local question-and-answer dialog consists of questions sent by the computer, which ends with ‘?’, where the user has to specify one of some values by typing an alphanumeric character string. If the question ends with ‘?’ only a yes/no answer is requested. Reasonable defaults make this dialog easy to use. Mistyping is detected and the question is repeated. No blank input default values are assumed. The menu of commands or status information can be displayed using standard commands. At any stage the user may get ‘HELP’ information. This is essential for error recovery. The user is provided with a detailed analysis of one’s errors and an explanation about what happened. In addition, some hints will be given on how to proceed. The command interpreter handles both local commands and central commands, and an indication on which central commands can also be tailored by the user. Level-3 programs are interruptible at any stage, and new subtasks from other level-3 programs can be interlaced into the running dialog if program cloning is supported by the operating system. This feature allows a very free and flexible use of CADACS.

**Graphics.** CADACS programs themselves do not generate graphics code. Graphics operations are performed by a central Graphics Manager, which itself is an autonomous package. Data are sent from the application programs to the Graphics Manager via level-1 communication interface. The Graphics Manager is a convenient
special task for formatting data into graphic form readable by the control engineer. The Graphics Manager can be implemented using a graphics processor, or on an intelligent terminal or as an internal graphics task on the same computer. This concept shows a great deal of flexibility in interfacing to different graphics device configurations and integrates diverse parts of CADACS into a unified system. A task-based Graphics Manager is available in CADACS. It incorporates device independent graphics which is based on different standards. All diagrams relevant to control engineering are implemented. The user has the choice of using default diagrams or tailor plots using a keyword-based descriptive language.

The following example illustrates the use of CADACS (KEDDC).

**CAD Example B.14**

To illustrate the user dialogue of the package, a hardcopy of some sections of a longer KEDDC session has been copied here. Below is the original dialogue of this session. Operator input is underlined, command lines are printed in bold letters and command menus are framed. Graphics output is not exactly placed in chronological order with the dialogue. The hardcopy is printed when the image is complete.

In order to understand this demonstration, section numbers are added to the left margin. Section (1) starts with Frequency Management. The command menu is shown which appears in a special window. Principal gain plot is requested as explained using ‘help’ at (2). The principal gains for a system read from file TGM are plotted at (3). At (4) Matrix Management is interlaced to generate a 3D-mesh surface for the system given in state space description. Simulation is interlaced at (6) and then a second Matrix Management at (7) to display the pole zero pattern of the discrete-time system to be controlled. At (8) Simulation is continued. After some configuration dialogue, which is not shown here, the screen shows the results of all sections at (9). The graphic viewpoints are numbered from left to right beginning at the bottom of the screen. The viewpoint number 5 is a small one which is inside of number 3.

(1) KEDDC

KED66: FR
FRM66: ??

**************************Frequency Management**************************

**SYSTEM:**
- RE Read system from file
- G(s),G(z),u(t),y(t),G(jw)
- g(t),h(t),(A,B,C)
- LI List system data
  - 1 WH Write step
  - FREQUENCY TABLE: response on file
- GO Generate frequency table
- EO Edit frequency table
- RO Read frequency table from file
- IO Input frequency table using dialog
- LO List frequency table
- WO Write frequency table on file

**FURTHER CALCULATIONS:**
- PO Popov-frequency response
- IV Inverse frequency response
- WC Magnitude/phase for spec. freq.
- TT additional deadtime
- RA Magnitude or phase margins
- FH Step response from freq. resp.

**CONNECTIONS:**
- S2 Read 2nd system from file
- // Negative feedback connection
- ** Parallel connection
- ** Series connection
FREQUENCY RESPONSE:
FG Calculate frequency response
RF Read frequency response from file
IF Input frequency response using dialog
EF Edit frequency response
LF List frequency response
WF Write frequency response on file
FM Create frequency response matrix on file
ST Status protocol

GRAPHICS:
BO Bode plot complete
BM Bode plot magnitude
BP Bode plot phase
NY Nyquist plot
NI Nichols plot
DH Step response
TR Toggle curve tracking on/off
MP Change no. of dots per dia.
PG Principal gain plot

FRM66: PK
/FR-004 THERE IS NO FREQUENCY TABLE !

(2) FRM66: HE

/FR-004 THERE IS NO FREQUENCY TABLE !
You have initiated an operation which requires a frequency table. This table isn't defined.
Read table from file with RO, or specify table interactively with IO, or generate it automatically with GO. The last one is the most comfortable way if the frequency range is initially unknown.

FRM66: GO

RECOMMENDED FREQUENCY RANGE FROM 1.79416 TO 899.2309
FREQUENCY RANGE FROM, TO = ? 1.1E3

NUMBER OF FREQUENCY VALUES = ? 100

FRM66: PK

FREQUENCY FACTOR = ?
FILENAME = ? TGM: :DA

file=TGM: :DA, type=UE
created using SMGR by SCHMID on 4. 9.1983 at 11.35
TURBOGEN. MODEL 1 MIMO

IT IS A TRANSFER MATRIX
NUMBER OF INPUTS = 2. NUMBER OF OUTPUTS = 2

(3) MAXIMUM PRINCIPAL GAIN IS MAPPED TO VIEWPOINT 4, CURVE 1
MINIMUM PRINCIPAL GAIN IS MAPPED TO VIEWPOINT 4, CURVE 2

(4) FRM66: @@
KEDDC: MM

MMG66: RS
FILENAME = ? VFELDE: :DA

file=VFELDE: :DA, type=MA
created using MMGR by SCHMID on 31.10.1980 at 13.25
TURBOGEN. STATE SPACE MODEL A2Y3
2 BLOCKS, 3 MATRICES IN 1ST BLOCK
BLOCKNO., MATRIXNO. = ? 2
MMG66: MS
DO YOU WANT GRAPHICS ? Y
MATRIX A
+ 0  0  0
0 +  0  0
0 + +  0
+  0  0 +
MATRIX B
+  0
0 +
+ +
+ +
MATRIX C
0  0  1  0
0  0  0  1
(5) MATRICES MESH SURFACE IS MAPPED TO VIEWPOINT 2, CURVE 1
MMG66: EX
(6) FRM66: @@
KEDDC: DG
DIG66: @@
(7) KEDDC: MM
MMG66: RS
FILENAME = ? TZM: :DA
file = TZM: :DA, type = MA
created using MMGR by SCHMID on 9. 9.1983 at 11.46
TURBOGEN, STATE SPACE MODEL A2YA
3 BLOCKS, 3 MATRICES IN 1ST BLOCK
BLOCKNO., MATRIXNO. = ? 2
MMG66: DM
SAMPLING INTERVAL = ? .2
MMG66: EI
DO YOU WANT GRAPHICS 2 Y

-------------------------------------------EIGENVALUES-------------------------------------------
<table>
<thead>
<tr>
<th>NO</th>
<th>REALPART</th>
<th>IMAGINARYPART</th>
<th>S/I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.385317</td>
<td>.285704</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>.385317</td>
<td>-.285704</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>.462243</td>
<td>.308177</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>.462243</td>
<td>-.308177</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>.644391</td>
<td>.250971</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>.644391</td>
<td>-.250971</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2.131015E-02</td>
<td>.437934</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2.131015E-02</td>
<td>-.437934</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>.715286</td>
<td>0.000000</td>
<td></td>
</tr>
</tbody>
</table>
EIGENVALUES ARE MAPPED TO VIEWPORT 5. CURVE 1
MMG66: NU
DO YOU WANT GRAPHICS? Y

-------------------------------ZEROS-------------------------------

<table>
<thead>
<tr>
<th>NO</th>
<th>REALPART</th>
<th>IMAGINARYPART</th>
<th>S/I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-2.03490</td>
<td>0.00000</td>
<td>+</td>
</tr>
<tr>
<td>2</td>
<td>-1.73257</td>
<td>0.00000</td>
<td>+</td>
</tr>
<tr>
<td>3</td>
<td>.770414</td>
<td>0.00000</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>.368101</td>
<td>.463010</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>.368101</td>
<td>-.463010</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>.449659</td>
<td>.360620</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>.449659</td>
<td>-.360620</td>
<td>-</td>
</tr>
</tbody>
</table>

ZEROS ARE MAPPED TO VIEWPORT 5. CURVE 2
MMG66: EX

(8) DIG66: RE
FILENAME = ? TGM: DA
DIG66: SI
SIMULATION STEP SIZE = ? .1
SIMULATION TIME = ? 10
SIGNAL U (1) IS MAPPED TO VIEWPORT 1, CURVE 1
SIGNAL U (2) IS MAPPED TO VIEWPORT 1, CURVE 2
SIGNAL Y (1) IS MAPPED TO VIEWPORT 3, CURVE 1
(9) SIGNAL Y (2) IS MAPPED TO VIEWPORT 3, CURVE 2

Figure B.28 shows some graphical outputs of this CADACS' example.

B.4.3 L-A-S

L-A-S (Linear Algebra and Systems) developed by Bingular and associates (1985, 1987) is a CACSD package which fully supports a full-fledged programming language to construct, test and evaluate various algorithms for analysis and design of control systems. The fundamental concept behind the L-A-S operator is that the L-A-S operator performs some described calculation on input data, indicates errors, if any, and generates output data (results). The output data of operator may be used as input data by any operator to be executed later.

The operators together with the names of input and output data are issued to the L-A-S language interpreter in the form of an L-A-S operator statement. The sequence of L-A-S operator statements performing a desired analysis/design task is called a L-A-S program. A typical example of an L-A-S program implementing the LQG/LTR (linear quadratic Gaussian/loop transfer recovery) design methodology is given later on. The following discussions are based on a recent paper by Bingulac (1988).

Due to the limited space only a few statements from the L-A-S program will be explained. It is hoped, however, that the basic features of the L-A-S language
will be grasped and its applicability in solving a wide range of system/control analysis and design problems would become apparent. Particular attention will be paid to the LQG part of the design methodology, while the LTR part will only be mentioned briefly. Cited references provide more details.

**L-A-S language overview.** In the L-A-S language there are two types of operator statements:

1. Single operator statement (SOS),
2. Multiple operator statement (MOS).

The syntax of the SOS is as follows:

\[
<label>:\text{<inp-fl>}(\text{<oper-fl>}) = \text{<out-fl>}
\]

The symbols ‘::’ ‘(’ and ‘)’ ‘=’ act as field delimiters. The \( <label> \), which is optional, is usually used for recursive calculations. The input field, \( <\text{inp-fl}> \), consists of zero or more variables names and/or constants, separated by commas. The operator field, \( <\text{oper-fl}> \), contains the operator mnemonic name. The output field, \( <\text{out-fl}> \), consists of zero or more variable names defined by the operator.
The syntax of the MOS is basically the same as that of the SOS, with the exception of the input field where, in addition to variable names or constants, also the generalized variables may be specified. The syntax for the generalized variable is as follows:

\[
\text{<inp-fl> (<oper-fl>)}
\]

Since in operator statements the operator name follows all input variable names, it may be concluded that the L-A-S operator statements are written in the postfix notation.

As a simple example of both SOS and MOS, consider the calculation of the matrix \( S \) given by: \( S = B R^{-1} B^T \). The SOS approach to the calculation of the matrix \( S \) is:

\[
\begin{align*}
R(-1) &= T_1 & \text{Matrix inversion; } & T_1 = R^{-1} \\
B(t) &= T_2 & \text{Matrix transportation; } & T_2 = B^T \\
B, T_1(*) &= T_3 & \text{Matrix multiplication; } & T_3 = B \cdot T_1 \\
T_3, T_2(*) &= S & \text{Matrix multiplication; } & S = T_3 \cdot T_2
\end{align*}
\]

The MOS approach allows that all four operators be specified within a single operator statement, that is,

\[
B, R(-1), B(t)(*), (*) = S
\]

The generalized variables in this case are: \( R(-1) \); \( B(t) \) as well as: \( R(1) \), \( B(t)(*) \). The rather elaborate example of the MOS, given at the end of the Program 1 later on, deserves special attention. Consider a calculation of the matrix \( F \) defined by:

\[
F = \begin{bmatrix}
C^T (C^T)^{-1} \\
-(C A^{-1} B)^{-1}
\end{bmatrix}
\]

Given matrices \( A \), \( B \), \( C \), the matrix \( F \) may be calculated by the following MOS:

\[
\begin{align*}
C(t), C(t)(*), (-1)(*), \\
C, A(-1), B(*)(*), (-1), -1(s*)(rti) &= F,
\end{align*}
\]

where the L-A-S operators \(*\) (scalar multiply) and RTI (Row tie) perform multiplication of a matrix with a scalar and tying two matrices by rows, respectively.

Below is a CAD example presented to investigate the use of L-A-S in inputting two frequency-domain blocks, convert them to state space, cascade them together, and simulate it under an input step excitation.

**CAD Example B.15**

In this CAD example, the system of CAD Ex. B.1 shown in Fig. B.5 is reconsidered. As seen, the system consists of three poles and a zero. A listing of an L-A-S session to simulate this system for a step input follows.
* (inp-f; input the coeff. of the first system’s denominator
1,0,4,1(dma,t) = f; input coeff. of the first system’s denominator
  f
  1.000  .400  1.000

* (pmi) = g; coeff. of numerator must be in a polynomial matrix form
Enter dimens. and order for polyn. matrix <g> : 1,1,1
PMF Matrix <g> has < 2> rows and < 1> columns
Matrix <g> ; Enter E,R,C,D,M,I,Z,P,N or H for HELP : m
g
  2
  1

* f,g(tfss,t) = a,b,c; convert transfer function to state space
Result from the operator <TFSS> ; variable <a>
  -.400  1.000
  -1.000  .000

Result from the operator <TFSS> ; variable <b>
  1.000
  .000

Result from the operator <TFSS> ; variable <c>
  1.000  -2.000

* 1.96,1(dma,t) = f1; input the coeff. of the second system’s denominator
  f1
  1.960  1.000

* (pmi) = g1; coeff. of numerator must be in polynomial matrix form
Enter dimens. and order for polyn. matrix <g1> : 1,1,0
PMF Matrix <g1> has < 1> rows and < 1> columns
  g1
  1

* f1,g1(tfss,t) = a1,b1,c1; convert to state space
Result from the operator <TFSS> ; variable <a1>
  -1.960

Result from the operator <TFSS> ; variable <b1>
  -1.000

Result from the operator <TFSS> ; variable <c1>
  -1.000

* a,b,c,a1,b1,c1(ccon,t) = a2,b2,c2; cascade the 2 systems together
Result from the operator <CCON> ; variable <a2>
\[
\begin{bmatrix}
-0.400 & -0.706 & -0.707 \\
1.414 & 0.020 & 1.980 \\
0.000 & -0.020 & -1.980
\end{bmatrix}
\]

Result from the operator \(<\text{CCON}>\); variable \(<b2>\)

\[
\begin{array}{c}
1.000 \\
0.000 \\
0.000
\end{array}
\]

Result from the operator \(<\text{CCON}>\); variable \(<c2>\)

\[
\begin{array}{c}
0.000 \\
0.707 \\
-0.707
\end{array}
\]

- \(a_2,b_2,c_2(\text{mtf},t)=f_2,g_2\): convert the result back to a transfer function

Characteristic Polynomial \(<f_2>\)

\[
f_2
\begin{bmatrix}
1.980 & 1.784 & 2.360 & 1.000
\end{bmatrix}
\]

Transfer Function Matrix \(<g_2>\)

\[
g_2
\begin{bmatrix}
2.000 \\
1.000 \\
0.000
\end{bmatrix}
\]

Polynomial Matrix \(<g_2>\) has 1 Columns

- \(20(\text{step}.,\text{sub})=u\); input a unit step response
- \((\text{dsc})=t\); input a scalar for the total time of simulation

\[
t
10
\]

- \(f_2,g_2,u,t(\text{rct})=y\); compute the response \(y\) of the system
- \(\_\text{ylab, -SYSTEM\_TIME\_RESPONSE}\)
- \(\_\text{xlab, -TIME}\)
- \(\_y(\text{dis})=\)

Figure B.29 shows a printer plot of the output response of the system. Figure B.30 shows the plot of the simulated system's output. Note the closeness of this response with the dashed plot of Figure B.6 obtained by simulating the same system by CTRL-C.

**The LQR/LTR problem.** A typical LQR/LTR problem in robust control theory (Doyle and Stein, 1981) consists of the following steps.

An \(n\)th order state space realization of the nominal plant \(G(s)\) including all disturbance processes is given by:

\[
\begin{align*}
\dot{x} &= Ax + Bu + w \\
y &= Cx + v
\end{align*}
\] (B.3)
where \( w \) and \( v \) are Gaussian white noises representing plant disturbance and measurement noise, respectively. \( \{A, B, C\} \) is a plant state space realization. It is desired to minimize a given performance index:

\[
J = E \left\{ \int_0^\infty (z^T z + \rho \ u^T u) \, dt \right\}
\]  

(B.4)

where \( z = H x \), usually \( H = C \), thus \( z = y \), and \( \rho \) is a scalar. It is well known (Ridgely and Banda, 1986; Doyle and Stein, 1981; Kwakernaak and Sivan, 1992) that the minimization of (B.4) subject to (B.3) is given by the solution to the LQG design problem, i.e. by \( u = K_c \hat{x} \), where \( K_c = B^T P / \rho \), while the \( n \times n \) matrix \( P \) is the positive definite solution to the algebraic matrix Riccati equation:

\[
PA + A^TP + H^TH - PB \frac{1}{\rho}B^TP = 0
\]  

(B.5)

The vector \( \hat{x} \), called the state vector estimate is given by:

\[
\dot{\hat{x}} = A \hat{x} + B u + K_f(y - C \hat{x})
\]  

(B.6)

\[
K_f = P_f C^T / \mu,
\]

where \( \mu I \) is the intensity of the measurement noise \( u \) and \( P_f \) is the solution to the filter Riccati equation of the form:

\[
P_f A + A^T P_f + FF^T - P_f C^T / \mu CP_f = 0
\]  

(B.7)

\( F \) is the intensity of the plant disturbance \( w \).

* \( y(plt) = \) plot the response of the system

<table>
<thead>
<tr>
<th>Column #</th>
<th>of the variable ( y )</th>
<th>min.value</th>
<th>max.value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000</td>
<td>(-.000)</td>
<td>3.193</td>
</tr>
<tr>
<td>2</td>
<td>.352</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>.796</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1.205</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1.487</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1.596</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.540</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.362</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1.130</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>.910</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>.754</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>.687</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>.759</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>.798</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>.919</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.037</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1.124</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>1.163</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>1.156</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1.112</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure B.29 Output response of system of CAD Example B.15 via printer plot using L-A-S.
Thus, the "LQG" part of the whole LQG/LTR design methodology basically consists of calculating the feedback regulator matrices $K_c$ and $K_f$ as well as solving the Riccati Eqs. (B.5) and (B.7).

Before mentioning briefly the computational requirements for the LTR (Loop Transfer Recovery) part, let us consider now how the LQG part, described above can be solved by the L-A-S language.

**L-A-S implementation of the LQG problem.** The listing of the L-A-S program solving the LQG/LTR problem described in Ridgely and Banda (1986) is given in Program 1. In this section some of the statements will be explained. Due to limited space, the particular emphasis will be given to the L-A-S statements implementing the LQG part, that is, equations (B.3)–(B.7).

By the input operator statements INP and RBF (Read binary file) given at the beginning of the L-A-S program, the matrices in the system realization $(A, B, C)$ may be defined and used by any operator statement to be executed later. The DMA (Define matrix) operator statements define scalars $\mu = 0.5$ and $q = 10^5$. By the operator DSC (Define Scalar), the user may enter scalars nd, dec and omi, which define the frequency range in which the singular values of some transfer function matrices are to be calculated. The L-A-S variable nd, dec and omi represent

- nd = # of frequency values per decade,
- dec = # of decades and
- omi = initial frequency

---

1 See also Sec. 4.5 for more discussions on robustness issues in control systems.
The next L-A-S statement is the call to the L-A-S subroutines GOM (Generate omega) which generates the row omg containing logarithmically spaced frequency values starting with the initial value omi. The listing and explanation of the subroutine GOM are given in Program 2. The operator statement EGV (Eigenvalues) calculates the eigenvalues of the matrix appearing in its input field. The next several statements calculate the following matrices: $Q = C^T C$, $S = BR^{-1}B^T$, $R$ is the $(m \times m)$ unity matrix, $m =$ column dimension of the input matrix $B$. The matrices $Q$ and $S$ are required by the RIC operator statement which solves the Riccati Eq. (B.5). The next several operator statements calculate the following matrices

$$K_c = B^T K_r, \quad A_r = A - S K_c$$

$$Q_f = FF^T \quad \text{and} \quad S_f = C^T R/\mu C$$

$K_c$ and $A_r$ are plant regulator matrix and plant closed-loop system matrix, respectively, while the matrices $Q_f$ and $S_f$ are Kalman filter weighting matrices required by the second RIC operator statement which calculates the solution $K_f$ to Eq. (B.9). The next several operator statements calculate the following matrices

$$K_f = \frac{K_f C^T}{\mu} \quad A_f = A - K_f C \quad \text{and} \quad A_k = A - K_f C - B K_c$$

Also the eigenvalues $e_f$ and $e_k$ of the matrices $A_f$ and $A_k$ are calculated.

The TZS (Transmission zeros) operator statements executed in the sequel calculate the transmission zeros of the following realizations

$$\{A, F, C\} \quad \text{transmission zeros} \quad z_1$$

$$\{A, F_f, C\} \quad \text{transmission zeros} \quad z_2$$

$$\{A_k, K_f, K_c\} \quad \text{transmission zeros} \quad z_3$$

As pointed out in Ridgely and Banda (1986) the transmission zeros of these realizations indicate the quality of the Loop transfer recovery achieved by the LTR part. It may be concluded from the L-A-S statements used, the transmission zeros of a realization $\{A, B, C\}$ are calculated by solving the generalized eigenvalue problem given by

$$A p = E \lambda p,$$  

with:  

$$A = \begin{bmatrix} A & B \\ C & 0 \end{bmatrix}, \quad E = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}$$

where $I_n$ is the $(n \times n)$ unity matrix. It is well known that the generalized eigenvalues $\lambda$ satisfying (B.11), represent the transmission zeros of the realization $(A, B, C)$.

The MTF (Matrix Transfer Function) operator statement executed after the calculation of the transmission zeros $z_3$ calculates the transfer function matrix

$$C(sI - A)^{-1}F = \frac{G(s)}{p(s)}$$

(B.12)
The output variables \( p \) and \( G \) of the MTF operator statement contain the coefficients of the characteristic polynomial: \( p(s) = \det (sI - A) \) and coefficients of all \( m^2 \) polynomials \( g_y(s) \) in the \((m \times m)\) numerator polynomial matrix \( G(s) = [g_y(s)] \). As it may be concluded from (B.12) \( g_y(s) \) are polynomials of up to \((n - 1)\)st order, each having \( n \) coefficients. The transfer function matrix \( G(s)/p(s) \) is calculated in order to calculate the singular values of the complex matrix \( G(j\omega)/p(j\omega) \) for each \( \omega \) contained in the row omg generated by the L-A-S subroutine GOM described earlier.

The required singular values expressed in [dB] are calculated by the L-A-S subroutines GSV and LOG. The listings and explanations of these two subroutines are given under Program 2. Finally, the singular values in [dB] are plotted versus log \((\omega)\) using the L-A-S operator statement DISL (Display logarithmically), see Fig. B.31.

The two groups of L-A-S statements which follow calculate and plot the singular values of the following transfer function matrices (Doyle and Stein, 1981)

\[
C(sI - A)^{-1} K_f ; \quad W(s) \quad K(s),
\]

where

\[
W(s) = C(sI - A)^{-1}B \quad \text{and} \quad K(s) = K_c(sI - A_c)^{-1}K_f.
\]

As explained earlier, the calculation and plotting of these singular values is performed by the following sequence of L-A-S operator statements and subroutines

MTF, GSV, LOG and DISL.

Some singular value plots and time domain responses obtained using the operator statements DISL and DIS are given at the end of the current discussion.

Since the control systems designs ultimately have to be evaluated in the time domain, the step responses of the obtained feedback system are to be calculated. This is done in the L-A-S program which follows this section by converting the closed-loop system realization \( \{A_c, B_c, C_c\} \) to the equivalent discrete realization. This is accomplished in L-A-S by the RDL (Discrete Realization) operator statement. The step responses to the two typical command signals of the obtained discrete model are calculated by the RDS (Response of a Discrete System) operator statement. The L-A-S language has, of course, operator statements for calculating response of continuous models (Bingulac et al., 1985), but here, for demonstration purposes, the responses of the equivalent discrete models are calculated. The step responses are plotted using the DIS (Display time response) operator statements, see Fig. B.32.

The last part of the attached LQG/LTR design consists of L-A-S statements (Program 2) calculating the “augmented system realization” \( \{\bar{A}, \bar{B}, \bar{C}\} \), where in front of the plant \( \{A, B, C\} \) the block of \( m \) integrators has been added, that is,

\[
\bar{A} = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} \quad \bar{B} = \begin{bmatrix} 0 \\ I \end{bmatrix} \quad \bar{C} = \begin{bmatrix} C & 0 \end{bmatrix}
\]
Figure B.31 Frequency responses of filter and plant singular values for the LQG/LTR Problem: (a) Filter singular values; (b) Plant singular values.

This is done by the L-A-S subroutine M41 which is described in Program 2. Augmenting the plant model with integrators (Doyle and Stein, 1981) is done in order to improve the performance of the closed-loop system.

Unlike other existing packages used in CAD of control system, the L-A-S language/package permits jumping back to any previously executed statement in the
current L-A-S program. By using this feature and by renaming the realization \( \{A, B, C\} \) with the augmented one \( \{A, B, C\} \), which is done by the MCP (Matrix Copy) operator statement, the user may jump back to the beginning of this L-A-S program, to the statement having the label "a," and perform exactly the same calculations, but now using different data for the plant realization as well as for the tunable parameters \( \mu \), \( \rho \) and the matrix \( F \).

**Programs listings.** Below are given the listings of two L-A-S programs for the LQG/LTR problem.

**PROGRAM 1**

```
-LQG/LTR-
(inp,r)=A,B,C
(sto)=
(rbf)=A,B,C,[abcL]
A,B,C(mcp)=Ao,Bo,Co
5(dma)=mu
B(mcp)=Ga
100000(dma,t)=q
(dsc)=nd,dec,omi
   _nd_=._5_;_dec_=._5_;_omi_=._01_
nd,dec,omi(gom,sub)=omg
omg(out)=
```

System realization \( \{A,B,C\} \) may be entered from the terminal keyboard of Disk file

\[ R = \{A,B,C\} \rightarrow Ro = \{Ao,Bo,Co\} \]

Definition of tunable parameters \( \{mu,Ga,q\} \)

Definition of the frequency range

The subroutine "GOM" generates row "omg" containing logarithmically spaced frequency values
a:A,B,C(out) =
A(egv,t) = eg
eg(out,e) =
musqr((-1)) = musi

This "output" statement has a label "a"
Eigenvalues of $A \rightarrow eg$

$1/2$

$1/(mu) \rightarrow musi$

$T$

$C * C \rightarrow Qo$ ; $Qo * q \rightarrow Q$

$B(cd) = m$

Row dimension of $B \rightarrow m$

$(m \times m)$ unity matrix $\rightarrow R$

$-1 \rightarrow T$

$B * R * B \rightarrow S$

$A,Q,S(ric,t) = Kr$

Riccati Solution $\{ A,Q,S \} \rightarrow Kr$

$T$

$B * Kr \rightarrow Kc$ Full feedback regulator

$A,S,Kr(*)(-,t) = Ar$

Eigenvalues $(A - S * Kr) \rightarrow er$

regulator poles

$Ga,Ga(t)(*,t) = Qf$

Kalman filter weighting matrices $(Qf,Rf)$

$R, mu(-1)(s*,t) = Rf$

$T$

$C(t), Rf, C(*)(*,t) = Sf$

$Ga * Ga \rightarrow Qf$ ; $R/mu = Rf$ ; $C * Rf * C = > Sf$

$T$

$A(t),Qf,Sf(ric,t) = Krf$

Kalman filter gain $(A,Qf,Sf) \rightarrow Krf$

$T$

$Kr f * C /mu \rightarrow Kf$ ; Filter feedback

$A,Kf,C(*)(-,t) = Af$

Filter $(A - Kf * C)$ poles $\rightarrow ef$

$A,Kf,C(*)(+)(-),t) = Ak$

$A - Kf * C$ poles $\rightarrow ef$

$A - Kf * C - B * Kc \rightarrow Ak$

$A - Kf * C - B * Kc \rightarrow Ak$

Controller $K(s)$ system matrix $Ak$

$K(s)$ poles $\rightarrow ek$

System order $\rightarrow n$

$n + m \rightarrow nm$

$nm,nm(dzm),n,n(dim),1,1(rmp,t) = Et$

$m,n,dzm = D$

$A,C,Ga,D(m41,sub) = Cfg$

$Cf(ou) =

$Cf,Et(tzs,t) = z1$

Transmission zeros $\{ Cfg \} \rightarrow z1$

$A,C,Kf,D(m41,sub) = Cfk$

$Cf(k,tz(t),z2) = z2$

Transmission zeros $(Cfk) \rightarrow z2$

$A,Kc,Kf,D(m41,sub) = KcKf$

$KcKf,Et(tzs,t) = z3$

$C *(I - A) * Ga = G(s)f(s)$ Plant

transfer function matrix
Characteristic polynomial coeff. $\Rightarrow f$
Numerator polynomial coeff. $\Rightarrow G$

$\text{st0} = \text{pol}$

$f(t), 1(\text{cmp}) = \text{pol}$

$G, omg, pol(Gsv, \text{sub}) = Wpl$

$Wpl, musi(s) = Wpl$

$Wpl(\log, \text{sub}) = WpdB$

$omi, nd(inc), dec(cct, t) = w$

$\text{ylab}_{[\text{dB}]} \_ \text{Plant} \_ C[ls - A]^{-1} \text{Ga}_\text{s} \_ \text{Sing} \_ \text{values}$

$\text{xlab, Frequency } [\text{rad/sec}]$

$WpdB(t), w(\text{disl}) = \text{Plot of max. and min. Sing. values, in [dB], of } WpdB \text{ vs. } \log(\text{omega})$

$A, Kf, C(mtf, t) = ff, Gf$

$C \ast (ls - A) \ast Kf = Gf(s)/ff(s)$

$\text{ff}(t), 1(\text{cmp}) = \text{pol}$

$Gf, omg, pol(Gsv, \text{sub}) = Wf1$

$Wf1(\log, \text{sub}) = WfdB$

$\text{ylab}_{[\text{dB}]} \_ \text{Filter} \_ \text{Sing} \_ \text{values}$

$WfdB(t), w(\text{disl}) = \text{Plot of max. and min. Sing. values, in [dB], of } WfdB \text{ vs. } \log(\text{omega})$

$WfdB, WpdB(\text{rtl})(t), w(\text{disl}) = \text{Plot of both } WfdB \text{ and } WpdB$

$(\text{sto}) = \text{pol}$

$A, Kf, Kc(mtf) = fk, K$

$Kc \ast (ls - Ak) \ast Kf = K(s)/fk(s)$

$\text{fk, f(p*)(t), 1(\text{cmp}) = pol}$

$G, K(pmm) = GK$

$G(s) \ast K(s) = GK(s)/pol(s)$

$\text{Gk, omg, pol(Gsv, sub) = Wgk}$

$Wgk(\log, \text{sub}) = GkdB$

$\text{ylab}_{[\text{dB}]} \_ G[s] \_ K[s] \_ \text{Sing} \_ \text{values}$

$GkdB(t), w(\text{disl}) = \text{Plot of Sing. values of } Wgk, \text{ in [dB], vs. } \log(\text{omega})$

$\text{ylav}_{[\text{dB}]} \_ \text{Filter} + G[s] \_ K[s] \_ \text{Sing} \_ \text{v} \_ \text{...}$

$WfdB, GkdB(\text{rtl})(t), w(\text{disl}) = \text{Plot of both } WfdB \text{ and } GkdB$

$\text{...Step_responses}$

$A, Kf, C(*) = -1(s*)B, Kc(*), Ak(m41, sub) = Act$

$Kf, Kf(-), Kf(rti) = Bcl$

$C, C, C(\text{ctc}, t) = Ccl$

$21(\text{step, sub}) = udl$

$.1(\text{dml}) = dt$

$dt, Atot, Btot(\text{drl}) = Ad, Bd$

$udl, udl(\text{t}) = z$

$Ad, Bd, Ctot, udl, z(\text{ctc})(\text{drl}) = \text{Resl}$

$\text{ylab, } \_ \text{Sideslip } \_ \text{Command } \_ \text{Step } \_ \text{Response}$

$\text{Response to the step change of the first input } = \Rightarrow \text{Resl}$
xlab_: Time_..([sec]
Res1,2(dis)=
Ad,Bd,Ctot,z,udl(cti|rds) = Res2
ylab_:Roll_Command..Step_Response
Res2,2(dis)=
Plot of Res1 vs. time (linear)
Response to the step change of
the second input => Res2
Augmented design: adding
integrators in front of the plant
A,m,n(dzm),B,m,m(dzm)(m41,sub) = Aau

| A  B |
| 0 |  => Aau ; | 0 |

B,B(− ),m,m(dim)(rti)= Bau
| 0 0 |
|Im|
|C 0 |
=> Bau

C,m,m(dzm)(cti,t)= Cau
Aau,Bau(out)=
New choice of tunable parameters: mu = 0.01
0.01(dma) = mu
C(t),C(t)*(− 1)*(− 1),B,− 1(s+)*(+)*(− 1)(rti,t) = Ga

(sto) =
| T | T 0 |
| 1 | C * ( C C ) |
| 0 | 1 |
| 0 | G(o) |

Augmented realization {Aau,Bau,Cau} = => {A,B,C}
By jumping back at the statement with the label
"a", another LQG/LTR design using the augmented
realization may be performed.

Notes: The LQR/LTR design implemented by the above L-A-S program fol-
lows the design procedure presented in Ridgely and Banda (1986).

PROGRAM 2: Listings and explanations of L-A-S subroutines called by the main
L-A-S program LQG/TR

Subroutine GOM

1 n,dec,omi(gom,sub)= omv
2 (nl) =
3 10(log),n(f)/exp = dom
4 omi(mcp) = omc
5 omi(mcp) = omv
6 n,n(− ) = j
7 k:j:inc = j
8 n,n(− ) = i
9 i:i:inc = i
10 omc,dom(− ) = omc
11 omv,omc(cti,t) = omv
12 i,n(ifj) = i,j
13 j;j,dec(ifj) = k,1,1
14 1:(lis) =
**Subroutine M41**
Defines the augmented matrix A

1. \( A11, A21, A12, A22(m41, sub) = A \)
2. \((nli) = \begin{bmatrix} A11 & A12 \\ A21 & A22 \end{bmatrix} = > A \)
3. \( A11, A21(rt1) = A1 \)
4. \( A12, A22(rt1) = A2 \)
5. \( A1, A2(ct1) = A \)

**Subroutine GSV**
Calculates \((k \times n)\) matrix \( Wm \)

1. \( Gs, sv, pol(Gsv, sub) = wm \)
2. \((nli) = \) Calculates \((k \times n)\) matrix \( Wm \)
3. \( sv(cdi) = n \)
4. \( sv, sv(-, sv(rt1)(t)) = sm \)
5. \( Gs(cdi)(sqr, t) = k \) where Sing. val. \{Gs(jwi)/pol(jwi)\} = > wmi
6. \( 1, k(dzm) = z \)
7. \( z(inc), z(rt1)(mtv)(dsm) = Sm \) Note that \( Sm /= sm ! \)
8. \( k(0,dzm) = wm \)
9. \( i: (nop) = sv = \{w1, w2, \ldots, w1, \ldots, wn\} \)
10. \( sm, 1(ctr) = si, sm \)
11. \( pol, si(gs, t) = pr, pi \) The \((k \times k)\) transfer function
12. \( pr, pi(f(+), pi, pi(f)(+, t) = pri \)
13. \( pr, pr(f, t) = pr \) matrix \( T(s) \) is defined by:
14. \( pi, pr(f, t) = pi \)
15. \( Gs, si(gs, t) = pr, gi \) \( T(s) = Gs(s)/pol(s), \) where
16. \( gr, pr(s+, gi), pi(s+)(+, t) = grn \)
17. \( gi, pr(s+, gi), pi(s+)(-, t) = gin \)
18. \( grn, gin(cti), gin, -1(s+), grn(cti)(rti)(grh) = w \)
19. \( w, Sm (+) = w \)
20. \( wm, w(t)(cti, t) = wm \) Characteristic polyn. coeff. = pol
21. \( sm(rdi)(ifj) = j, j, i \) Numerator polynomials coeff. = Gs
22. \( j: (lis) = \)

**Subroutine STEP**
Defines the \( n \) dimensional column \( u \)

1. \( n(step, sub) = u \)
2. \((nli) = \)
3. \( 1, n(dzm)(inc)(t) = u \) containing \( n \) unities. i.e.
4. \( (lis) = u = \{ui\}; \ u = 1 ; \ i = [1, n] \)

**Subroutine LOG**
Defines the \((2 \times n)\) matrix \( V \)

1. \( w(log, sub) = V \)
2. \((nli) = \)
3. \( 10(log), 20(dma)(f, t) = lnlo \)
4. \( w(cdi) = m \) \( V = \begin{bmatrix} v1 \\ v2 \end{bmatrix} \)
5. \( w(mcp) = V \)
6. \( V(rdi) = r \)
7. \( r, 2(ifj) = b, b, m \) The elements \( v1i \) and \( v2i \) of the
Sec. B.4 Other Design Packages

8 \( m:r,r(\text{dim})=1 \)  \quad \text{rows \; v1 and v2, respectively are defined by:}
9 \( l,1(\text{ctr})=r1 \)
10 \( l,r(\text{dec})(\text{ctr})=x,r2 \)
11 \( r1,r2(\text{rli}),v(s^*)=v \)
12 \( b:V(\text{mtr})=V \)
13 \( V(\log),l10(1)0(-1)(s^*,t)=vv \)  \quad v1i = 20log(wf) ; v2i = 20log(wL)
14 \( vv,m(\text{vtn},t)=V \)
15 \( \text{(Lis)}= \)  \quad \text{wf and wL are elements in the first and the last rows of the (m x n) matrix \; w.}

CAD Example B.16

In this example, the third-order system described at the end of CAD Ex. B.9 concerns the numerical inversion of the Laplace transform. An implementation of this problem on the L-A-S follows.

* \( \text{(inp)}=f \); input the coefficients of the denominator

Enter dimensions for \(<f>\) : 1,4
Matrix \(<f>\); Enter E,R,C,D,M,I,Z,P,N or H for HELP : m
5,5,25,5,1

* \( \text{(pmi)}=g \); coeff. of numerator must be in a polynomial matrix form

Enter dimens. and order for polyn. matrix \(<g>\) : 1,1,3
Matrix \(<g>\); Enter E,R,C,D,M,I,Z,P,N or H for HELP : m
0
1
1

* \( \text{(inp)}=t0 \); input the initial value vector

Enter dimensions for \(<t0>\) : 3,1
Matrix \(<t0>\); Enter E,R,C,D,M,I,Z,P,N or H for HELP : m
1
-4
14.75

* \( 20(\text{step,sub})=u \); input a unit step response vector

* \( (\text{dsc})=t \); input a scalar for the total time of simulations

10

* \( f,g,u,t,t0(\text{rct})=y \); compute response \( Y \) of the system

A printer plot of the numerically inverted output of the system is shown in Fig. B.33a. Once again, three simple L-A-S commands will give us a display plot of the response which is shown in Fig. B.33b.

There are many other non-MATLAB packages which one could also try to describe. The most notable ones are program CC by Thompson and Wolf (1983),
and package LUND (Elmqquist, 1977). However, due to lack of space, we only give a brief description of each.

Program CC (Thompson and Wolf, 1983) is an IBM-PC type CACSD package which has been implemented in executed BASIC. It can treat multivariable systems described in both time and frequency-domain, including "transfer function matrix" form. It should be noted that among MATLAB-based programs only IMPACT (Rimvall, 1983) and program M by Lawrence Livermore National Laboratory (Lawver, 1985) have this feature. Special features in CC are graphical display of transfer functions, partial fraction expansion, inverted Laplace and z-transform functions, symbolic manipulation of transfer functions, and state space and frequency domain analysis of multirate sampled-data systems. The symbolic manipulation of transfer functions is a rather unique feature of CC. A typical example of this manipulation follows.

\[
G(P) = \frac{5s^2 + .5s + 5}{(s + 1)(s^2 + 2P + 5)}
\]

\[
G(S) = \frac{4.8}{s + 2} + \frac{-2s - 9.5}{[(s + 1) + 2]} 
\]

**B.5 CACSD TOOLS—A BRIEF SURVEY**

In this appendix, an attempt has been made to give a brief introduction into an increasingly vast field, *computer-aided control systems design*. It is estimated that over 50 different packages of different degrees of development exist in the world today. A true survey of all these packages is, in fact, a near impossibility. Cellier and Rimvall (1988) have made an extensive effort to survey as many CACSD as they could gather. In this section, we present a brief survey of some 22 CACSD packages, some not surveyed before, and try to give a perspective of their usefulness.

Table B.12 shows a brief survey of 22 CACSD packages. This table is an abbreviated version of a more extensive one of Cellier and Rimvall (1988) with new additions. We have chosen 14 aspects of CACSD packages—the first six are considered as basic attributes which should be available in all packages. The next five attributes are considered as more advanced ones which may not be available in all programs. The final three attributes are a matter of choice on the part of the developers and purchasers of the particular program.

Among the MATLAB-based programs, this particular survey reveals that MATRIXx is the most complete CACSD package. However, three packages PC (PRO) MATLAB, CTRL-C, and IMPACT are close behind as second best. Then
### TABLE B.12  A brief survey of 22 CACSD Packages.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>C</td>
<td>M</td>
<td>P</td>
<td>C</td>
<td>C</td>
<td>I</td>
<td>M</td>
<td>C</td>
<td>L</td>
<td>L</td>
<td>T</td>
<td>T</td>
<td>C</td>
<td>T</td>
<td>W</td>
<td>I</td>
<td>M</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>A</td>
<td>T</td>
<td>A</td>
<td>C</td>
<td>A</td>
<td>O</td>
<td>M</td>
<td>A</td>
<td>A</td>
<td>U</td>
<td>A</td>
<td>I</td>
<td>C</td>
<td>R</td>
<td>C</td>
<td>C</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>S</td>
<td>U</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>R</td>
<td>T</td>
<td>-</td>
<td>T</td>
<td>N</td>
<td>P</td>
<td>T</td>
<td>D</td>
<td>N</td>
<td>S</td>
<td>M</td>
<td>I</td>
<td>D</td>
<td>A</td>
<td>D</td>
<td>A</td>
<td>N</td>
<td>P</td>
<td>B</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>R</td>
<td>M</td>
<td>P</td>
<td>T</td>
<td>A</td>
<td>L</td>
<td>A</td>
<td>D</td>
<td>D</td>
<td>P</td>
<td>-</td>
<td>R</td>
<td>P</td>
<td>S</td>
<td>C</td>
<td>A</td>
<td>O</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-</td>
<td>I</td>
<td>A</td>
<td>A</td>
<td>R</td>
<td>C</td>
<td>A</td>
<td>C</td>
<td>O</td>
<td>D</td>
<td>E</td>
<td>A</td>
<td>A</td>
<td>K</td>
<td>P</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td>X</td>
<td>T</td>
<td>C</td>
<td>O</td>
<td>T</td>
<td>B</td>
<td>S</td>
<td>M</td>
<td>S</td>
<td>C</td>
<td>D</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>L</td>
<td>L</td>
<td>-</td>
<td>(K)</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-</td>
<td>S</td>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>I</td>
<td>C</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**GLOBAL CLASSIFICATION**

| Continuous Systems | 0 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 0 | 1 | 1 | 0 | 2 | 2 |
| Discrete Systems   | 0 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 0 | 2 | 0 | 1 | 2 | 0 | 2 |
| Time Domain        | 0 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 2 | 2 | 0 |
| Frequency Domain   | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 2 | 0 | 2 | 2 | 1 | 0 | 2 | 2 | 2 | 2 | 1 | 0 | 0 | 2 | 2 | 2 |
| SISO               | 0 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 |
| Multivariable      | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 0 | 2 | 1 | 1 | 1 | 1 | 2 | 2 | 0 | 2 | 2 | 0 | 0 | 1 | 0 | 2 | 2 |
| Nonlinear Systems  | 0 | 2 | 2 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 2 | 1 | 0 | 2 |
| Adaptive Control   | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Identification     | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 1 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 2 | 1 | 0 | 0 |
| Real-Time Interface| 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Extendability      | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 1 | 2 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| Source Code Availability| 2 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 1 | 1 | 0 | 2 | 0 | 2 | 0 | 0 | 2 | 1 | 1 | 0 | 2 | 0 | 2 |
| Maintenance        | 0 | 2 | 2 | 2 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Cost               | 2 | 0 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Total No. of Points**

| 6 | 18 | 21 | 19 | 12 | 17 | 18 | 7 | 23 | 22 | 13 | 14 | 14 | 17 | 14 | 13 | 9 | 11 | 10 | 12 | 13 |
comes CONTROL_lab as third best, and fifth overall. Note that the original MATLAB received a mere six points which reinforces the fact that MATLAB was not intended for control systems design.
Among the non-MATLAB packages the results are not surprising. Comprehensive packages such as CADACS (KEDDC) and LUND scored very high indeed. Most other packages are somewhat equal in strength. It is noted that among all packages in Table B.12, CADACS of Schmid (1985) ranks as the top CACSD program available today. Interested users outside Germany and Austria may contact the first author through the postcard at the end of the book for the availability of CADACS.

The survey presented here is by no means an objective one. This survey was in fact a rather subjective one. In fact, a truly objective survey for a field such as CACSD may not be possible since not all such programs and packages are available to a reviewer and they are often changing in characteristics and capabilities. Moreover, the use of 0-1-2 points system adopted in Table B.12 is somewhat arbitrary and a different point system could potentially draw different conclusions. This survey, as indicated earlier, is by and large, based on the responses to a questionnaire conducted by Cellier and Rimval (1988).

### B.6 FUTURE OF CACSD PROGRAMS

The computer-aided design of control system has come a very long way in a few short years. In fact it was only 7 years ago when the first comprehensive survey on the subject was published by Jamshidi and Herget (1985) which has been translated into Russian. A new edition of this book is under preparation (Jamshidi and Herget, 1993). However, thus far the emphasis on CACSD packages has been on the program, and not so much on the data. The next generation packages should make use of clear distinction between the program (static codes in memory) and the data (a portion of memory that changes its content during program execution). The future packages should incorporate the user interface as a very important element in the success of their wide-spread use.

In recent years, the integration of control theory and artificial intelligence in various forms such as "connection networks," "neural networks," "fuzzy logic systems," "expert systems," or "rule-based systems" have become a very popular topic, called by some, "intelligent control." In a similar fashion, CACSD packages are now being developed with the help of artificial intelligence. One such effort is by Robinson and Kisner (1988) as well as McClung and Jamshidi (1991) who have developed an object-oriented AI-based CAD package for nuclear reactor control. They have established it in LISP (LISt Programming) on a LISP machine TI Explorer, and on a regular MACII and MAC/SE. With the use of LISP, CACSD packages would be capable of modifying their codes automatically. However, the drawback is the slower speed of LISP and its difficult user interface.

Another potentially big impact of CACSD environment would come about when parallel computer architecture becomes reality. This would open the way to many other possibilities such as the uses of PROLOG (PROgramming in LOGic) and its more efficient characteristic. Finally, nonnumerical controller designs are perhaps becoming a reality in the future. The emergence of new symbolic languages such as MACSYMA (Symbolics, 1983), REDUCE (RAND Corp., 1985) and MuMath (Microsoft, 1984) would help towards that goal.
Bibliography

ACKERMANN, J., 1972, ‘‘Der Entwurf linearer Regelungssysteme in Zustandsraum,’’ Rege-
lungstechnik und Prozessdatenverarbeitung, 7, pp. 297–300.
AGATHOKLIS, P., F. E. CELLIER, M. DJOUDJEVIC, P. O. GREPPER, AND F. J. KRAUS, 1979,
‘‘INTOPS, Educational Aspects of Using Computer-Aided Design in Automatic Control,’’
in Proceedings of the IFAC Symposium on Computer-Aided Design, Zürich, Switzerland,
ANSI, 1985, American National Standard for Information Systems, Computer Graphics, 
Graphical Kernel System (GKS), Functional Description (ANSI X3.124–1985) and FOR-
AOKI, M., 1971, ‘‘Aggregation,’’ in Optimization Methods for Large-Scale Systems with 
AOKI, M., 1978, ‘‘Some Approximation Methods and Control of Large-Scale Systems,’’ IEEE 


B


C


D


F


G


K


L


LAWVER, B., 1985, EAGLES, An Interactive Environment and Program Development Tool, Personal Communication, Dynamics and Controls Group, Lawrence Livermore National Laboratory, University of California, Livermore, CA.


M


N


O


P


Bibliography


R


RIMVALL, C. M., 1983, IMPACT, Interactive Mathematical Program for Automatic Control, Swiss Federal Institute of Technology, ETH-Zentrum, CH-8092 Zurich, Switzerland, p. 208.


S


T


Bibliography


V


W


X


Y


Z


Index

A

Ackermann formula, 116
Adaptive control, 366
system design methods, 427
Aggregated models, 286
state, 297
Aggregation, 285, 286
balanced, 306
dynamic exactness, 289
matrix, 289
perfect, 288
Algebraic matrix Riccati Equation (AMRE), 253, 362
Algorithm:
controllable canonical form, 101
controllable Hessenberg form, 107
direct, 96, 101, 115, 116, 124
interaction prediction, 328
iterative, 182, 184
observable canonical form, 149
Hessenberg form, 149
design, 152, 155
output feedback, 177, 184
parallel, 124, 127
pole placement, 113, 115, 122, 127, 134
recursive, 107
ANSI standard, 371
APL—a programming language, 362
APPOLLO domain, 371
Associative, 343
Asymptotic stability, 80–82, 198
tracking, 5, 218

B

Balanced aggregation, 306
BASIC, 370
Basis, 344
Bass and Gura formula, 115
BIBO stable, 85, 87
Binet-Cauchy formula, 352, 353, 357
Brief survey of CACSD packages, 451
<table>
<thead>
<tr>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACSD methods, 365</td>
</tr>
<tr>
<td>programs, a classification of, 369</td>
</tr>
<tr>
<td>future of, 453</td>
</tr>
<tr>
<td>software, 360</td>
</tr>
<tr>
<td>tools, brief survey of, 450</td>
</tr>
<tr>
<td>CAD LAB, 418</td>
</tr>
<tr>
<td>laboratory for systems &amp; robotics,</td>
</tr>
<tr>
<td>417</td>
</tr>
<tr>
<td>techniques, 361</td>
</tr>
<tr>
<td>tools, 361</td>
</tr>
<tr>
<td>CADACS (KEDDC), 425, 430–31</td>
</tr>
<tr>
<td>features of, 427</td>
</tr>
<tr>
<td>program structure of, 429</td>
</tr>
<tr>
<td>Canonical decomposition, 76</td>
</tr>
<tr>
<td>form, 30</td>
</tr>
<tr>
<td>observable form, 66</td>
</tr>
<tr>
<td>state difference equations, 272</td>
</tr>
<tr>
<td>Categorization of CONTROL/lab, 404</td>
</tr>
<tr>
<td>Cayley-Hamilton theorem, 35, 36, 44, 48, 349</td>
</tr>
<tr>
<td>Centrality, concept of, 285</td>
</tr>
<tr>
<td>Centralized tasks, 428</td>
</tr>
<tr>
<td>Centrifugal torque, 22</td>
</tr>
<tr>
<td>Characteristic polynomial, 27, 31, 346</td>
</tr>
<tr>
<td>desired, 99</td>
</tr>
<tr>
<td>Code-driven interface, 370</td>
</tr>
<tr>
<td>Column vector, 343</td>
</tr>
<tr>
<td>Command-driven interface, 370</td>
</tr>
<tr>
<td>Companion form, 114</td>
</tr>
<tr>
<td>Compensator design, 174</td>
</tr>
<tr>
<td>Complementary sensitivity function, 10, 210</td>
</tr>
<tr>
<td>Complete solution, 41</td>
</tr>
<tr>
<td>Complex eigenvalues, 83</td>
</tr>
<tr>
<td>Comprehensive CACSD package, 369</td>
</tr>
<tr>
<td>Computational aspects, 105</td>
</tr>
<tr>
<td>Computer-aided design, 15</td>
</tr>
<tr>
<td>Computer-aided control systems design (CACSD), 360, 450</td>
</tr>
<tr>
<td>languages, 359–410</td>
</tr>
<tr>
<td>packages, 359–410</td>
</tr>
<tr>
<td>Conjugacy, 323</td>
</tr>
<tr>
<td>Conjugate direction method, 323</td>
</tr>
<tr>
<td>transpose, 345</td>
</tr>
<tr>
<td>Connection networks, 453</td>
</tr>
<tr>
<td>Continuous hypersurface, 243</td>
</tr>
<tr>
<td>discrete and hybrid systems, 384</td>
</tr>
<tr>
<td>continuously differentiable, 24</td>
</tr>
<tr>
<td>CONTROL.Lab, control system design methods, 427</td>
</tr>
<tr>
<td>Control kit, 418</td>
</tr>
<tr>
<td>Control systems, adaptive, 6</td>
</tr>
<tr>
<td>classical, 11</td>
</tr>
<tr>
<td>closed-loop, 4</td>
</tr>
<tr>
<td>modern, 11</td>
</tr>
<tr>
<td>stochastic, 6</td>
</tr>
<tr>
<td>open-loop, 4</td>
</tr>
<tr>
<td>CONTROL.lab, 84, 97, 137, 257, 371, 403–6, 408, 419</td>
</tr>
<tr>
<td>categorization of, 404</td>
</tr>
<tr>
<td>CSCS, 405</td>
</tr>
<tr>
<td>Controllability, 56, 59, 98, 199, 306</td>
</tr>
<tr>
<td>indices, 101</td>
</tr>
<tr>
<td>matrix, 101</td>
</tr>
<tr>
<td>Controllable, 98</td>
</tr>
<tr>
<td>canonical form, 30, 101</td>
</tr>
<tr>
<td>Hessenberg form, 106</td>
</tr>
<tr>
<td>Controller, PI, 8, 197</td>
</tr>
<tr>
<td>PID, 8, 207</td>
</tr>
<tr>
<td>Convolution, 41</td>
</tr>
<tr>
<td>Coordinate transformation, 101</td>
</tr>
<tr>
<td>Coordination, 321</td>
</tr>
<tr>
<td>Coordinator problem, in hierarchical control, 324</td>
</tr>
<tr>
<td>Coriolis torque, 22</td>
</tr>
<tr>
<td>Cost function, 230</td>
</tr>
<tr>
<td>Costate vector, 242</td>
</tr>
<tr>
<td>CSMP (continuous systems modeling program), 360</td>
</tr>
<tr>
<td>CSSL specifications, 360</td>
</tr>
<tr>
<td>CTRL-C, 371, 372</td>
</tr>
<tr>
<td>Cyclic, 120, 177, 197</td>
</tr>
<tr>
<td>Cyclicity, 199</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data structure, 360</td>
</tr>
<tr>
<td>Data-driven interface, 370</td>
</tr>
<tr>
<td>Dead-beat control, 173</td>
</tr>
<tr>
<td>Decentralized control, 285, 332, 333</td>
</tr>
<tr>
<td>Decentralized output, iterative solution to, 335</td>
</tr>
</tbody>
</table>
Decomposition, 74, 321
Description of systems, 2
  input-output, 2
  state-variable, 2
Design, 360
  methods, adaptive control, 427
  tools, 360
  optimal control, 230
Detectability, 257
Determinant, 351–54
Development of CAD methods and programs, 365–71
Diagonal form, 32
Difference equations, 18
Differential matrix Riccati equation, 252
Dimension of a matrix, 344
Diophantine equation, 218
Direct digital control, 428
Direct method, 115, 116
Discrete-state regulator problem, 273
Discrete-time system, 18, 26, 42, 69, 173
  matrix Riccati equation, 274
  optimal control, 232
Discretization, 47
Distributive, 343
Disturbance rejection, 9, 210, 218
Disturbance signal, 7, 9
Disturbances, 56
Dominant set of eigenvectors, 297
DSL (digital simulation language), 360
Dual, problem in hierarchical control, 323
Duality, 68
Dyadic, 96, 99, 119, 120
Dynamic compensator, 208
  and static output feedback, 220
Dynamic equations, 19

E

Eigenvalue, 33, 49, 99, 105, 346–48
Eigenvalue assignment (see also Pole placement), 99, 133
Eigenvector, 346–48
Eigenvector assignment, 133
EISPAK, 372
Elementary operations, 352
Engineering graphics, 374
Equilibrium point, 80
Estimator, state, 96, 139
EXEC FILE, 412
Experimentation with models, 360
Expert systems, 453

F

Fadeev algorithm, 27
Feedback gain (matrix), 97, 116, 120, 122, 136
Field, 343
First-level subsystem problem, 324
Fixed modes, 334
Form-driven interface, 370
FREDOM, 418, 419
FREDOM-TIMDOM, tree structure for, 420
Frequency-domain criteria, 70
Full rank, 96, 122
  state feedback, 163
Functions of a matrix, 348, 349, 350
Fundamental matrix (see also Transition matrix), 251
Fuzzy logic systems, 453

G

Gaussian elimination, 105
General aggregation, 287
Generalized variables, 436
  eigenvector, 347
Global problem in hierarchical control, 322
Globally asymptotic stability, 80
Graphic-driven interface, 370
Graphic-driven kernel system (GKS), 370
Graphics, 430

H

H₂ Control theory, 15
Hamilton-Jacobi equation, 233, 235, 239
Hamiltonian function, 242
  matrix, 362
Index

Helix function, 245
HELP (see also MATRX, CTRL-C, CONTROL.lab, IMPACT, and MATLAB), 372
Hermite form, 134
Hessenberg form, 34
Hessian matrix, 323
Hierarchical control, 285, 319
High-order systems, 124
Hyperplane, 243
Hypersurface, 243

I

IBM/PC, 28
Identification methods, 425
III-Conditioned matrix, 105
IMPACT, 371
Inner (scalar) product, 344, 345
Integrated CAD packages, comprehensive, 369
Interaction balance, 324
prediction, 324
variables, 322
Interactive mode, 369
operation, 430
Interface, 371
code-driven, 370
command-driven, 370
data-driven, 370
form-driven, 370
graphic-driven, 370
window-driven, 371
Interpreter, 434
INTRPRD (see also LSSPAK), 330
Inverse formulas, 351–54
of a matrix, 351
Inverted pendulum, 20, 233, 274
Iterative solution to decentralized output, 335

J

Jordan block, 120
form, 120
Jurry-Blanchard criterion, 89

K

Kalman filter, 12, 165
Kernel, 134

L

L-A-S, 434
eample of, 346
implementation of the LQG Problem, 440
language overview, 435
LQG/LTR, 434–38
matrix multiplication, 346
matrix inversion, 436
matrix transportation, 346
operator, 434
program, 434
Laplace transform, 25, 35
Large-scale systems, 285, 365
design methods, 285
tool box, 417
Leverrier-Faddeev algorithm, 27
Linear algebra, 342–58, 423
Linear differential equations, 19
Linear independence, 344
Linear independence, 57, 58, 92
Linear quadratic Gaussian, 13
regulator, 13, 177, 239
Linear spaces, 343, 344
systems, 19
time-invariant system, 18
time-varying system, 59
transformation, 345
Linearized model, 22, 66
Linearly dependent, 344
independent, 69
LINPACK, 259, 372
LISP, 453
Loop transfer recovery, 15, 165, 166
LQ problem, 363
LQR of PC-MATLAB, 257
LSSPAK/PC, 330, 417–19, 423
BALANC, 424
CHAIN, 424
COMSCS, 424
DYNCOM, 424
GOALCR, 424  
INTPRD, 424  
model reduction (MODRED), 424  
PADMOD, 424  
PADROUT, 424  
RICRKUT, 257  
ROBDEC, 424  
SIMUL, 424  
STAB, 424  
STRCON, 424  
STROBS, 424  
Luenberger observer, 13, 96, 141, 150  
Lyapunov function, 89  
    stability, 88

M

M files, 409, 415  
MACSYMA, 453  
Markov parameters, 140  
Mathematical definition, 231

MATLAB, 361
    tool boxes and extensions, 415  
    double-precision complex matrix, 361  
EXEC, 368  
EYE, 361  
GOTO, 368  
IF, 368  
LOGSPACE, 377  
ONES, 361  
SAVE/LOAD, 368  
SVD, 149  
WHILE, 368

MATLAB-based CACSD packages, 371  
MATLAB-deriven CACSD software programs, 364

Matrix Riccati equation, 240  
Matrix theory, 342  
MATR1Xx, 381, 384, 392, 401–2  
    AFEEEDBACK, 388  
    APPEND, 384, 388  
    BODE, 400  
    capabilities of, 381  
    capabilities: control design and system analysis, 384  
    capabilities: graphics, 382  
    capabilities: system descriptions and simulations, 383  
    capabilities: data MATR1Xx, analysis capabilities, 385  
    capabilities: system identification, signal processing, 385  
command file, 402  
CONNECT, 384, 389  
data transformations, 385  
spectral analysis, 385  
DEREGULATOR, 274  
DIAG, 391  
estimator, 391  
FEEDBACK, 384, 387  
filter design, 385  
functions, 402  
HILBM, 402  
integration algorithms, 385  
LIN, 400  
LSIM, 386  
macros, 402  
NORMAL, 386  
on an IBM PC/AT, 274  
PARALLEL, 384, 387  
PLOT, 401  
POLEPLACE, 386, 390  
RAND, 386  
RETF, 402  
SERIES, 384, 387  
SIM, 401  
SIN, 401  
structure of, 381  
system identification, 385  
SYSTEM-BUILD, 391  
three-dimensional plot, 382  
TIME, 401  
user routines, 402

Maximum principle, 247  
Mayne and Murdoch Formula, 116  
MIMO linear time-invariant systems, 383  
Minimal design problem, 14  
Minimal polynomial, 120, 349, 356  
Minimal realization, 77  
Minimal system, 99  
Minimum parameter data representation, 366  
Minimum principle, 235, 242
Minor of A, 352
Modal aggregation, 289, 296
Modal matrix, 296
Model, linearized, 22, 66
Modeling, 2
Modeling, analytical, 2
Modeling, experimental, 2
Modern control systems, 11
MS-DOS, 28
Multi-input multi-output (MIMO) systems, 365
Multivariable digital control system design, 417
MuMath, 453

N
Near-optimum design via aggregated models, 313
Necessary optimality condition for some typical problems, 248
Neural networks, 453
Non-MATLAB CACSD software packages, 364
Nonlinear systems, 406
time-varying systems, 89
Nonnegative definite matrices, 357
Nonsingular transformation, 24, 70
Normal smooth function, 243
Normal vector, 243
Norm of a matrix, 344–46
Nuclear reactor, 247
Null space, 134
Numerical solutions of matrix Riccati equations, 256
stability, 105
Nyquist criterion, 89

O
Observability, 56, 64, 96, 199, 306
index, 143, 148–49
Observable canonical form, 31, 48
Hessenberg form, 148
Observer design, 4, 139
Observer design, full-order, 139
reduced-order, 150
Observer-based controller, 156, 163
Operator statement, 434
Optimal control problem, 231
solution of, 235
design, 230
Optimal pole placement design, 278
Orthogonal matrices, 149, 356
transformation, 106, 107
Output feedback, 13, 174, 187, 208, 333
dynamic, 174, 187, 208
stabilization, 334
static, 174, 175

P
Parallel, 410
PC-MATLAB, 25, 409, 412
control tool box, primitives of, 411
commands, 410
ABCDCHECK, 414
DLSIM, 412
DSTEP, 412
EIG, 415
LQR, 412
NARGCHECK, 414
PLOT, 42
RAND, 412
TITLE, 412
XLABEL, 412
YLABEL, 412
PD controller, 8, 187
Performance index, 137, 230, 239
Perturbation, 286
regular, 287
singular, 287
PI controller, 8, 197
PID controller, 8, 207
Pole assignment (see also Pole placement), 13, 98–99
Pole placement:
high-order systems, 124
output feedback, 176
single-input, 113
multi-input, 119
Pole-zero cancellations, 71, 93
<table>
<thead>
<tr>
<th>Page</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>Poles, 49</td>
</tr>
<tr>
<td>27</td>
<td>Polynomial matrix, 27</td>
</tr>
<tr>
<td>103</td>
<td>Popov method, 103</td>
</tr>
<tr>
<td>90, 350, 351, 356-57</td>
<td>Positive definite matrices, 90, 350, 351, 356-57</td>
</tr>
<tr>
<td>306</td>
<td>Principle component analysis, 306</td>
</tr>
<tr>
<td>371</td>
<td>PRO-MATLAB, 371</td>
</tr>
<tr>
<td>368</td>
<td>Program information center (PIC), 368</td>
</tr>
<tr>
<td>429</td>
<td>Program structure of CADACS (KEDDC), 429</td>
</tr>
<tr>
<td>453</td>
<td>PROLOG, 453</td>
</tr>
<tr>
<td>187</td>
<td>Proportional plus derivative output feedback, 187</td>
</tr>
<tr>
<td>197</td>
<td>integral output feedback, 197</td>
</tr>
<tr>
<td>107</td>
<td>QR factorization, 107</td>
</tr>
<tr>
<td>90, 350, 351</td>
<td>Quadratic forms, 90, 350, 351</td>
</tr>
<tr>
<td>371</td>
<td>QUERY, 371</td>
</tr>
<tr>
<td>50, 61, 355</td>
<td>Rank, 50, 61, 355</td>
</tr>
<tr>
<td>77</td>
<td>Realization, 77</td>
</tr>
<tr>
<td>453</td>
<td>REDUCE, 453</td>
</tr>
<tr>
<td>243</td>
<td>Regular point, 243</td>
</tr>
<tr>
<td>5, 211</td>
<td>Regulation, 5, 211</td>
</tr>
<tr>
<td>13, 177, 239</td>
<td>Regulator, linear quadratic, 13, 177, 239</td>
</tr>
<tr>
<td>297</td>
<td>Residual state, 297</td>
</tr>
<tr>
<td>27</td>
<td>Resolvent matrix, 27</td>
</tr>
<tr>
<td>7, 34</td>
<td>Response to initial condition, 7, 34</td>
</tr>
<tr>
<td>input, 40</td>
<td>input, 40</td>
</tr>
<tr>
<td>164, 210</td>
<td>Return difference matrices, 164, 210</td>
</tr>
<tr>
<td>166</td>
<td>Riccati equation, 166</td>
</tr>
<tr>
<td>256</td>
<td>numerical solutions, 256</td>
</tr>
<tr>
<td>313</td>
<td>matrix, 313</td>
</tr>
<tr>
<td>8, 197</td>
<td>Controller, PI, 8, 197</td>
</tr>
<tr>
<td>8, 207</td>
<td>PID, 8, 207</td>
</tr>
<tr>
<td>22</td>
<td>Robot manipulator, 22</td>
</tr>
<tr>
<td>371</td>
<td>ROBOT-S, 371</td>
</tr>
<tr>
<td>418</td>
<td>Robotics tool box, 418</td>
</tr>
<tr>
<td>15, 366, 417</td>
<td>Robust control, 15, 366, 417</td>
</tr>
<tr>
<td>417</td>
<td>Robust control tool box, 417</td>
</tr>
<tr>
<td>417</td>
<td>$H_2$ and $H_\infty$ optimal control synthesis, 417</td>
</tr>
<tr>
<td>417</td>
<td>$H_2QG$, 417</td>
</tr>
<tr>
<td>417</td>
<td>HINF and LINF, 417</td>
</tr>
<tr>
<td>417</td>
<td>LQG, 417</td>
</tr>
<tr>
<td>417</td>
<td>LTRU and LTRY, 417</td>
</tr>
<tr>
<td>417</td>
<td>YOULA, 417</td>
</tr>
<tr>
<td>165</td>
<td>Robust observers, 165</td>
</tr>
<tr>
<td>162</td>
<td>Robustness with observers, 162</td>
</tr>
<tr>
<td>343</td>
<td>Row vector, 343</td>
</tr>
<tr>
<td>425</td>
<td>Ruhr-University Bochum, 425</td>
</tr>
<tr>
<td>453</td>
<td>Rule-based systems, 453</td>
</tr>
<tr>
<td>256</td>
<td>Runge-Kutta method, 256</td>
</tr>
<tr>
<td>93</td>
<td>Sampling, 93</td>
</tr>
<tr>
<td>353</td>
<td>Schur determinant identity, 353</td>
</tr>
<tr>
<td>258</td>
<td>Schur vector, 258</td>
</tr>
<tr>
<td>257</td>
<td>decomposition, 257</td>
</tr>
<tr>
<td>51</td>
<td>Schur's identity, 51</td>
</tr>
<tr>
<td>367</td>
<td>Scientific subroutine package (SSP), 367</td>
</tr>
<tr>
<td>324</td>
<td>Second-level coordinator problem, 324</td>
</tr>
<tr>
<td>307</td>
<td>Second-order modes, 307</td>
</tr>
<tr>
<td>9, 210</td>
<td>Sensitivity, 9, 210</td>
</tr>
<tr>
<td>13, 157</td>
<td>Separation principle, 13, 157</td>
</tr>
<tr>
<td>5</td>
<td>Servomechanism, 5</td>
</tr>
<tr>
<td>244</td>
<td>Set smooth k-fold, 244</td>
</tr>
<tr>
<td>34, 96, 101</td>
<td>Similarity transformation, 34, 96, 101</td>
</tr>
<tr>
<td>428</td>
<td>Simulation, 428</td>
</tr>
<tr>
<td>26, 365</td>
<td>Single-input single-output systems (SISO), 26, 365</td>
</tr>
<tr>
<td>354-58</td>
<td>Singular value decomposition theorem, 354-58</td>
</tr>
<tr>
<td>210, 358</td>
<td>values, 210, 358</td>
</tr>
<tr>
<td>343, 344</td>
<td>Spaces, linear, 343, 344</td>
</tr>
<tr>
<td>367</td>
<td>Sparse matrix techniques, 367</td>
</tr>
<tr>
<td>358</td>
<td>Square matrix, 358</td>
</tr>
<tr>
<td>308</td>
<td>Squaring up, 308</td>
</tr>
<tr>
<td>7, 56</td>
<td>Stability, 7, 56</td>
</tr>
<tr>
<td>80</td>
<td>in the sense of Lyapunov, 80</td>
</tr>
<tr>
<td>164</td>
<td>margin, 164</td>
</tr>
<tr>
<td>80</td>
<td>of linear systems, 80</td>
</tr>
<tr>
<td>80</td>
<td>zero-input, 80</td>
</tr>
<tr>
<td>85</td>
<td>zero-state, 85</td>
</tr>
<tr>
<td>257</td>
<td>Stabilizability, 257</td>
</tr>
<tr>
<td>209, 333</td>
<td>Stabilization, 209, 333</td>
</tr>
<tr>
<td>34</td>
<td>State equation, solution of, 34</td>
</tr>
<tr>
<td>18</td>
<td>State equations, 18</td>
</tr>
</tbody>
</table>
State estimator, 96, 139
  closed-loop, 141
  open-loop, 140
State feedback, 13, 96, 163
  regulator problem, 250
  space, 12, 209
  transformation, 24
  variable, 2
State-space, 18
Steady-state solution, 7, 8, 203, 218
Steepest descent method, 323
Subspace, 343
SUN workstation, 371
Superposition, 34
Supremal coordinator, 320
Survey of CACSD Packages, 451
Swiss Federal Institute of Technology in Zurich, 368
Sylvester equation (Criterion, theorem), 124, 350
Sylvester’s inequality, 358
Symmetric matrix, 90
System, balanced, 306
SYSTEM-BUILD, 371, 380, 381, 400, 412
System-ID tool box, 416
  ARMAX, 416
  ARX, 416
  BJ, 416
  IV, 416
  IV4, 416
  OE, 416
  PEM, 416
  PA, 416
Systems, classification, 2, 3
  continuous-time, 3
  definitions of, 2
  discrete-time, 3
  distributed-parameter, 2
  deterministic, 3
  linear, 3
  linear time invariant, 18
  linear time-varying, 59
  lumped-parameter, 2
  nonlinear, 3
  stochastic, 3
  time-invariant, 4
SYSTYEM-BUILD, capabilities of, 392

T
  Tangent hyperplane, 244
  plane, 245, 246
Tektronix's PLOT 10 IGL, 403
The control kit, 418
The discrete-time maximum principle, 267
Three-dimensional representation of a matrix, 375
Three-terms controller, 187
TI explorer, 453
TIMDOM/45:
  ROUTH, 420
  analysis, 422
  BODENY, 420
  CFAGGR, 423
  design, 421, 422
  estimation/filtering, 423
  EXACT, 423
  FULOBS, 423
  JURY, 420
  KALMAN, 423
  LISTRG, 422
  LYAP, 422
  MATCF, 421
  MODAL, 423
  model reduction, 422
  MOMENT, 421
  NEAROAP, 422
  PADE, 421
  PARA, 422
  POLPLA, 422
  REDOBS, 423
  RICATI, 422
  ROOTLC, 420
  SEPTIM, 423
  SOL, 420
  SOLI, 422
  SOL2, 422
  TPBVP, 422
  TRACK, 422
TIMDOM/PC, 28, 370, 418–21
  CON, 422
  OBS, 422
  RESMAT, 422
  STAB, 422
Time domain (see also State-space methods), 365
Index

Time-domain analysis, 19
Toeplitz matrix, 115, 139
Tool boxes, 409, 418
Total stability, 88
TPBV (see also Two-point boundary-value), 247
Trace of a matrix, 28
Tracking, 5, 209, 210, 211
Tracking system design, 216
Transfer function matrix, 18
Transformation, orthogonal, 345
Transition matrix, 34
Transmission zero, 50, 51, 165, 166, 219, 441
Transversality condition, 243, 246
Two-terms controller, 187
Two-point boundary-value (TPBV) problems, 247

U

Uniform asymptotic stability, 80–83
Unitary transformation, 345
Unity rank, 119, 187

V

Vandermonde matrix, 357
VAX environment, 409
Vector space, 343
Vector transversal, 245
VMS, 409

W

Well-conditioned, 105, 137
Window-driven interface, 371

Z

Zero-input stability, 80
Zero-state stability, 85
Zeros, 49