

Computer Controlled Systems

International Series on

MICROPROCESSOR-BASED SYSTEMS ENGINEERING

VOLUME 8

Editor

Professor S. G. Tzafestas, *National Technical University, Athens, Greece*

Editorial Advisory Board

Professor C. S. Chen, *University of Akron, Ohio, U.S.A.*

Professor F. Harashima, *University of Tokyo, Tokyo, Japan*

Professor G. Messina, *University of Catania, Catania, Italy*

Professor N. K. Sinha, *McMaster University, Hamilton, Ontario, Canada*

Professor D. Tabak, *Ben Gurion University of the Negev, Beer Sheva, Israel*

The titles published in this series are listed at the end of this volume.

Computer Controlled Systems

Theory and Applications

by

GEORGE A. PERDIKARIS

*University of Wisconsin-Parkside, Kenosha,
Wisconsin, U.S.A.*



Springer Science+Business Media, LLC

Library of Congress Cataloging-in-Publication Data

Perdikaris, George A.

Computer controlled systems : theory and applications / George A. Perdikaris.

p. cm. -- (Microprocessor based systems engineering ; v. 8)
Includes index.

1. Automatic control. 2. Microprocessors. I. Title.

II. Series: International series on microprocessor-based systems engineering ; v. 8.

TJ223.M53P43 1991

629.8'95--dc20

91-29108

ISBN 978-90-481-4100-5

ISBN 978-94-015-7929-2 (eBook)

DOI 10.1007/978-94-015-7929-2

Printed on acid-free paper

All Rights Reserved

© 1991 Springer Science+Business Media New York

Originally published by Kluwer Academic Publishers 1991

Softcover reprint of the hardcover 1st edition 1991

No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission from the copyright owner.

CONTENTS

PREFACE	ix
Chapter 1: INTRODUCTION	1
1.1 OVERVIEW	1
1.2 CONTINUOUS (ANALOG) CONTROL SYSTEMS	2
1.2-1 Open-Loop Control Systems	2
1.2-3 Closed-Loop (Feedback) Control Systems	3
1.3 DISCRETE (DIGITAL) CONTROL SYSTEMS	5
1.3-1 Open-Loop Digital Control Systems	6
1.3-2 Closed-Loop Digital Control Systems	7
1.4 INTERFACE DEVICES	9
1.4-1 Analog-to-Digital Converter (ADC)	9
1.4-2 Digital-to-Analog Converter (DAC)	12
1.4-3 Incremental and Absolute Encoders	14
1.5 IMPLEMENTATION CONSIDERATIONS	16
1.5-1 Sampling Frequency Selection	16
1.5-2 Quantization Errors and Effects	17
1.5-3 Computer Wordlength	18
1.5-4 Computational Delays	18
Chapter 2: CONTINUOUS-TIME SYSTEMS	19
2.1 INTRODUCTION	19
2.1-1 Continuous Singularity Functions	21
2.1-2 Systems Analysis and Design	24
2.2 DIFFERENTIAL EQUATIONS	25
2.2-1 Electrical Networks	26
2.2-2 Mechanical Systems	29
2.2-3 Control Systems	30
2.3 TIME RESPONSE USING CLASSICAL METHODS	32
2.3-1 Solution of Homogeneous Differential Equations	32
2.3-2 Solution of Nonhomogeneous Differential Equations	40
2.4 THE LAPLACE TRANSFORM	53
2.4-2 TABLES OF LAPLACE TRANSFORMS	58

2.4-3	Partial Fraction Expansion.....	64
2.5	TIME RESPONSE USING TRANSFORM METHODS.....	70
2.6	TRANSFER FUNCTIONS AND BLOCK DIAGRAMS.....	77
2.6-1	Transfer Functions and Differential Equations.....	78
2.6-2	Transfer Functions of Electrical Networks.....	79
2.6-3	Transfer Functions of Simple Block Diagrams.....	81
2.6-4	Block Diagram Reduction.....	85
2.6-5	Steady-State Error Analysis.....	92
2.7	TRANSFER FUNCTIONS AND SIGNAL FLOW GRAPHS.....	94
2.7-1	Signal Flow Graph Definitions.....	95
2.7-2	Mason's Gain Formula.....	96
2.8	CONTINUOUS SYSTEMS POLES, ZEROS, AND STABILITY.....	102
2.8-1	Routh's Stability Criterion.....	105
2.8-2	Root Locus Diagram.....	110
2.9	FREQUENCY RESPONSE CONCEPTS.....	114
2.9-1	Frequency Response Plots.....	114
2.9-2	Frequency Plots of the Open-Loop Transfer Function.....	125
2.9-3	Correlation of Transient Time and Frequency Responses.....	128
2.10	DIGITAL SIMULATION METHODS.....	131
2.10-1	System Simulation by Programming Block Diagrams.....	132
2.10-2	System Simulation Using ACSL.....	134
2.10-2	System Simulation Using SIMULAB.....	135
Chapter 3:	DISCRETE-TIME SYSTEMS.....	139
3.1	INTRODUCTION.....	139
3.1-1	Discrete Singularity Functions.....	141
3.2	DIFFERENCE EQUATIONS.....	144
3.3	TIME RESPONSE USING CLASSICAL METHODS.....	147
3.3-1	Solution of Homogeneous Difference Equations.....	148
3.3-2	Solution of Nonhomogeneous Difference Equations.....	153
3.4	SAMPLING THEORY.....	157
3.4-1	The Zero-Order-Hold (ZOH) Element.....	164
3.5	THE Z TRANSFORM.....	167
3.5-1	Z Transform Theorems.....	168
3.5-2	Tables of Z Transforms.....	173
3.5-3	Method of Long Division.....	176
3.5-4	Partial Fraction Expansion.....	179
3.6	TIME RESPONSE USING Z TRANSFORM METHODS.....	182
3.7	TRANSFER FUNCTIONS AND BLOCK DIAGRAMS.....	186
3.7-1	The Pulse Transfer Function.....	187
3.8	DISCRETE SYSTEMS POLES, ZEROS, AND STABILITY.....	201
3.8-1	Bilinear Transformation.....	205
3.8-2	Routh's Stability Criterion.....	206
3.8-3	The Jury Stability Criterion.....	208
3.8-4	Root Locus Diagram for Discrete systems.....	211
3.9	FREQUENCY RESPONSE OF DISCRETE SYSTEMS.....	212
3.10	DIGITAL SIMULATION AND Z TRANSFORMS.....	218
3.10-1	Numerical Differentiation and Integration.....	218
3.10-2	Numerical Solution of Differential Equations.....	227
3.10-3	Simulation of Discrete Systems Using SIMULAB.....	231
Chapter 4:	STATE VARIABLE METHODS.....	235
4.1	INTRODUCTION.....	235

4.2	STATE SPACE CONCEPTS CONTINUOUS SYSTEMS	236
4.2-1	State Transition Equation	241
4.2-2	State Equations and Transfer Functions.....	249
4.2-3	From Transfer Functions to State Equations	250
4.2-4	Phase-variable Canonical Form	255
4.3	STATE SPACE CONCEPTSDISCRETE SYSTEMS	258
4.3-1	Solution of Discrete State Equations.....	261
4.3-2	Discrete State Equations and Transfer Functions	264
4.3-3	From Discrete Transfer Functions to State Equations	265
4.4	STATE EQUATIONS DISCRETE FROM CONTINUOUS.....	271
4.6	CONTROLLABILITY AND OBSERVABILITY.....	275
4.7	CASE STUDY ANALYSIS OF A MOTOR DRIVE SYSTEM.....	280
Chapter 5:	DIGITAL SIGNAL PROCESSING	291
5.1	INTRODUCTION	291
5.2	DIGITAL FILTER CONCEPTS	292
5.3	ANALOG FILTER CONCEPTS	295
5.3-1	Butterworth Filter Forms	296
5.3-2	Binomial Filter Forms.....	299
5.3-4	Other Types of Analog Filters.....	300
5.4	DESIGN OF IIR DIGITAL FILTERS	302
5.4-1	IMPULSE INVARIANCE METHOD	302
5.4-2	Step Invariance or Zero-Order-Hold (ZOH) Method	303
5.4-3	Pole-Zero Matching Method.....	304
5.4-4	Bilinear Transformation Method	305
5.5	DESIGN OF FINITE IMPULSE RESPONSE (FIR) FILTERS	311
5.5-1	Lagrangian Multipliers Method.....	312
5.5-2	Least-Squares Method.....	314
5.5-3	FIR Filters for Differentiation, Smoothing, and Prediction.....	315
5.6	DIGITAL FILTER IMPLEMENTATION CONSIDERATIONS.....	316
Chapter 6:	COMPUTER CONTROLLED SYSTEMS	322
6.1	INTRODUCTION	322
6.2	OPEN-LOOP CONTROL Deadbeat Algorithms	323
6.3	THE LAPLACE TRANSFORM METHOD	324
6.3-1	First-Order Plant	324
6.3-2	Second-Order PlantOverdamped Case	326
6.3-3	Second-Order Plant Underdamped Case.....	330
6.3-4	"Positive Cast" or "Posicast" Control	334
6.3-5	Second-Order Plant Critically-Damped Case	335
6.4	THE Z TRANSFORM METHOD	337
6.5	THE STATE VARIABLE METHOD.....	340
6.5-1	The nth-Order Plant Case	343
6.6	CLOSED-LOOP CONTROL Deatbeat Algorithms	348
6.6-1	The Plain (Free) Integrator Plant.....	349
6.6-2	First-Order Plant Case.....	351
6.6-3	Second-Order Plant Case	353
6.6-4	The nth-Order Case.....	356
6.7	DEADBEAT ALGORITHMS FOR RAMP INPUTS	358
6.7-1	The Plain Integrator Plant.....	359
6.7-2	First-Order Plant Case.....	361
6.8	ALGORITHMS FOR CONTROLLING DISTURBANCES.....	365
6.8-1	The Z Transform Method	366
6.8-2	The State Variable Method	370

6.9	A CONTROL SCHEME FOR REFERENCE AND/OR DISTURBANCE.....	374
6.9-1	The Plain Integrator Plant Step Inputs.....	378
6.9-2	The First-Order Plant Step Inputs.....	379
6.9-3	The Plain Integrator Plant Ramp Inputs.....	381
6.10	SIMULATION OF COMPUTER CONTROL SYSTEMS	383
6.10-1	Simulation Languages.....	383
Chapter 7:	COMPUTER CONTROLLED SYSTEMS	392
7.1	INTRODUCTION.....	392
7.2	THE ANALOG AND DIGITAL PID ALGORITHMS.....	395
7.3	THE ANALOG AND DIGITAL PDF AND PDFF ALGORITHMS	397
7.3-1	PDFF Control of the Plain Integrator Plant.....	401
7.3-2	PDFF Control of the Double Integrator Plant	403
7.3-3	Tuning of Control Algorithms by Standard Filter Forms	403
7.4	CASE STUDY DIGITAL CONTROL OF A MOTOR DRIVE PLANT	408
7.4-1	Velocity and Position Plant Models.....	408
7.4-2	Digital Control of Velocity and Position Plants	410
7.4-3	Filtering the Derivative (K_d) Path.....	417
7.4-4	A Second-Order Derivative (K_j) in the Feedforward Path	419
7.4-5	Time and Frequency Analysis of Digital Control Systems.....	421
7.4-6	Steady-State Error Analysis of Digital Control Systems	425
7.5	PHASE-LEAD CONTROL ALGORITHMS.....	428
7.6	CONTROL ALGORITHMS FOR PLANTS WITH DEAD TIME	430
7.7	ZIEGLER-NICHOLS TUNING METHOD FOR PID ALGORITHMS.....	433
7.8	DIGITAL CONTROLLER IMPLEMENTATION CONSIDERATIONS.....	436
Chapter 8:	SYSTEM IDENTIFICATION.....	440
8.1	INTRODUCTION.....	440
8.2	System Identification A Time Domain Approach.....	440
8.2-1	Identifying the Discrete Transfer Function $G(z)$	441
8.2-2	Determining $G(s)$ from $G(z)$	443
8.2-3	Identifying the Pulse Transfer Function $G_hG(z)$	447
8.3	A MODEL REDUCTION SCHEME	448
8.3-1	Model Reduction from Transfer Function Models.....	448
8.3-2	Model Reduction from a State Space Model.....	451
8.4	CASE STUDY REDUCED-ORDER MODELS FOR A MOTOR DRIVE.....	455
APPENDIX A:	REFERENCES.....	461
APPENDIX B:	REAL-TIME PROGRAMS	464
INDEX.....		474

Preface

The primary objective of the book is to provide advanced undergraduate or first-year graduate engineering students with a self-contained presentation of the principles fundamental to the analysis, design and implementation of computer controlled systems. The material is also suitable for self-study by practicing engineers and is intended to follow a first course in either linear systems analysis or control systems.

A secondary objective of the book is to provide engineering and/or computer science audiences with the material for a junior/senior-level course in modern systems analysis. Chapters 2, 3, 4, and 5 have been designed with this purpose in mind. The emphasis in such a course is to develop the mathematical tools and methods suitable for the analysis and design of real-time systems such as digital filters. Thus, engineers and/or computer scientists who know **how** to program computers can understand the mathematics relevant to the issue of **what** it is they are programming. This is especially important for those who may work in engineering and scientific environments where, for instance, programming difference equations for real-time applications is becoming increasingly common. A background in linear algebra should be an adequate prerequisite for the systems analysis course.

Chapter 1 of the book presents a brief introduction to computer controlled systems. It describes the general issues and terminology relevant to the analysis, design, and implementation of such systems.

Chapter 2 lays the foundation of systems analysis using the more familiar continuous concepts. It develops and applies the classical and Laplace transform methods to analyze, model, and simulate systems, both in time and frequency domains. Many of the systems analyzed in Chapter 2 are designed in later chapters as practical control systems.

Chapter 3 parallels the development of Chapter 2, but for discrete-time or discrete systems. The important theories of sampling and $\mathcal{Z}$ transforms are introduced and applied to model, simulate, and analyze discrete systems. Numerical integration and differentiation methods are developed because of their important role in discretizing continuous filters and controllers.

Chapter 4 is devoted to the topic of state-space techniques. The mathematics of modern systems theory are presented and integrated for continuous and discrete systems. The state equations are formulated and solved using time as well as transform techniques. The problem of obtaining discrete state equations from continuous is also considered, with emphasis on algorithms suitable for computerization.

The discrete and continuous mathematics of Chapters 2, 3, and 4 are used throughout the book. Furthermore, these mathematics form the necessary background for advanced studies in areas such as adaptive control, optimization theory, and for literature investigation.

Chapter 5 presents methods for transforming analog prototype filters into digital filters by applying mathematical tools developed in Chapters 2, 3, and 4. The design and implementation of digital filters is viewed as an intermediate step between systems in general and computer controlled systems in particular. The types of filters studied are used in later chapters in the design of digital control systems.

Chapters 6 and 7 deal with the digital control problem. These chapters deal with the design and implementation of algorithms used in various control schemes or strategies. Transform as well as state variable methods are employed so that the reader can associate the new material with what he/she already knows. This approach adds to the reader's circumspection of the entire computer control problem.

The coverage of **Chapter 6** emphasizes the computer control of plants for which a mathematical model is known or can be readily obtained. The concepts of open-loop and closed-loop control using finite-settling-time algorithms, also called deadbeat algorithms, are

presented. A general scheme for controlling plants when reference and/or external disturbance inputs enter the control loop is developed. Applications are illustrated by digital simulation as well as by laboratory measurement.

Chapter 7 deals with the computer control of plants for which a mathematical model may or may not be readily available. It presents and applies the proportional-integral-derivative (PID), and the integral with proportional-derivative feedback (PDF) control schemes. The PDF control scheme with added feedforward control action (PDFF) is also considered. The control of more complex processes as well as applications emphasizing the computer control of motor drive systems are included.

Chapter 8 presents a practical method that can be used to identify a process by a suitable mathematical model from its transient response. The problem of mathematical model reduction by a systematic approach is also included.

The **Appendices** include sample programs for real-time signal processing and control applications. Information about author's programs used in this book is also included.

The book was written by the author on an Apple Macintosh computer using Microsoft's **Word**. The block diagrams were drawn using Deneba System's **Canvas**, and most of the plots were generated using Synergy Software's **Kaleidagraph**.

The experimental charts of Chapter 6 were recorded in the laboratory. These charts were subsequently scanned and included in the manuscript. The plant models were programmed on an analog computer which was controlled by a digital microcomputer.

The experimental results of Chapter 7 were plotted from data "captured in real-time" as the industrial drives were controlled by a VME-based Motorola 68020 microcomputer. The real-time programs are written in the C language using cross-compiler software by Intermetrics.

Some computer simulations were performed using the Advanced Continuous Simulation Language (ACSL) by Mitchell & Gauthier Associates, and some using MATLAB and SIMULAB by The MathWorks. Most of the system analysis and design results were obtained using the author's own set of programs. Data from these programs can be plotted using a spreadsheet or a plotting program such as Kaleidagraph. For information and availability of either an IBM PC or an Apple Macintosh version of the author's programs, please contact the author.

This book project has been influenced by many technical works and people. Of those works listed in the references, special thanks to the "pioneer" textbook writers C. L. Smith, and particularly Cadzow & Martens on whose deadbeat control ideas much of Chapter 6 is based. My special thanks to my esteemed colleague Dr. A.S. Rao, whose model reduction scheme is highlighted in Chapter 8. Many thanks to Dr. Ron Brown and Tim Ruchti of Marquette University for their valuable suggestions and contributions, and to Dr. Tom Higgins, University of Wisconsin-Madison, and Dr. Spyros Tzafestas, National Technical University of Athens, for prodding me to complete this book.

I am indebted to Tom Beck, Bill Hammel, Mike Lucas, Keith VanPatten, Mark Sevey and many others at UNICO, Inc., for their continued support of UW-Parkside's Real Time Systems (RTS) laboratory and related research projects.

I would also like to thank the following people for their helpful comments and encouragement over the years: Chuck Rohrer, TechniSoft; Ken Swanson and Gerald O'Callaghan, Eaton-Dynatomic; Rabah Rennane and Tom Rehm, Allen-Bradley; Dr. Steven Bomba, Johnson Controls; and Bill Erickson and David Gerlach, Rexroth-Indramat. Thanks to all of my colleagues at UW-Parkside and especially to Dr. Walter Feldt, Paul Sorensen, Tom Labus, Padmini Varanasi, and to James Raymaker who literally "manned" the controls in the RTS laboratory during the data collection phase of the book.

Dedicated to Tia, Christina, and Tommy (Θαναῶνς)



'Concern for man himself and his fate must always form the chief interest of all technical endeavors - in order that the creations of our mind shall be a blessing and not a curse to mankind.'

Albert Einstein

INTRODUCTION

1.1 OVERVIEW

Industrial automation incorporating digital computers is becoming increasingly important in the production of goods and services. The continued progress in computer hardware and software has helped make computer control of machines and processes a technological and economic reality.

The rising manufacturing and material costs have created a major problem of cost control for many firms. As with most cost control problems, the basic factor involved in the solution is the need for increased productivity. With overhead and materials accounting for most costs, industries realize that they must strive to manufacture as efficiently as possible.

In addition to the cost-control problem, industries find it necessary to install and operate complex systems, to meet strict governmental regulations regarding product liability and pollution control, and to emphasize product value and quality in order to survive and remain competitive.

The processing and monitoring capabilities of modern-day computers have played a vital role in influencing the industry's decision to increase the use of robots and other computerized equipment in place of conventional control instrumentation. Modern factory automation systems incorporate the technology known as **computer integrated manufacturing (CIM)**, whose basic premise is efficient manufacturing. What this means is that manufacturers will be able to bring new products to the market faster, keep inventories low, make products of consistently high quality, have flexible facilities which allow for the manufacture of a variety of products, and improve communication within the company as well as with its customers and suppliers.

Industrial automation systems often involve the application of computers to process information and control various machines and processes in real time. For instance, a microcomputer may be used to control the speed of an automobile, or the temperature and/or humidity of an office building. For these and other applications the main concern is to develop and implement solutions that meet problem requirements in a cost-effective manner.

This book presents mathematical tools and methods necessary for modeling, analyzing, designing, simulating, and implementing real-time systems. The tools and methods are applied to design and implement digital filters and practical computer controlled systems. Results obtained analytically are verified by both computer simulation and laboratory experimentation.

1.2 CONTINUOUS (ANALOG) CONTROL SYSTEMS

By definition, a **system** consists of components organized to accomplish specific goals or objectives. A typical objective of a physical system is to transform incoming signals by mathematical operations in order to produce desired results. A system can be represented graphically by a block diagram as shown in Fig. 1.2-1. In a **block diagram**, each component of the system is represented by its own block. The flow and direction of information from one component to the next is shown by arrows.

In Fig. 1.2-1, x represents the **input** signal and y the **output** signal. If the input and output signals are continuous functions of time, represented by $x(t)$ and $y(t)$, respectively, both the signals and the system are called **continuous** or **analog**. On the other hand, if the input and output signals are defined only at discrete instants of time, then the input and output signals are represented by $x(kT)$ and $y(kT)$, respectively, and both the signals and the system are called **discrete** or **digital**.

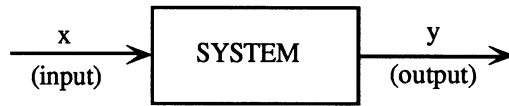


Fig. 1.2-1 Block diagram representation of a system.

In some cases, the system output is proportional to the input. Then the behavior of the system is not time-dependent and such systems are called **nondynamic**. For instance, if a system can be represented by an arbitrary constant K , the above block diagram can be described or modeled by the equation

$$y = Kx \quad (1.2-1)$$

Thus, if $K=1$, the output equals the input, etc.

If the input signal is corrupted by noise, for another example, then the system in Fig. 2.1-1 can be a **filter**, which is designed to eliminate noise or to at least minimize its influence. In such cases, the system output cannot change instantaneously and its behavior depends on time. These systems are called **dynamic**.

1.2-1 Open-Loop Control Systems

In general, the successful operation of a system requires some form of **control**. In other words, if a system cannot attain established goals, we can modify its performance by controlling it.

Consider the system shown in Fig. 1.2-2, for instance, where the operation performed by the system is integration. If the input signal is a **step** function, defined as a function having a value of zero (0) before it is switched on at $t=0$, and a value of one (1)

immediately after it is switched on, the integrated system output will be a function which increases with time as shown in the figure. This function is called a **ramp** function.

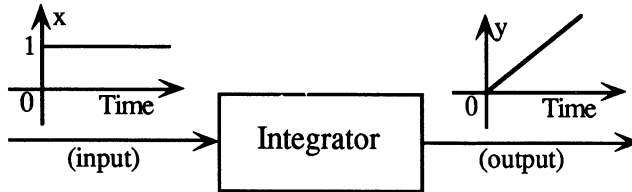


Fig. 1.2-2 Block diagram of an integrator system.

Systems such as the integrator of Fig. 1.2-2 are common in the real world, but the performance characteristics shown in Fig. 1.2-2 are usually unacceptable. The behavior of such systems can be made acceptable by either open-loop control or closed-loop control. In either case, acceptable performance usually implies that the output of the control system follows or tracks the step function input with some delay or lag, not instantaneously.

There are two basic configurations of control systems; the **open-loop** control system, and the **closed-loop** control system. A block diagram of an open-loop control system is shown in Fig. 1.2-3. The system to be controlled is the **plant** or **process**, where plant refers to machines or equipment, and process refers to a system of operations. For instance, a motor is a plant, while a chemical operation is a process. The plant/process output represents the variable to be controlled or maintained at a desired level, called the **controlled variable**. The overall input to the control system is called **reference** or **setpoint** and represents the desired or target value set as a goal by the system designer. The **control law** or **algorithm**, also called the **controller**, is designed to make the plant meet problem specifications, even in the presence of external **disturbances**. For each controlled variable, there is a corresponding **manipulated variable** or **controller output**. An example of an open-loop control system is a space heater, which stays on regardless of the room or outdoor temperature.

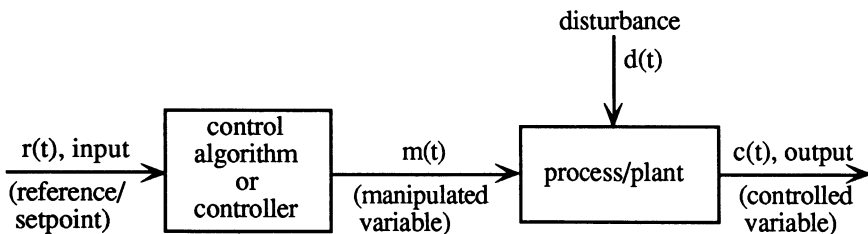


Fig. 1.2-3 Block diagram of an open-loop control system.

1.2-3 Closed-Loop (Feedback) Control Systems

An open-loop control system is simple and economical. However, in an open-loop control system, there is no direct connection between the controlled variable and the controller. Thus, the control system is not affected by the controlled variable which can

deviate from the desired input value due to external disturbances and possible parameter variations. To adjust the manipulated variable so that the difference or **error** between the desired input and the controlled output is minimized or eliminated, we can use **closed-loop** or **feedback control**. A block diagram of a closed-loop control system is shown in Fig. 1.2-4.

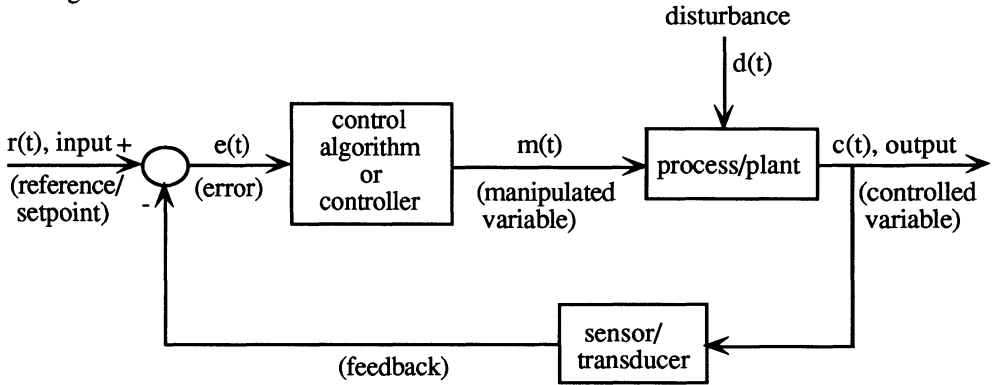


Fig. 1.2-4 Block diagram of a closed-loop control system.

From Fig. 1.2-4, the **sensor/transducer** component is used to measure the actual value of the controlled variable called **feedback**. The feedback is then compared with the desired reference value or setpoint to form the algebraic difference or error. Based on the error value, the controller calculates a manipulated variable value which is sent to the plant. The objective of the controller is to correct the error, even in the presence of disturbances. The corrective action taken by the controller continues until the error is zero, within design specifications. When this occurs, the desired/target input value is equal to the actual output value, which is the goal of the control system.

A home heating system is an example of a closed-loop control system. The temperature of the house is the controlled variable, and the furnace is the plant to be controlled. The controller senses the difference between the thermostat setting (reference input value) and the actual house temperature (feedback), and determines whether to turn the furnace on or off. The controller's objective is to maintain constant temperature, corresponding to the thermostat setting, in spite of changes in outside weather conditions or disturbances.

Closed-loop control systems are commonly classified by the type of corrective action the controller takes when it senses a deviation of the controlled variable from the reference input. The **single-loop** control system shown in Fig. 1.2-4 is considered the standard or canonical form for a control system. Depending on the objectives of the application, the controller block can take a variety of mathematical representations.

A popular controller is the **proportional-integral-derivative (PID)** controller, whose output combines proportional, integral, and derivative control actions as shown in Fig. 1.2-5. The **controller gains** K_P (**proportional**), K_I (**integral**), and K_D (**derivative**) are adjusted or tuned to achieve certain objectives. In some cases, only the proportional (P) control action is needed; more often, however, both the proportional and integral control actions (PI) are required; and there are cases where all three control actions (PID) must be used.

Another practical controller is the **PDFF** controller shown in Fig. 1.2-6. This controller includes integral control action in the forward path, like the PID controller, but

its proportional (P) and derivative (D) control actions are in the feedback (F) path. In addition, the PDFF controller has **feedforward** (F) control action, thus the name PDFF. Like the PID controller, the PDFF controller can also be operated in a variety of ways. In addition, as we will see in Chapter 7, the PDFF controller can be operated as a PID controller.

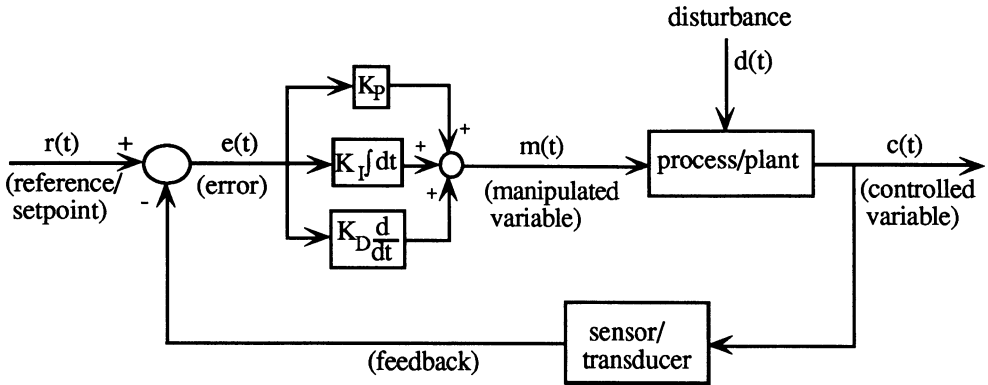


Fig. 1.2-5 Block diagram of PID control system.

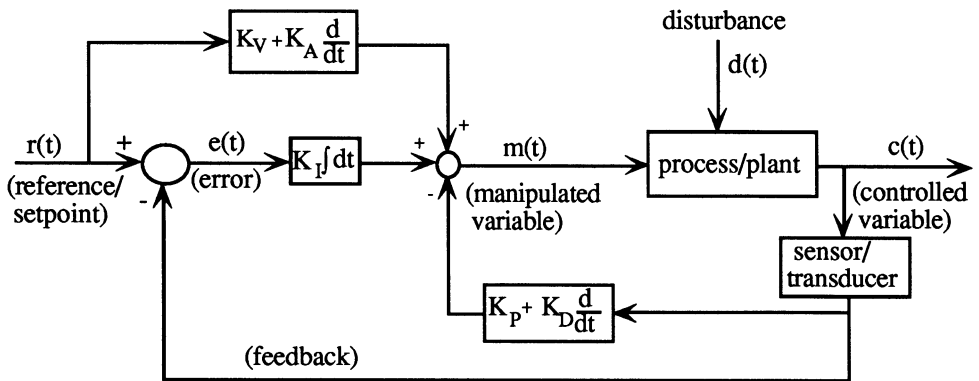


Fig. 1.2-6 Block diagram of PDFF control system.

1.3 DISCRETE (DIGITAL) CONTROL SYSTEMS

Controllers or filters modeled by continuous mathematics can be implemented by electrical and electronic devices, such as resistors, capacitors, inductors, and operational amplifiers, and the resulting system is referred to as **analog** or conventional. In the past, filters and control algorithms were implemented by analog hardware. However, the increased speed and decreased size and cost of modern microprocessors have permitted their economical use to implement filters and controllers in real time.

The software implementation offers several advantages. First, it offers flexibility, because the characteristics of the filter or control law can be easily changed by simply reading in a new set of parameter values. Second, a software implementation results in a more reliable system, because there are no physical components to age or deteriorate with temperature variations. This in turn implies consistency in system performance, and independence from bulky component problems associated with analog realizations. Finally, a greater degree of accuracy can be attained by programming equations, because the parameters are controlled by the designer who can set them to obtain an efficient and cost-effective system that meets problem specifications.

1.3-1 Open-Loop Digital Control Systems

In a conventional control system, the controller performs calculations like an analog computer. When the calculations dictated by the controller are performed by a digital computer, the plant is actually controlled by the computer which implements the control algorithm. The overall system is thus a **computer controlled system**, also called a **computer/digital control system**, or a **direct digital control (DDC) system**.

Controlling a process by a computer usually implies that the computer is an integral part of a feedback control system. In this configuration, the computer samples the response of the process at specific instants of time and compares its findings with prescribed values stored internally. Any deviation from the desired behavior initiates corrective action as dictated by the control algorithm, also stored in the computer. Digital control systems can be open loop or closed loop.

An open-loop digital control system is shown in Fig. 1.3-1. From the diagram, the computer input $r(kT)$ must be a number, and the computer output $m(kT)$ can only be a number. Thus, the need for the interface devices, in this case an **analog-to-digital converter (ADC)**, and a **digital-to-analog converter (DAC)**. If the reference input signal is continuous (analog), it must be changed to discrete (digital) form by the ADC device before it is sent to the computer. On the other hand, if the plant is analog, the output of the computer or the controller must be changed from discrete (digital) to continuous (analog) form before it is sent to the plant.

In the case of a digital filter implementation, the process/plant block in Fig. 1.3-1 can be removed, and the disturbance (noise) input can be combined with the signal as shown in Fig. 1.3-2. The objective of the **digital filter** (computer) is to remove the noise from the signal+noise combination by performing appropriate calculations. The output of the filter can then be displayed, if desired, via the DAC interface.

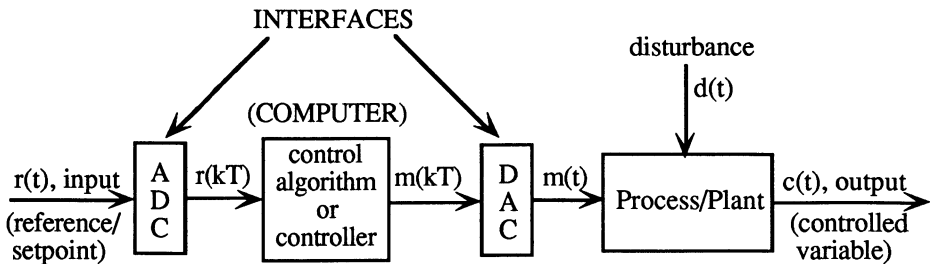


Fig. 1.3-1 Block diagram of an open-loop digital control system.

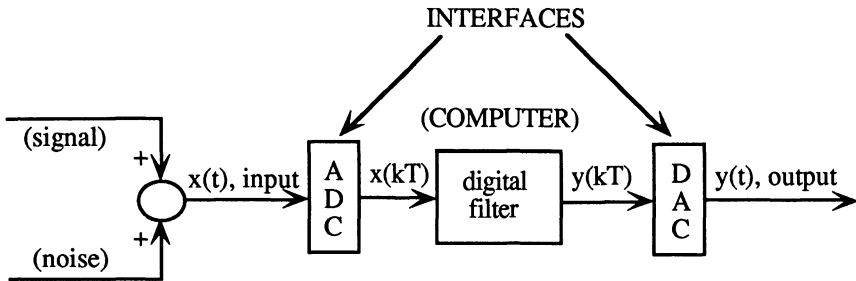


Fig. 1.3-2 Block diagram of a digital filter system.

1.3-2 Closed-Loop Digital Control Systems

A closed-loop digital control system is shown in Fig. 1.3-3. It can be seen from the diagram that ADC and DAC interfaces have been assumed, and that the ADC location in this control system implies that the error signal is calculated in the computer, along with the control algorithm. As in the case of the analog closed-loop control system, a common objective of the controller is to minimize the error $e(kT)$, even in the presence of disturbances.

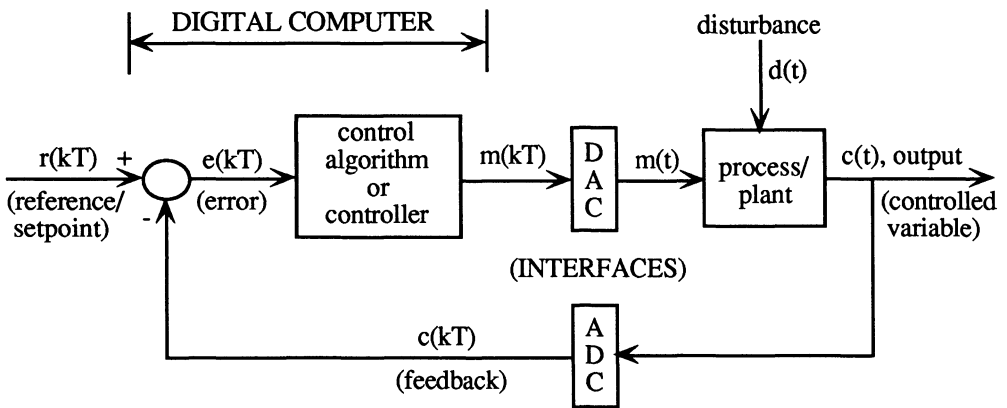


Fig. 1.3-3 Block diagram of a closed-loop digital control system.

The analog PID and PDFF closed-loop control systems considered earlier can be converted into digital counterparts by adding appropriate interfaces, and by converting the dynamic operations of continuous differentiation and integration to equivalent discrete forms. As an example, consider the digital PID closed-loop control system shown in Fig. 1.3-4. Again, ADC and DAC interfaces have been assumed, and the $\mathcal{Z}$ transform notation has been employed to represent the numerical integration and differentiation operations.

Observe that the proportional, integral, and derivative gains have lower-case

subscripts, compared with the analog PID controller system where the subscripts are upper case. This convention is followed throughout the book.

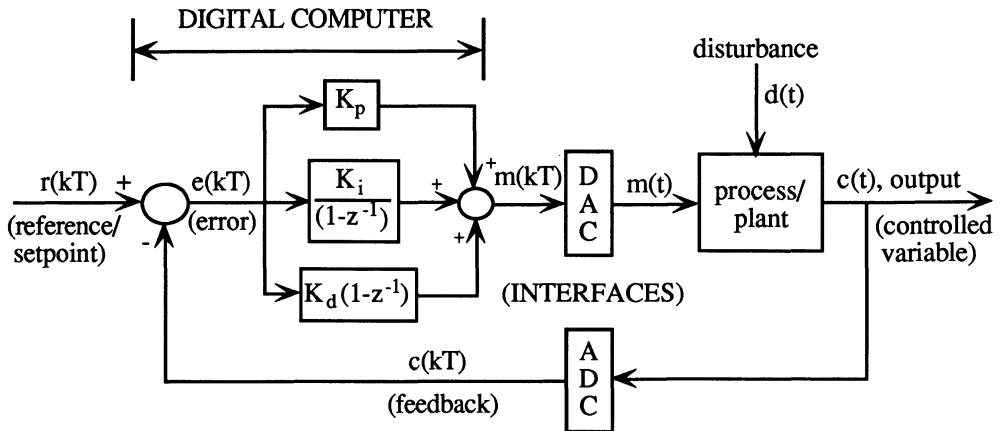


Fig. 1.3-4 Block diagram of a digital PID control system.

Note that in addition to the ADC and DAC interfaces, the symbol T has been added in the diagrams representing digital filters or digital control systems. In order to convert analog signals into a sequence of samples (numbers), so that they can be processed by the computer, it is necessary to pick a spacing or interval for the samples. This time interval between successive samples is called the **sampling time** or **sampling period** and is denoted by T . In the case of the closed-loop control system, the controller reads the ADC every T seconds, performs its calculations, and outputs its value to the plant via the DAC. It is assumed that the calculation time is small relative to the sampling time T , otherwise the performance of the digital system may not be satisfactory.

Control systems, analog or digital, are sometimes classified either as regulators or servomechanisms. A **regulator** control system is required to maintain the controlled variable at a constant value in spite of external disturbances. The reference input is also set to constant value (setpoint). A home heating control system is an example of a regulator. The objective of the control algorithm is to maintain the temperature constant regardless of the outside temperature. The reference input value is the constant setting of the thermostat. The cruise control system of a car is another example. Its objective is to maintain relatively constant speed, set by the driver, in the presence of disturbances such as driving uphill or downhill.

A **servomechanism**, on the other hand, is required to track or follow a time-varying input such as a sinusoidal, polynomial, or trapezoidal function, often referred to as **profile**. The controlled output of a servomechanism is usually the position, velocity, or acceleration of a motor shaft. The velocity profile during acceleration and deceleration of a car by the driver is an example of a servomechanism, the control system of a robotic arm is another.

1.4 INTERFACE DEVICES

In general, each component of a system can be represented by a mathematical model that describes the relationship between its input and output variables. In the time domain, continuous systems can be represented by differential equations or continuous state space equations and discrete systems by difference equations or discrete state space equations. In the frequency domain, on the other hand, differential equations are converted to continuous transfer function models by the Laplace transform, and difference equations are converted to discrete transfer functions by the $\mathcal{Z}$ transform.

A digital control system is usually designed to meet performance specifications in a cost-effective manner. To verify this assumption, one can model the overall system and analyze it using analytical and computer-aided techniques. If the system does not meet the problem specifications, appropriate modifications can be made and the iterative process continued until the performance of the system is deemed satisfactory.

To simulate a digital control system realistically, appropriate mathematical models for each block, including the interface devices, must be found.

1.4-1 Analog-to-Digital Converter (ADC)

In a digital control system, the plant output or controlled variable is usually an analog signal which is sampled every T seconds and approximated by a sequence of numbers. Each number is then encoded into a binary word using a finite number of binary digits (bits). The conversion of an analog signal into a corresponding digital (binary) signal is called **quantization**, and the device that performs this function is called **analog-to-digital converter**, usually abbreviated as ADC or A/D.

Quantization is an approximation process, because the number of bits or wordlength of the ADC device is limited. Equivalently, the amplitude of the analog signal being sampled is represented inside the computer with a digital word of finite resolution. Hence, the analog-to-digital conversion operation involves **quantization error**, and the uncertainty that results from it is called **quantization noise**. Typical ADC devices have wordlengths of 10 to 12 bits, resulting in relatively small quantization errors for most applications.

To illustrate the previous concepts consider a 3-bit ADC, with an input voltage range (VR) of ± 10 volts, that is, a full dynamic range of $(10+10)=20$ volts.

With a wordlength of 3 bits, the number of levels available for quantizing the analog signal is $2^3=8$. Thus, the interval between successive levels, called the **quantization level** or **quantization step size**, is given by

$$q = \frac{(10+10)}{2^3} = \frac{20}{8} = 2.5 \text{ volts}$$

We can view the output of the ADC, call it $x(kT)$, as the sum of the actual input $x(t)$, and a random error (noise) $e(kT)$, called the **quantization error**, whose value is bounded between $\pm q/2$. That is,

$$x(kT) = x(t) + e(kT) \quad (1.4-1)$$

$$\text{and} \quad -\frac{q}{2} < e(kT) < \frac{q}{2} \quad (1.4-2)$$

The objective is to model the function of devices such as the ADC so that we can use it properly when simulating digital filters or digital control systems. Towards this objective, the ADC is shown represented in the block diagram of Fig. 1.4-1.

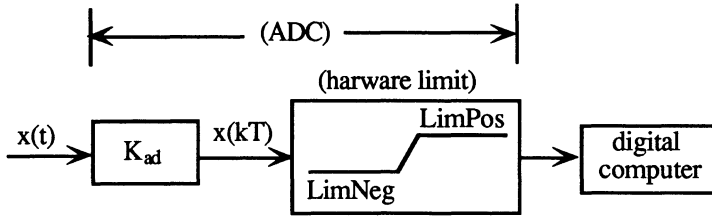


Fig. 1.4-1 Block diagram of an analog-to-digital converter (ADC) and computer.

The limiter is part of the hardware, but is useful when simulating the ADC function on a digital computer. For a two's complement, n-bit ADC, the positive and negative limit settings are given by

$$\text{LimPos} = 2^{(n-1)} - 1, \text{ and } \text{LimNeg} = -2^{(n-1)} \quad (1.4-3)$$

Then, when writing a simulation program, the following set of statements

IF $x(kT) > \text{LIMPOS}$ THEN $x(kT) = \text{LIMPOS}$
 IF $x(kT) < -\text{LIMNEG}$ THEN $x(kT) = -\text{LIMNEG}$

implements the hardware limiter. The parameter K_{ad} represents the **gain or scale** of the ADC device and is given by

$$K_{ad} = \frac{2^n}{VR} \quad (1.4-4)$$

where VR is the full dynamic range of the ADC. For $n=3$ and $VR=20$, for instance, the ADC gain is

$$K_{ad} = \frac{8}{20} = \frac{1}{q} = 0.4$$

Then, for an analog signal of $x(t)$ volts, the ADC output number, **rounded** to the nearest level, is given by

$$x(kT) = \text{round}[x(t) * K_{ad}] \quad (1.4-5)$$

Specifically, if $x(t)=5$ volts, the ADC output, in both decimal as well as in binary for a 3-bit word, is

$$x(kT) = \text{round}(5 * 0.4) = 2_{10} = 010_2$$

and if $x(t)=3$ volts, then

$$x(kT) = \text{round}(3 * 0.4) = 1_{10} = 001_2$$

To accommodate positive as well as negative values, that is, for a bipolar ADC, let's assume two's complement representation. To form the **two's complement** of an n-bit number, leave all least significant zeros and the first nonzero digit unchanged, and replace 1's by 0's and 0's by 1's in all other higher significant digits. For instance, for the 3-bit case, the decimal value -2 is written in binary as -010. Using the suggested rule, we obtain the two's complement representation 110.

The relationship between the analog input and digital output for a 3-bit, ± 10 volt, two's complement ADC is shown in Fig. 1.4-2. Note that the two's complement notation allows one extra level in the negative direction (-10), but the 3-bit ADC using this notation cannot represent a $+10$ volt signal value, because $\text{LimPos} = 2^{(3-1)} - 1 = 3_{10} = 011_2$ or 7.5 volts.

If the ADC is a 12-bit, ± 10 volt, two's complement device, the quantization step is

$$q = \frac{(10+10)}{2^{12}} = \frac{20}{4096} = \frac{10}{2048} = 0.00488 \text{ volts}$$

and the gain is

$$K_{ad} = \frac{2048}{10} = 204.8$$

Thus, if the analog input value is $x(t) = 2.5$ volts, the ADC output is found to be $x(kT) = \text{round}(2.5 * 204.8) = 512_{10} = 001000000000_2$.

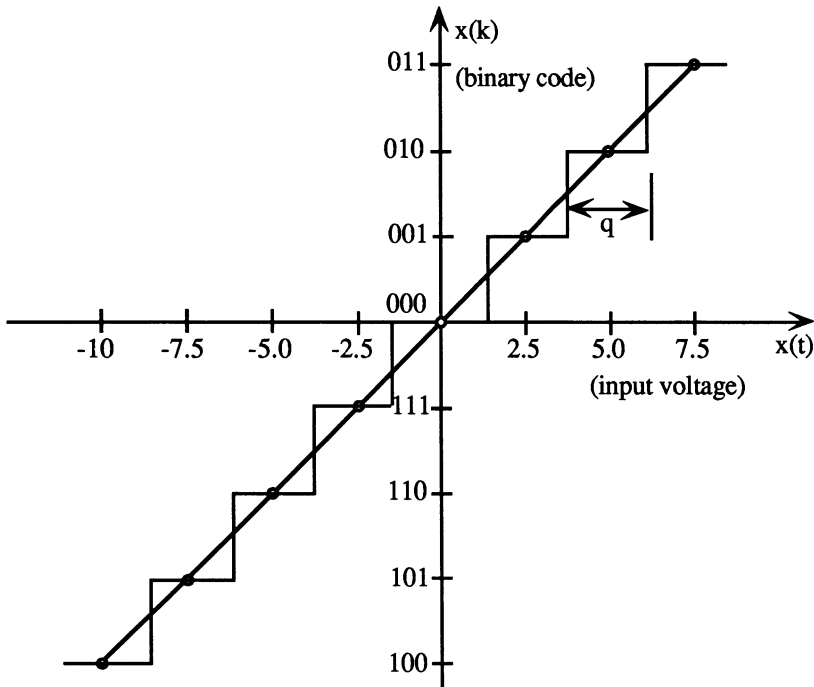


Fig. 1.4-2 Input voltage versus binary output for 3-bit, ± 10 volt ADC using two's complement notation.

1.4-2 Digital-to-Analog Converter (DAC)

The output of the digital controller (manipulated variable) is in digital form and must be converted to analog form before it can be communicated to the analog plant. The device that accomplishes this is called a **digital-to-analog converter**, abbreviated as DAC or D/A. A block diagram representation of a DAC is shown in Fig. 1.4-3.

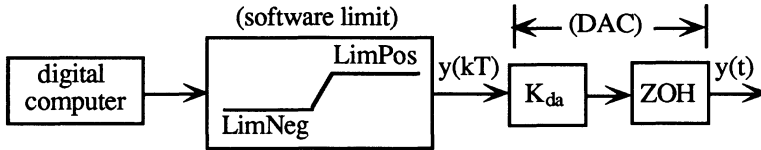


Fig. 1.4-3 Block diagram of a digital-to-analog converter (DAC) and computer.

The "software" limiter in Fig. 1.4-3 must be programmed by the designer the same way the ADC limiter was implemented earlier. The parameter K_{da} represents the **gain** or **scale** of the DAC device and is given by

$$K_{da} = \frac{VR}{2^n} \quad (1.4-6)$$

where VR is the full dynamic range of the DAC. For the DAC operation, there is a one-to-one correspondence between the digital (binary) signal $y(kT)$ and the analog output $y(t)$.

The sequence of numbers that represent the digital signal inside the computer is converted into an analog signal by a polynomial extrapolator called **zero-order-hold (ZOH)**. The function of the ZOH is to hold a numerical value constant (zero-order polynomial) until the next one arrives. Thus, a digital signal input to the DAC appears as a staircase function at its output.

Assume, again, a 3-bit, ± 10 volt DAC, using the two's complement notation. Then, the quantization level q , and the DAC gain K_{da} are given by

$$K_{da} = q = \frac{VR}{2^3} = \frac{20}{8} = 2.5 \text{ volts}$$

For instance, if the binary number is $y(kT)=001$, the DAC output is $y(t)=001*2.5=2.5$ volts. Also, if $y(kT)=011$, then $y(t)=011*2.5=7.5$ volts, etc.

The relationship between the digital input and analog output for the 3-bit, ± 10 volt, two's complement DAC is shown in Fig. 1.4-4.

If the DAC is a 12-bit, ± 10 volt, two's complement device, the gain is

$$K_{da} = \frac{20}{4096} = 0.00488$$

Thus, the 12-bit number $y(kT)=001000000000_2=512_{10}$ results in the output voltage value

$$y(t) = 512 * K_{da} = 2.5 \text{ volts}$$

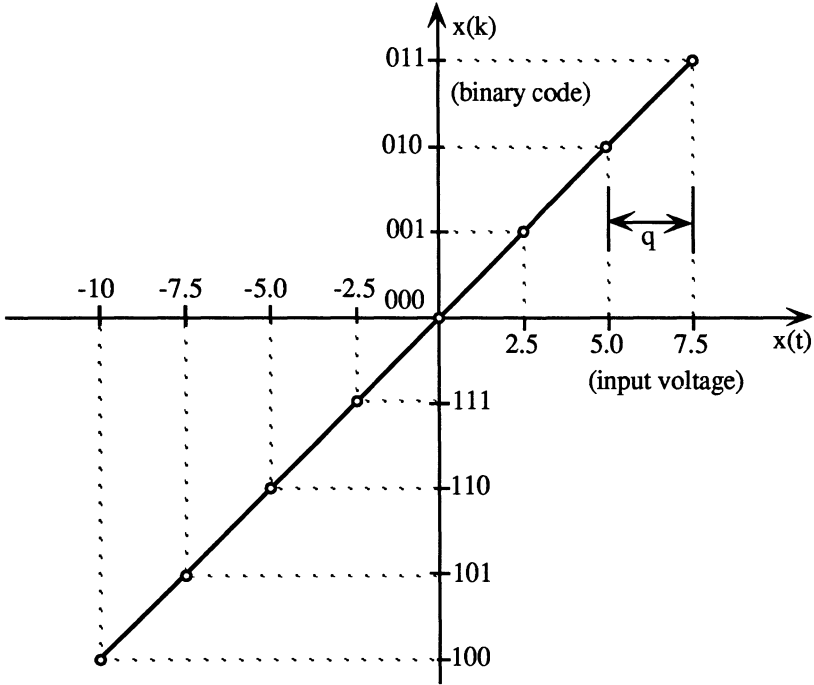


Fig. 1.4-3 Input voltage versus binary output for 3-bit, ± 10 volt DAC using two's complement notation.

To estimate the number of bits B needed to represent a value of D decimal digits, a useful rule can be derived from the quantization relationship

$$\frac{1}{2^B} \leq \left(\frac{1}{10^D} \right), \text{ or } 2^B \geq 10^D \quad (1.4-7)$$

Taking logarithms and solving for B , leads to the result

$$B \geq 3.32D \quad (1.4-8)$$

This means that for 3 decimal places we need 10 bits resolution, and for 4 decimal places 14 bits of resolution are required.

ADC and DAC devices can be classified as unipolar or bipolar, depending on their voltage range capability. They are also classified according to the binary code used to represent numbers, that is, two's complement, binary offset, etc., their number of bits of resolution, and the method or technology employed to carry out their functions and build the devices themselves.

1.4-3 Incremental and Absolute Encoders

In many industrial applications, it is common to control the speed (velocity) and/or position (displacement) of the shaft of a motor drive plant (motor and its power supply) by a digital computer. Since the controlled variable is a mechanical quantity, a transducer is needed to convert shaft rotation into a signal that can be read by the computer. A transducer that can accomplish this is the analog tachometer (tach) generator, which converts shaft rotation into an analog voltage signal. An ADC can then be used to sample the velocity signal and communicate it to the digital controller for appropriate action. This approach leads into "drift" problems when the shaft position signal is reconstructed inside the computer by integrating the velocity signal obtained from the analog tach.

An alternative approach is to use an optical encoder, in the place of the ADC, to sense angular velocity and/or position. A common optical encoder type is the **incremental encoder**, also known as **motor pulse generator** (MPG) or just **pulse generator** (PG). The shaft velocity is determined by counting pulses from a device register every T seconds, where T is the control sampling time. The pulses come from an up/down counter and represent information about the speed and rotation of the motor shaft.

The signal from the pulse generator is proportional to the change in position from the previous sample. Therefore, the pulse generator acts as a differentiator. A block diagram model of a pulse generator is shown in fig. 1.4-4, where K_{dt} represents the gain of the device (dt for digital tach), and the **truncated** value $y(kT)$ is subsequently differentiated.

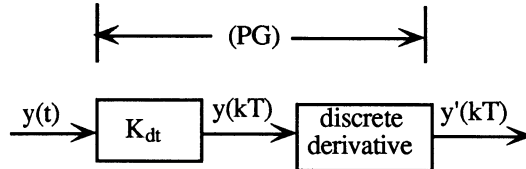


Fig. 1.4-4 Block diagram of an incremental encoder or pulse generator (PG).

For instance, if the input to the PG represents position/displacement in radians (rad), the units of K_{dt} are (pulses/rev) or (pulse/ 2π rad). Hence, the units of $y(kT)$ are pulses, and after differentiation we obtain the velocity signal $y'(kT)$ in (pulses/T). Specifically, a 2000 pulses/rev PG, has a gain given by

$$K_{dt} = \frac{\text{\#pulses}}{\text{rev}} = \frac{2000 \text{ (pulses)}}{2\pi \text{ (rad)}} \quad (1.4-9)$$

An **absolute encoder** accumulates pulses and monitors motor shaft position or displacement, instead of velocity. Shaft position information can also be obtained by integrating the output of the incremental encoder inside the computer.

The ADC, DAC, and encoder interface devices are found in problems and/or examples involving modeling, simulation, and design of computer controlled systems. An industrial control system incorporating motors, computers, and typical interfaces is considered in the following example.

Example 1.4-1 An Industrial Automation System

A scanned picture of a computer controlled system is shown in Fig. 1.4-5a. A block diagram of the physical system is shown in Fig. 1.4-5b. This is a coil processing "sort-and-feed" system, designed to produce cut-to-length and cut-to-hole parts for various industrial applications. As can be seen from the block diagram, the system incorporates encoder interfaces, motor drive plants and microcomputers for implementing the control algorithms. Each motor drive has its own micro-based controller, while a central computer is used to "supervise" the operation for production efficiency and to store information for management use.

When the actual control and management functions are not concentrated in a single location, but distributed throughout the plant, as shown in Fig. 1.4-5, the system is called a **distributed control system**. Implementing distributed control systems implies sophisticated software for both data communication and control.

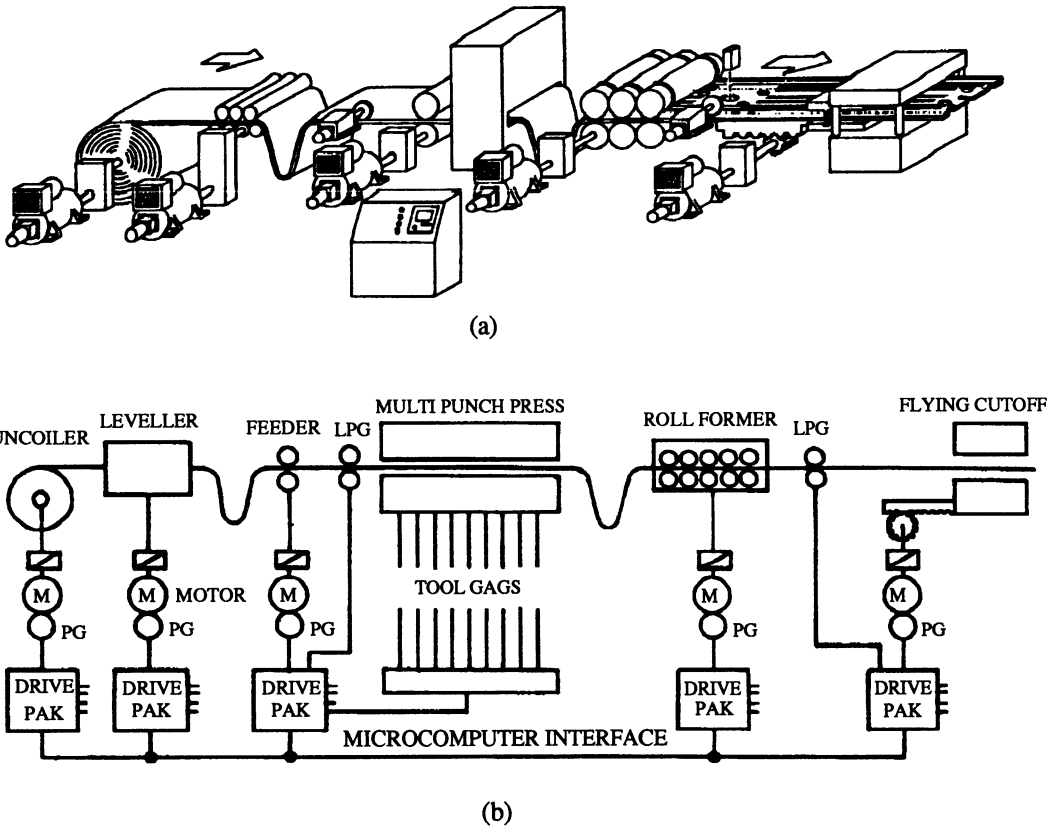


Fig. 1.4-5 (a) Picture of a "sort-and-feed" computer controlled system and (b) its block diagram. (Courtesy UNICO Inc., Franksville, Wisconsin.)

1.5 IMPLEMENTATION CONSIDERATIONS

As discussed earlier, a programmed controller is flexible, reliable, and its accuracy can be set by the designer to meet problem objectives. However, the software implementation has certain limitations. The sampling time, computational requirements, including speed and integer or floating-point arithmetic, the wordlength of the computer used to program the digital algorithm, and the wordlength of the interface devices affect the performance of the control system and must be properly coordinated.

The design of digital controllers is often carried out by first designing an analog prototype controller that meets the problem requirements for the analog control system. Once this is accomplished, the analog controller transfer function (math model) is converted to the corresponding digital controller transfer function by some approximation technique. The motivation for this approach is that successful designs of analog controllers are well established, and that the designer's main concern is to find the discrete equivalent of the analog controller so that the performance of the digital control system resembles that of its analog counterpart.

An alternative method for designing digital controllers is to carry out the design directly in the discrete time or discrete frequency domain using appropriate techniques. The advantage of this approach is its directness, because the performance of the digital system depends on the exact placement of the poles and zeros which specify the system transfer function model. The disadvantage is the difficulty facing a designer who wishes to visualize the exact location of the z -plane poles and zeros in order to achieve response characteristics that meet problem requirements.

1.5-1 Sampling Frequency Selection

Regardless of the controller design method followed, the choice of an appropriate sampling time or, inversely, **sampling rate** is of utmost importance. A relatively large sampling time is desirable because it allows ample time for the computations required by the control algorithm. This, in turn, implies that a relatively slow microcomputer and associated interfaces, such as ADCs and DACs, can be used for a low-cost system. On the other hand, a small sampling time (large sampling rate) requires a much faster computer, usually with longer wordlength and more expensive interfaces. In general, a system designer must select a sampling time that balances the cost of the system with its performance characteristics.

In addition to the computer cost problem associated with the selection of the sampling time, there are also other important factors that must be considered by the designer if a digital control application is to succeed. When designing a controller, the ability of the control system to follow or **track** a given type of reference input is of prime importance. In terms of the control system shown in Fig. 1.3-3, tracking the reference input $r(kT)$ by the controlled variable $c(t)$ implies that $r(kT)$ must be reconstructed by the plant $G_p(s)$. Then, according to the **sampling theorem**, to reconstruct the input signal $r(t)$ from its samples $r(kT)$, "the sampling rate ($1/T$) must be at least twice as high as the highest frequency contained in $r(kT)$ ". Thus, since the control system must track $r(kT)$, the highest frequency in $r(kT)$ corresponds to the desired **bandwidth** of the closed-loop control system.

The bandwidth of the control system criterion provides a lower limit on the selection of the sampling rate. Practical considerations dictate, however, that the sampling rate must be much higher than twice the bandwidth. "Rules-of-thumb" suggest a sampling rate that is 10-20 times larger than the bandwidth of the closed-loop system. A

good way for choosing a sampling rate is to use these suggestions as guidelines and experiment with various sampling rates until an acceptable one is determined.

When designing a digital controller from an analog prototype, it is important that the performance characteristics of a digital control system resemble those of the analog control system as closely as possible. In order for this to happen, the control system sampling time must be relatively small compared to the time constant of the plant that dominates its transient response, called the **dominant time constant**. If this basic rule is not observed, the **aliasing** (distortion) effects that result may lead to degraded performance for the digital control system.

Selecting a sampling time that is small relative to the dominant time constant of the plant is not straightforward when an exact model of the plant is not readily available. This suggests the possibility of selecting T by some kind of an iterative scheme, for instance, by making an educated guess for a starting value of T and keep changing it, up or down, until an acceptable compromise has been achieved.

Computer control algorithms are usually designed to perform well when an external **disturbance** in the form of **noise** or a **load change** enters the control loop, perhaps in addition to the reference input. Since the disturbance may consist of signals of various frequencies, it becomes important to estimate their range. Like $r(kT)$, the disturbance signal can be interpreted as another input in the control system. This implies that the sampling rate of the control system must be high in relation to the highest frequency of the disturbance signal. Otherwise, the sampling theorem is violated and deterioration of system performance will occur.

1.5-2 Quantization Errors and Effects

In designing digital algorithms for either filtering or control purposes, we often assume that the resulting transfer functions can be characterized by their coefficients which are represented accurately using as many decimal places as needed. Then, to determine whether a digital system meets design specifications, we perform conventional time/frequency domain analysis, assuming adequate computational accuracy because we use floating-point arithmetic.

In many real-time applications, **fixed-point (integer) arithmetic** is used because it results in faster and more economical implementations. A fixed-point implementation implies that the coefficients of the algorithm, reference input, and internal data must be represented by a finite number of bits, or a finite **wordlength**. If the representation is not exact, there will be quantization errors which must be properly handled by the system designer.

The first type of error is caused by quantizing the feedback signal using devices such as the ADC, that is, by using a finite number of bits to represent the output samples. One of the factors that should be considered when selecting an ADC is the noise (error) level present in the analog signal. The wordlength of the ADC must be such that its quantization error does not exceed the error in the signal of a transducer.

The second type of error occurs when the result of an addition or multiplication operation is rounded in order to fit in a prescribed wordlength. To avoid the first and second type of errors, the wordlengths of the interface devices, as well as of the digital computer used for fixed-point operations, must be sufficiently large and coordinated.

A third type of error results when the coefficients of the algorithm are represented by a finite number of bits. If the encoding of the coefficients is not accurate enough, the resulting system may exhibit characteristics which differ substantially from those of the original design.

1.5-3 Computer Wordlength

Microcomputers have limited wordlengths, usually 8 or 16 bits. Therefore, the error input and the manipulation output of a digital controller are quantized and the effects of the quantization on the performance of the system must be investigated. Lack of sufficient accuracy for coefficient encoding and/or internal calculations may lead to persistent oscillations, referred to as **limit cycle**. The limit cycle phenomenon is characterized by a periodic output, even though there is no input causing it.

Care must also be exercised, particularly with smaller wordlength computers, so that **arithmetic overflow** does not occur. At the very least, overflow should be monitored, and the calculation results should be limited to prevent it.

1.5-4 Computational Delays

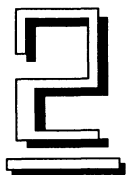
To implement a digital filter/controller in **real-time**, it is necessary that the microcomputer is fast enough so that the arithmetic calculations dictated by the algorithm can be performed well within the sampling time interval. If the **computational delay** is too large relative to the sampling time, it can have adverse effects on the performance of the digital system. The computational delay can be determined experimentally, for instance, by turning a signal on when entering the digital algorithm and turning the signal off when the calculations are finished. Once the computational delay is known, it can then be handled analytically by the system designer.

PROBLEMS

- 1.1 Identify a few control systems from your home and/or workplace environments and classify them as open-loop or closed-loop. What interfaces would be needed to convert these system to digital control systems? Discuss the pros and cons of "to computer control or not to computer control."
- 1.2 Write a computer program, in any high-level language, to simulate the ADC and DAC interfaces. Sample the function $f(t)=\sin 2\pi t$, using a sampling time $T=0.01$ sec from $t=0$ to $t=1$ sec. Then, plot and compare the samples obtained when the ADC and DAC interfaces have the following resolution: (a) 12 bits, (b) 10 bits, (c) 8 bits, (d) 6 bits, and (e) 4 bits. Assume voltage ranges of ± 10 volts.
- 1.3 For the function of Problem 1.2, calculate the root-mean-square (RMS) error between the given $f(t)$ and the quantized/sampled value for each ADC/DAC resolution case. If the sampled value of $f(t)$ is denoted by $x(kT)$, with $T=0.01$ sec, the RMS error for N samples is given by

$$\text{RMS} = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} [f(t_i) - x(iT)]^2}$$

Plot the RMS error versus the number of bits of resolution from $t=0$ to $t=1$ sec.



CONTINUOUS-TIME SYSTEMS

Systems, Signals, and Models

2.1 INTRODUCTION

A **system** consists of interrelated components or parts, called **subsystems**, organized to accomplish certain goals or objectives. For instance, a single-input-single-output (SISO) system carries out its objectives by transforming the incoming signal or **input** into a suitable **output** signal or **response**. The transformation process usually implies manipulation of the incoming information by performing mathematical operations such as addition, subtraction, multiplication, differentiation and integration.

The term system can imply various types of systems including physical, medical, business and environmental. Some of these systems are linear, others nonlinear; some have single inputs and/or outputs, others have multiple inputs and/or outputs. A single-stage RC network, for example, is a physical system whose purpose may be to remove high-frequency noise from signals. A university, as another example, is an educational system with related inputs, subsystems and objectives.

Systems are classified according to certain characteristics or properties. To define these characteristics, consider a SISO system represented by the block diagram shown in Fig. 2.1-1.

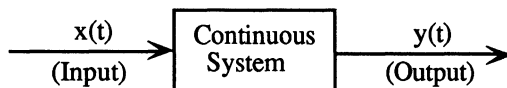


Fig. 2.1-1 Block diagram model of a continuous system.

The system **input (excitation)** is denoted by $x(t)$ and the **output (response)** is denoted by $y(t)$. Both $x(t)$ and $y(t)$ are **time signals**, because their value is a function of the independent variable time, denoted by t . If a system is at rest, that is, if no internal energy in the form of initial conditions is present before the input is applied, the input-output transformation relationship may be represented by the functional notation

$$y(t) = P[x(t)] \quad (2.1-1)$$

where P (for Processing) is a transformation operator that characterizes the system.

A **signal** is defined as **continuous-time**, or just **continuous**, within a range, if its value is specified for all t in that range, except possibly at a finite number of discontinuities. The signal shown in Fig. 2.1-2 is an example of a continuous signal in the range $a \leq t \leq b$. However, the function $f(t)$ is not continuous for $a \leq t \leq b$, because of the discontinuities at $t=t_1$ and $t=t_2$. Thus, the mathematical definition of a continuous signal is less constrained than the definition of a continuous function.

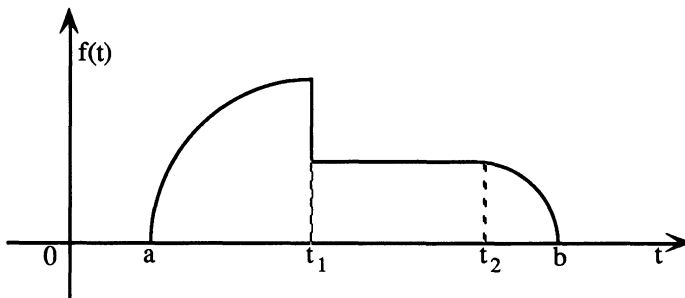


Fig. 2.1-2 Continuous signal, discontinuous function.

A **continuous-time (continuous) system** is a system whose inputs and outputs are continuous signals.

A system is **homogeneous**, if its response to the input

$$x(t) = ax_1(t) \quad (2.1-2)$$

is the response of the system to the input $x_1(t)$ multiplied by 'a', for all values of $x_1(t)$ and any constant 'a'. That is, for a homogeneous system

$$y(t) = P[ax_1(t)] = aP[x_1(t)] \quad (2.1-3)$$

A system is said to be **linear** if its response to the input given by

$$x(t) = ax_1(t) + bx_2(t) \quad (2.1-4)$$

$$\text{is} \quad y(t) = P[ax_1(t) + bx_2(t)] = aP[x_1(t)] + bP[x_2(t)] \quad (2.1-5)$$

for all values of the signals $x_1(t)$ and $x_2(t)$, and for any constants 'a' and 'b'. Equivalently, a system is linear if, and only if, the transformation operator P is linear.

The property expressed by Eq. (2.1-5) is also known as the **principle of superposition**; it implies that the system responds homogeneously to two independent inputs which are applied simultaneously.

A system is called **deterministic** if all of its parameters are completely defined. Conversely, if one or more parameters are not known exactly the system is called **nondeterministic**.

A system is **realizable** or **causal** if it does not respond before an input is applied. For a realizable system, if the input $x(t)$ is zero for $t < 0$, then the output $y(t)$ is zero for

$t < 0$. A **nonrealizable (noncausal)** system, on the other hand, implies the existence of an output even before an input is applied.

A system is called **time-invariant** or **constant-parameter** if the response to a time-shifted input signal is a time-shifted output signal. That is, for a time-invariant system

$$P[x(t-T)] = y(t-T) \quad (2.1-6)$$

for any values of time t and delay or shift T . Thus, for a time-invariant system, the output depends on the form or shape of the input, but not on its time of origin.

2.1-1 Continuous Singularity Functions

The analysis and design of systems is often facilitated by the use of a class of time functions known as **singularity** functions. These functions and their derivatives are continuous everywhere except at a single point of discontinuity. The singularity functions are used to describe input or test signals containing discontinuities and can be generated from one another by successive differentiation or integration.

Perhaps the most familiar of the singularity functions is the **unit-step** function defined as

$$u(t) = \begin{cases} 1 & \text{for } t \geq 0 \\ 0 & \text{for } t < 0 \end{cases} \quad (2.1-7)$$

A plot of the unit-step function is shown in Fig. 2.1-3a. As can be seen from the figure, the unit-step function is a continuous-time function whose value is zero for negative time and unity for positive time, including the point of discontinuity at $t=0$.

A **delayed unit-step** function is a unit-step function displaced by T seconds (sec) to the right; its origin is shifted from $t=0$ to $t=T$. This function is described by

$$u(t-T) = \begin{cases} 1 & \text{for } t \geq T \\ 0 & \text{for } t < T \end{cases} \quad (2.1-8)$$

Its plot is shown in Fig. 2.1-3b. In general, the argument $(t-T)$ of the unit-step function $u(t-T)$ is zero at the point of discontinuity. It is positive when the step function is unity, and negative when the step function is zero. Thus, a unit-step function that starts at $t=-T$ and moves to the right is described by

$$u(t+T) = \begin{cases} 1 & \text{for } t \geq -T \\ 0 & \text{for } t < -T \end{cases} \quad (2.1-9)$$

while a unit-step function that starts at $t=T$ and moves to the left is given by

$$u(T-t) = \begin{cases} 1 & \text{for } t \leq T \\ 0 & \text{for } t > T \end{cases} \quad (2.1-10)$$

Plots of the functions of Eq. (2.1-9) and Eq. (2.1-10) are shown in Fig. 2.1-3c and Fig. 2.1-3d, respectively.

Nonunity or arbitrary-size step functions can be constructed by simply pre-multiplying the unit-step function by a constant of desired magnitude and sign. Step functions can be combined with other step functions or with other types of functions in order to synthesize waveforms convenient for the analysis and testing of systems. The unit-area rectangular **pulse** function shown in Fig. 2.1-4a, for instance, can be formed

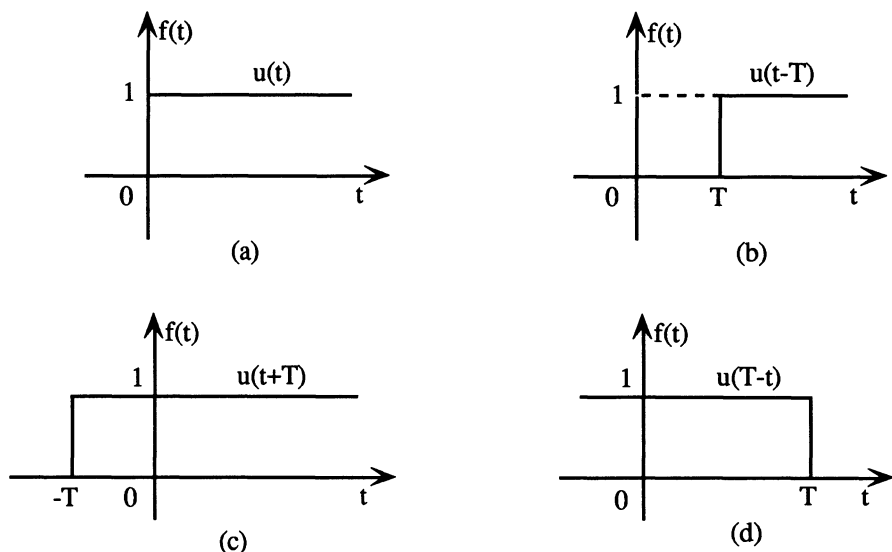


Fig. 2.1-3 The unit-step function (a) and its various shifted forms.

by subtracting two step functions, and thus, may be described by the analytical expression

$$f(t) = \frac{u(t) - u(t-T)}{T} \quad (2.1-11)$$

Alternatively, the pulse function may be obtained by multiplying two step functions and is expressed by

$$f(t) = \frac{u(t)u(T-t)}{T} \quad (2.1-12)$$

For another example, the cycle of a sine wave defined for $0 \leq t \leq T$ as shown in Fig. 2.1-4b can be represented by

$$f(t) = [u(t) - u(t-T)] \sin \pi t \quad (2.1-13)$$

An infinite number of singularity functions can be obtained by successively differentiating or integrating the unit-step function. However, the derivative of the unit-step function does not exist at the point of discontinuity, $t=0$, and appropriate approximations involving limits must be utilized.

The **unit-impulse** function is defined as the first derivative of the unit-step function and is denoted by

$$\delta(t) = \begin{cases} \infty & \text{for } t=0 \\ 0 & \text{for } t \neq 0 \end{cases} \quad (2.1-14)$$

The ideal unit-impulse function represents a signal of infinite height or magnitude, zero width or duration, and unity area underneath the implied curve. A plot of the unit-impulse function is shown in Fig. 2.1-5a. The derivative of the unit impulse is the **unit doublet**

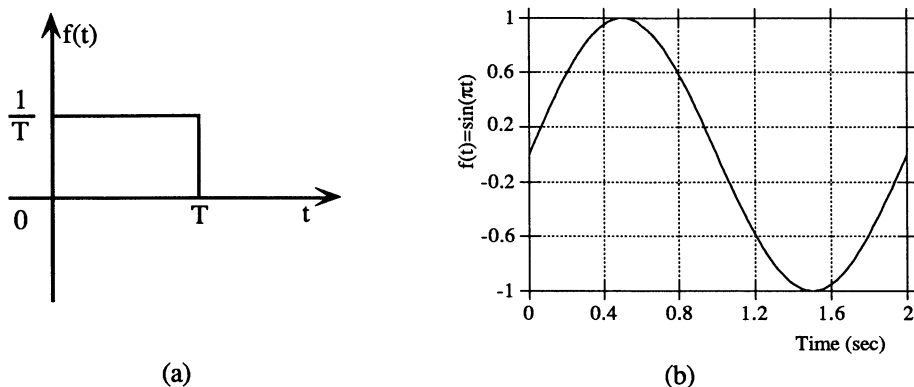


Fig. 2.1-4 The unit-area rectangular pulse (a) and one cycle of a sine waveform (b).

shown in Fig. 2.1-5b; the derivative of the unit doublet is the **unit triplet**, and so forth. In addition, an impulse of area 2 is denoted by $2\delta(t)$, while a unit impulse delayed T sec is denoted by $\delta(t-T)$, where

$$\delta(t-T) = \begin{cases} \infty & \text{for } t=T \\ 0 & \text{for } t \neq T \end{cases} \quad (2.1-15)$$

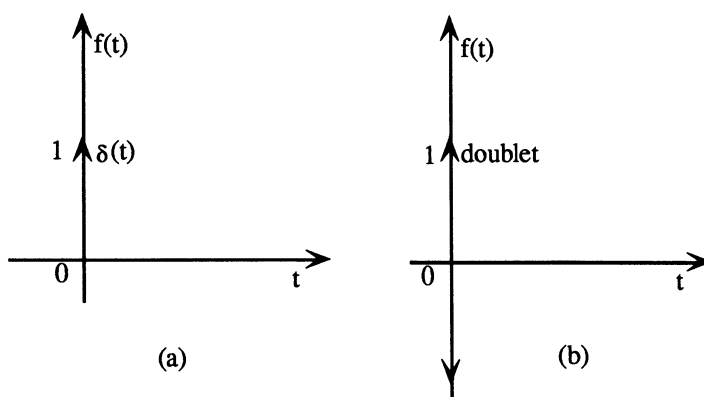


Fig. 2.1-5 The unit-impulse function (a) and the unit-doublet function (b).

The impulse function is attractive mainly because it is simpler to determine the impulse response of a system than its corresponding pulse or step responses. Even though it is not possible to generate an ideal impulse signal in the laboratory for testing purposes, a practical solution is to approximate the ideal impulse signal with a rectangular pulse of width which is relatively "narrow" when compared to the smallest time constant of the system. This approximation is consistent with the interpretation that the unit-impulse function is the limit of a unit-area rectangular pulse as its width approaches zero.

If the unit-step function is integrated, rather than differentiated, the singularity function that results is known as the **unit-ramp** function, denoted by

$$f(t) = \begin{cases} 1 & \text{for } t \geq 0 \\ 0 & \text{for } t < 0 \end{cases} \quad (2.1-16)$$

and shown in Fig.2.1-6a . The above description can be expressed analytically as

$$f(t) = tu(t) \quad (2.1-17)$$

where $u(t)$ is the unit-step function. A unit-ramp function delayed by T sec is written as

$$f(t) = (t-T)u(t-T) \quad (2.1-18)$$

and is shown in Fig. 2.1-6b. A nonunit-ramp function, or a ramp function of slope $k_1 \neq 1$, is expressed as

$$f(t) = k_1 tu(t) \quad (2.1-19)$$

for any constant k_1 , positive or negative. Integrating the ramp function results in a unit-parabola function, and so on.

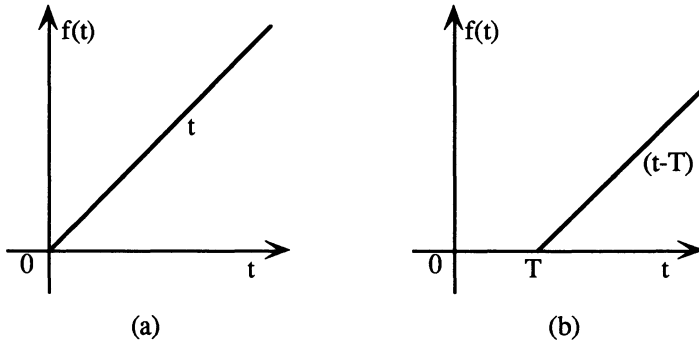


Fig. 2.1-6 The unit-ramp function (a) and the delayed unit-ramp function (b).

2.1-2 Systems Analysis and Design

The analysis and design methods presented in this book are directed toward linear systems. Even though in reality linear systems do not exist, many physical systems can be approximated and represented over a certain range of operation by linear models and can be studied using linear mathematics.

The main objective of **system analysis** is the development of a suitable mathematical model. This involves the application of relevant laws and principles of science, engineering, etc., to obtain appropriate mathematical relationships that represent the system. The resulting equations are then solved to determine and evaluate system response characteristics due to common types of inputs. Obtaining system characteristics that meet certain performance requirements is the basic goal of **system design**.

There are several types of mathematical models and solution methods available for determining the characteristics of linear systems. A SISO continuous system, for instance, can be modeled in the time domain by a differential equation or by a set of differential equations and in the frequency domain by a transfer function. Appropriate techniques and tools such as the digital computer are then employed to carry out solutions obtained from the mathematical models.

2.2 DIFFERENTIAL EQUATIONS

The differential equation model of a system with a single input $x(t)$ and a single output $y(t)$ may be written in the form

$$\begin{aligned} y^{(n)}(t) + b_1 y^{(n-1)}(t) + \dots + b_{n-1} y'(t) + b_n y(t) = \\ = a_0 x^{(m)}(t) + a_1 x^{(m-1)}(t) + \dots + a_{m-1} x'(t) + a_m x(t) \end{aligned} \quad (2.2-1)$$

where m and n are positive integers, and $y^{(i)}$ denotes the i th time derivative.

Differential equations and the systems they represent are classified in various ways. In Eq. (2.2-1), the variable $y(t)$ represents the **output** or **response** of the system; $y(t)$ is also known as the **solution** or **dependent variable** of the differential equation. The function $x(t)$ represents the system **input** or **excitation**; $x(t)$ is also called the **driving** or **forcing function**.

If the forcing function is zero, the differential equation is called **homogeneous**, otherwise it is called **nonhomogeneous**. The variable ' t ' is called the **independent variable** and represents time.

The **order** of a differential equation is represented by n , which denotes the order of the highest derivative of $y(t)$. The power to which the highest derivative is raised is referred to as the **degree** of the differential equation.

The parameters $a_0, a_1, \dots, a_m$ and $b_1, b_2, \dots, b_n$ represent the **coefficients** of the differential equation. If the coefficients are constant, the equation is by definition **time-invariant**; if the coefficients are functions of time, on the other hand, the equation is **time-variant**.

If the derivatives are ordinary, that is, if there is only one independent variable, the differential equation is called **ordinary**; if the derivatives are partial, however, the equation is a **partial** differential equation.

A differential equation may be further classified as **linear** if the dependent variable $y(t)$ and all of its derivatives are of first degree; otherwise, the equation is **nonlinear**.

Example 2.2-1 Differential Equations

Consider the differential equation

$$y''(t) + 5y'(t) + 10y(t) = 0$$

This is a first-degree, second-order, ordinary, homogeneous, linear, constant-coefficient (time-invariant) differential equation. On the other hand, the equation

$$t^2 y''(t) + 5y'(t) + t^3 y(t) = \sin 2t$$

is a nonhomogeneous, time-variant differential equation, because its coefficients are functions of the independent variable t . The equation is also linear, because $y(t)$ and all of its derivatives are of first degree.

The following two differential equations are nonlinear, and, in addition, the second equation is of second degree:

$$y^2 y'(t) + 5y(t) = 10$$

and

$$[y''(t)]^2 + 5y'(t) + 10y(t) = 20t$$

The differential equation model of a physical system may be given or it may have to be derived from another representation of the system by applying relevant natural laws. In the following sections, some common electrical, mechanical, and control systems are used as examples to help illustrate how physical systems can be modeled using differential equations. Similar mathematical models can be developed for other types of systems.

2.2-1 Electrical Networks

Consider the passive RC network shown in Fig. 2.2-1a. The input of the system is the applied voltage $v_1(t)$ and the output is the voltage across the capacitor $v_2(t)$. Using fundamental relationships, we obtain the equation

$$v_1(t) = Ri(t) + v_2(t) = RCv_2'(t) + v_2(t) \quad (2.2-1)$$

which after rearranging becomes

$$v_2'(t) + \frac{1}{RC} v_2(t) = \frac{1}{RC} v_1(t) \quad (2.2-2)$$

Eq. (2.2-2) is the desired differential equation model of the RC network.

A similar differential equation can be used to describe the behavior of the active RC network shown in Fig. 2.2-1b. A practical assumption is that the input current and voltage of the operational amplifier can be neglected. Applying Ohm's law to the first stage of the network gives

$$i_1(t) = \frac{1}{R} v_1(t), \quad i_R(t) = \frac{1}{R} v_o(t), \quad \text{and} \quad i_c(t) = C v_o'(t) \quad (2.2-3)$$

Applying Kirchhoff's current law at the summing junction (or node), we obtain

$$i_1(t) + i_c(t) + i_R(t) = 0 \quad \text{or} \quad i_c(t) + i_R(t) = -i_1(t) \quad (2.2-4)$$

Substituting the current relationships in the last equation and rearranging gives the differential equation

$$v_o'(t) + \frac{1}{RC} v_o(t) = -\frac{1}{RC} v_1(t) \quad (2.2-5)$$

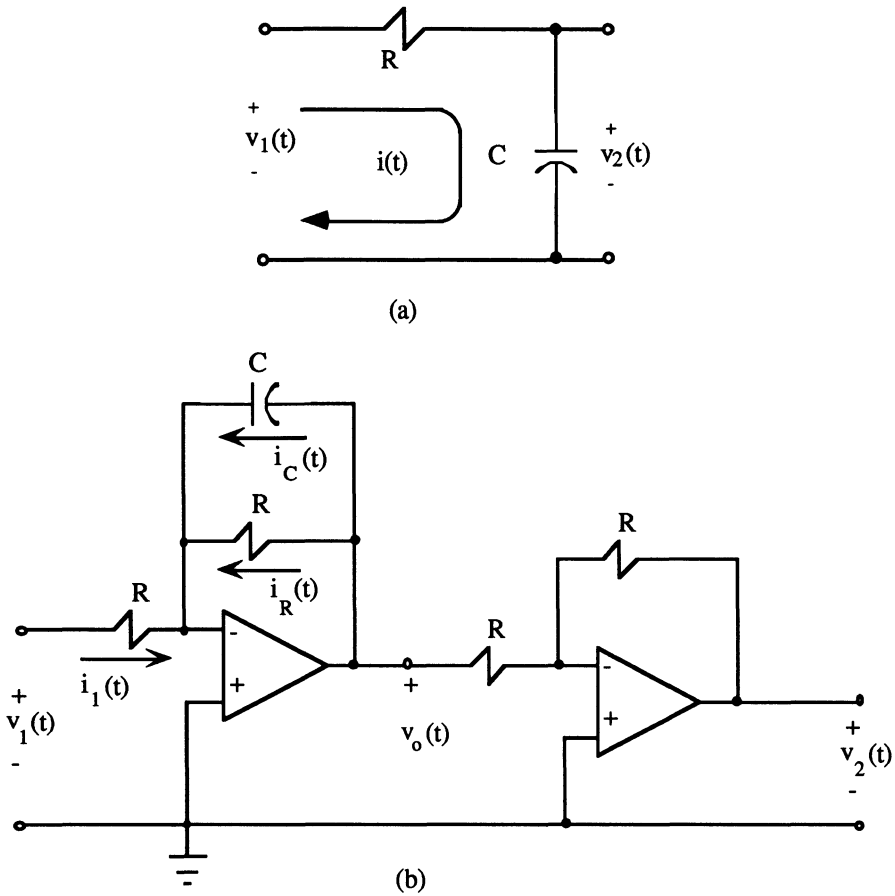


Fig. 2.2-1 Passive (a) and active (b) first-order network systems.

Eq. (2.2-5) resembles Eq. (2.2-2), except for the negative sign. The effect of the second stage in Fig. 2.2-1b is to invert the sign. That is, since $v_2(t) = -v_0(t)$, the previous differential equation becomes

$$v_2'(t) + \frac{1}{RC} v_2(t) = \frac{1}{RC} v_1(t) \quad (2.2-6)$$

which is identical to Eq. (2.2-2).

Next, consider the passive RLC network shown in Fig. 2.2-2a. This system has two energy storage elements, the capacitor and the inductor, and its differential equation model is of second order. To derive this equation, observe that from Kirchhoff's voltage law

$$v_1(t) = Ri(t) + Li'(t) + v_2(t) \quad (2.2-7)$$

Then, since $i(t) = Cv_2'(t)$, the differential equation becomes

$$v_1(t) = RCv_2'(t) + LCv_2''(t) + v_2(t) \quad (2.2-8)$$

or, in normalized form,

$$v_2''(t) + \frac{R}{L} v_2'(t) + \frac{1}{LC} v_2(t) = \frac{1}{LC} v_1(t) \quad (2.2-9)$$

It can be shown that the differential equation of the active network shown in Fig. 2.2-2b is also of second order and is given by

$$v_2''(t) + \left(\frac{1}{R_1 C_1} + \frac{1}{R_2 C_1} + \frac{1}{R_3 C_1} \right) v_2'(t) + \frac{1}{R_2 R_3 C_1 C_2} v_2(t) = \frac{1}{R_1 R_2 C_1 C_2} v_1(t) \quad (2.2-10)$$

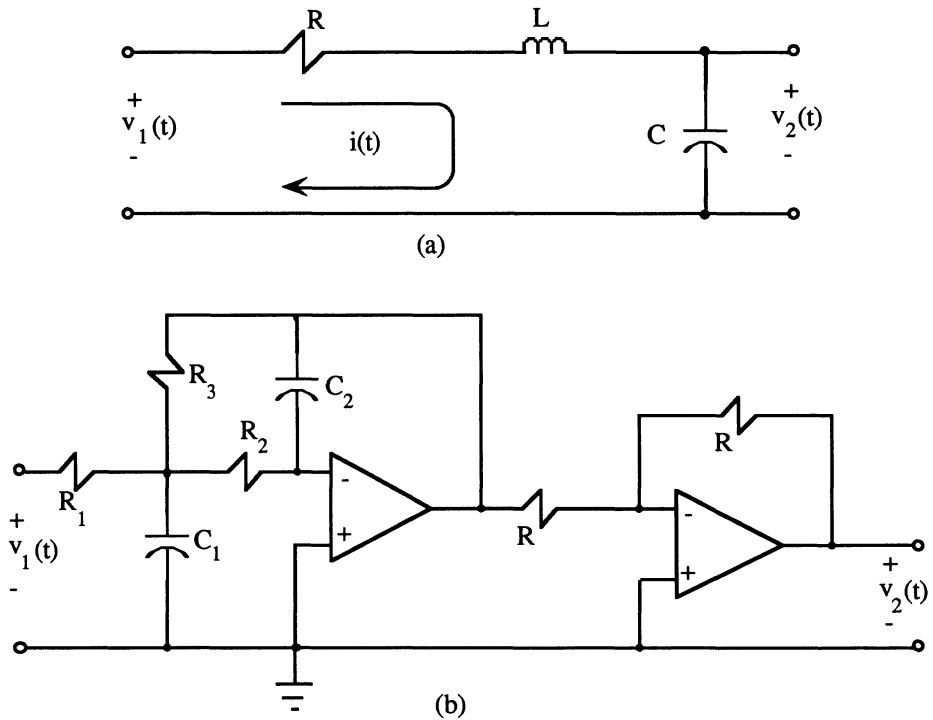


Fig. 2.2-2 Passive (a) and active (b) second-order network systems.

2.2-2 Mechanical Systems

Consider the spring-mass translational system shown in Fig. 2.2-3a. The mass M is resting on a surface, and is also connected to the wall through a dashpot and a spring. The spring constant K defines the stiffness of the elastic elements of the system, and the spring force F_s is proportional to the position or displacement $y(t)$. The dashpot represents viscous friction, which assumes that the damping force F_b is proportional to the velocity $y'(t)$, and B denotes the damping coefficient.

Assume that the system output (displacement) $y(t)$ is zero with the system at equilibrium, and that $y(t)$ is positive to the right and negative to the left. The system input is represented by the applied force $F_i(t)$ which will act to move the mass to the right (positive F_i) and will be opposed by the spring and frictional forces (negative F_s and F_b). The sign convention applies not only to displacement and force but to velocity and acceleration as well. Newton's second law is

$$\sum F = Ma \quad (2.2-11)$$

where 'a' represents the acceleration of the mass M . Since the sum of the forces is equal to Ma , we obtain

$$F_i + F_s + F_b = Ma = My''(t) \quad (2.2-12)$$

Also, since

$$F_s = -Ky(t), \text{ and } F_b = -By'(t) \quad (2.2-13)$$

the differential equation model in normalized form is

$$y''(t) + \frac{B}{M}y'(t) + \frac{K}{M}y(t) = \frac{1}{M}F_i(t) \quad (2.2-14)$$

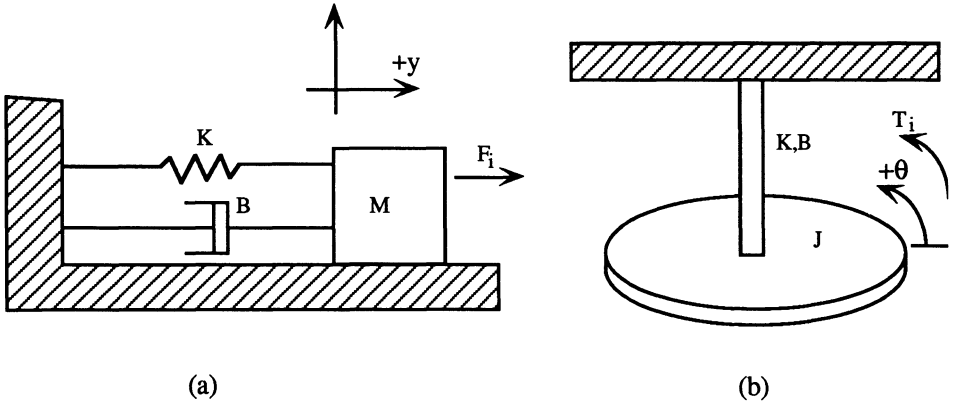


Fig. 2.2-3 Damped spring-mass (a) translational and (b) rotational systems.

The principles presented above can be extended to model the rotational system shown in Fig. 2.2-3b. The fundamental law for such a system is

$$\sum T = J\alpha \quad (2.2-15)$$

where T represents torque, J (mass) moment of inertia, and α angular acceleration. If θ is the angular displacement,

$$T_s + T_b + T_i = J\theta''(t) \quad (2.2-16)$$

Then, since

$$T_s = -K\theta, \text{ and } T_b = -B\theta'(t) \quad (2.2-17)$$

the differential equation model for the rotational system is

$$\theta''(t) + \frac{B}{J}\theta'(t) + \frac{K}{J}\theta(t) = \frac{1}{J}T_i(t) \quad (2.2-18)$$

2.2-3 Control Systems

Control systems can be represented by differential equations usually derived from some pictorial model of the system such as a block diagram or a signal flow graph. A simple block diagram model of a control system is shown in Fig. 2.2-4a. The system input is represented by $r(t)$, where 'r' stands for **reference**, and the system output by $c(t)$, where 'c' stands for **controlled variable**. The process or plant to be controlled acts physically as an integrator and K is an arbitrary constant. From the block diagram, the error signal is given by

$$e(t) = r(t) - c(t) \quad (2.2-19)$$

where

$$c(t) = K \int_0^t e(t) dt \quad (2.2-20)$$

By differentiating the last equation and solving for $e(t)$ we obtain

$$e(t) = \frac{1}{K} c'(t) \quad (2.2-21)$$

Substituting Eq. (2.2-21) in Eq. (2.2-19) results in the differential equation model

$$c'(t) + Kc(t) = Kr(t) \quad (2.2-22)$$

The block diagram representation of another control system is shown in Fig. 2.2-4b. To derive a differential equation description of this system, assume that the integrator blocks have been removed; then, the differential equations which describe the input signals of the two integrators are

$$c'(t) = y(t) \quad (2.2-23)$$

and

$$y'(t) = -\omega_n^2 c(t) - 2\zeta\omega_n y(t) + \omega_n^2 r(t) \quad (2.2-24)$$

where the symbols ζ and ω_n denote general constants.

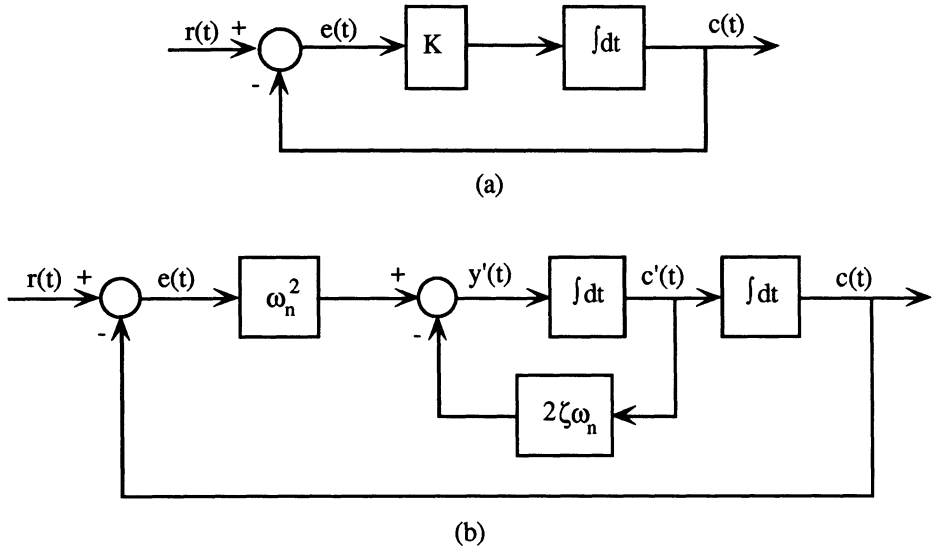


Fig. 2.2-4 Block diagram model examples of control systems.

Eq. (2.2-23) and Eq. (2.2-24) represent a system of simultaneous differential equations. By differentiating the first equation and setting its derivative $y'(t)$ equal to the right side of the second equation, we obtain

$$c''(t) = y'(t) = -\omega_n^2 c(t) - 2\zeta\omega_n c'(t) + \omega_n^2 r(t) \quad (2.2-25)$$

Rearranging Eq. (2.2-25) results in the second-order differential equation model

$$c''(t) + 2\zeta\omega_n c'(t) + \omega_n^2 c(t) = \omega_n^2 r(t) \quad (2.2-26)$$

The remarkable characteristic/property of the differential equation models for the systems considered so far is that, except for input/output notation, they are mathematically similar. This implies that the different systems they describe behave in an analogous fashion. For instance, a second-order spring-mass mechanical system may be represented by the same differential equation as an RLC electrical network. For this reason, knowledge and experience gained from the analysis of a familiar type of system can be extended to other types of systems, if these systems can be modeled by similar differential equations.

2.3 TIME RESPONSE USING CLASSICAL METHODS

The performance of physical systems can be analyzed and evaluated from their time responses due to known inputs. The objective of this section is to obtain and interpret the basic time response characteristics using classical methods.

Consider the n th-order, linear, time-invariant differential equation given by Eq. (2.2-1), that is

$$\begin{aligned} y^{(n)}(t) + b_1 y^{(n-1)}(t) + \dots + b_{n-1} y'(t) + b_n y(t) = \\ = a_0 x^{(m)}(t) + a_1 x^{(m-1)}(t) + \dots + a_{m-1} x'(t) + a_m x(t) \end{aligned} \quad (2.3-1)$$

where again $x(t)$ and $y(t)$ represent the input and output variables, respectively, and n and m are positive integers. To simplify the notation, the derivative operator is replaced by the symbol 's'. That is,

$$s = \frac{d}{dt}, \quad s^2 = \frac{d^2}{dt^2}, \text{ etc.} \quad (2.3-2)$$

The differential equation then becomes

$$(s^n + b_1 s^{n-1} + \dots + b_{n-1} s + b_n) y(t) = (s^m + b_1 s^{m-1} + \dots + b_{m-1} s + b_m) x(t) \quad (2.3-3)$$

To solve the above differential equation implies to find a function $y(t)$ which contains no derivatives and satisfies the equation. Equivalently, when the $y(t)$ is substituted in the differential equation, it must make the left-hand side of the equation equal to the right-hand side.

One of the standard classical methods for solving differential equations is known as the method of **undetermined coefficients**. The method is applied first to solve homogeneous differential equations; then, it is generalized for the nonhomogeneous case.

2.3-1 Solution of Homogeneous Differential Equations

For the homogeneous case, the n th-order differential equation becomes

$$(s^n + b_1 s^{n-1} + \dots + b_{n-1} s + b_n) y(t) = 0 \quad (2.3-4)$$

$y(t)$ is required to be nontrivial (i.e., $y(t)$ cannot be zero), suggesting that the term in parentheses must be zero, that is

$$s^n + b_1 s^{n-1} + \dots + b_{n-1} s + b_n = 0 \quad (2.3-5)$$

Eq. (2.3-5) is called the **characteristic equation** of the differential equation and thus of the system the differential equation represents; the characteristic equation is also called **auxiliary** or **index**. The n **roots** (or **zeros**) of the characteristic equation are known as **characteristic roots** or **eigenvalues**. The characteristic roots may be real and distinct, real and repeated, or complex. Furthermore, complex roots can be either simple or repeated but always occur in complex-conjugate pairs as long as the coefficients in Eq. (2.3-5) are real.

The solution obtained from the homogeneous part of a differential equation is called **complementary**; this solution is also referred to as **homogeneous, source-**

free, natural and transient. The complementary solution becomes the **total or general solution** in the event that the only system excitation comes from initial conditions, that is, from energy stored inside the system, rather than from an external input source.

The term "complementary" implies that the left-hand side solution of Eq. (2.3-1) complements the solution obtained from the right-hand side in order to form the total solution. The terms "homogeneous" and "source-free" imply that the system input, $x(t)$, is zero. The term "natural" refers to the character or composition of the system, rather than the dependence of this solution on external input signals. Finally, the term "transient" refers to the temporary nature of the response, which tends to vanish as time increases.

If the characteristic equation has a root with a positive real part, the complementary solution contains a term which increases exponentially with time. Roots with negative real parts, on the other hand, lead to exponential terms that decay with time and eventually die out or become zero. If the characteristic equation has a pair of purely imaginary roots, the complementary solution contains sinusoidal terms of constant amplitude. Repeated imaginary roots, however, lead to a response that increases with time. In general, systems studied in this book exhibit transient responses that vanish as t approaches infinity.

The solution of first-order, second-order, and higher-order differential equations is considered next.

First-Order Equations. Consider the first-order homogeneous equation

$$y'(t) + ay(t) = 0 \quad (2.3-6)$$

where the coefficient 'a' is a constant. Using the s operator notation of Eq. (2.3-2), the above equation can be written as

$$(s + a)y(t) = 0$$

The characteristic equation is

$$s + a = 0 \quad (2.3-7)$$

The characteristic root (eigenvalue) is

$$s_1 = -a \quad (2.3-8)$$

which is a real number. The solution of all such first-order, homogeneous differential equations is given by

$$y_h(t) = k_1 e^{s_1 t} = k_1 e^{-at} \quad (2.3-9)$$

where k_1 is an arbitrary constant. The value of k_1 depends on the initial state of the system, commonly known as the **initial condition**, denoted by $y(0)$ or y_0 . Thus, the solution can be written as

$$y_h(t) = y(0)e^{-at} \quad (2.3-10)$$

To verify that Eq. (2.3-10) is the actual solution of the differential equation, we must show that $y_h(t)$ satisfies Eq. (2.3-6). From Eq. (2.3-10),

$$y_h'(t) = -ay(0)e^{-at}$$

Substituting $y_h'(t)$ and $y_h(t)$ in Eq. (2.3-6) gives

$$-ay(0)e^{-at} + ay(0)e^{-at} = 0$$

which verifies that indeed Eq. (2.3-10) is the solution.

The solution of Eq. (2.3-6) is shown plotted in Fig. 2.3-1 for positive values of a . It can be shown that the decreasing or decaying exponential function has dropped to 36.8 percent of its initial value of $y(0)$ or k_1 in $1/a$ seconds. The inverse of the constant a is known as the **time constant** of the first-order system, and is usually denoted by τ . That is,

$$\text{Time constant} = \tau = \frac{1}{a} \text{ sec} \quad (2.3-11)$$

The exponential function in Eq. (2.3-10) will be within 2 percent of its final value in four time constants or 4τ . The time required for a response to settle within a certain percentage of its final value is referred to as the **settling time**, denoted by T_s . Thus, for a first-order system,

$$\text{Settling Time} = T_s \cong 4\tau = \frac{4}{a} \text{ sec} \quad (2.3-12)$$

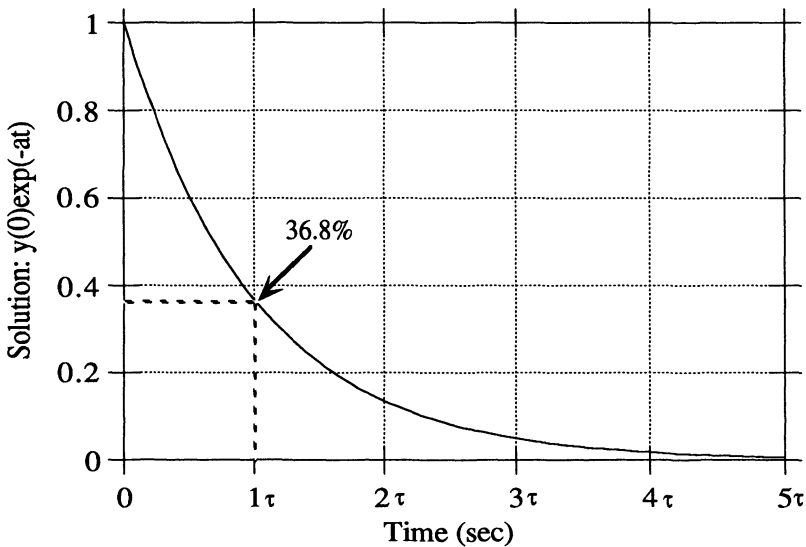


Fig. 2.3-1 Solution of a first-order, homogeneous differential equation.

Example 2.3-1.

Consider the first-order differential equation

$$y'(t) + 5y(t) = 0$$

given the initial condition $y(0) = 1$. It is desired to find $y(t)$.

The characteristic equation is

$$s + 5 = 0$$

and the characteristic root is

$$s_1 = -5, \text{ a real number.}$$

The solution then becomes

$$y(t) = k_1 e^{-5t}, \text{ for all } t \geq 0$$

To determine k_1 , apply the initial condition $y(0)=1$. This yields

$$y(0) = 1 = k_1$$

and the complete solution becomes

$$y(t) = e^{-5t}u(t)$$

Note that the time constant is $\tau=0.2$ sec and the settling time is $T_s=0.8$ sec. For all practical purposes, this means that the transient response of the system will vanish in about 0.8 sec.

Second-Order Equations. Consider the second-order differential equation

$$y''(t) + 2\zeta\omega_n y'(t) + \omega_n^2 y(t) = 0 \quad (2.3-13)$$

The coefficients of Eq. (2.3-13) are expressed in terms of the popular system parameters ζ and ω_n , where ζ is the **damping ratio** and ω_n the **(undamped) natural frequency**. The units of ω_n are rad/sec and ζ is unitless. The characteristic equation is

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \quad (2.3-14)$$

The characteristic roots or eigenvalues can be expressed in the form

$$s_1 = -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2}, \text{ and } s_2 = -\zeta\omega_n - j\omega_n\sqrt{1-\zeta^2} \quad (2.3-15)$$

The imaginary component of the complex roots is called the **damped natural frequency**. That is

$$\text{Damped natural frequency} = \omega_d = \omega_n\sqrt{1-\zeta^2} \quad (2.3-16)$$

The roots of Eq. (2.3-14) can be real and distinct (simple), real and repeated (equal), or complex (simple or repeated), depending on the value of the damping ratio ζ . That is, if $\zeta > 1$ the roots are real and distinct, if $\zeta = 1$ the roots are real and repeated, and if $\zeta < 1$ the roots occur in complex-conjugate pairs. In general, systems are classified according to the value of their damping ratio as shown in Table 2.3-1.

Overdamped, critically-damped, and underdamped systems are discussed in the following paragraphs.

Case 1: $\zeta > 1$. For an **overdamped** system, the characteristic roots are real and distinct (or unequal) and they are given by

$$s_1 = -\zeta\omega_n + \omega_n\sqrt{\zeta^2-1}, \text{ and } s_2 = -\zeta\omega_n - \omega_n\sqrt{\zeta^2-1} \quad (2.3-17)$$

Table 2.3-1 System classification according to the damping ratio ζ

ζ	Roots	System Type
$\zeta > 1$	Real and Distinct	Overdamped
$\zeta = 1$	Real and Repeated	Critically damped
$0 < \zeta < 1$	Complex (Negative Real Part)	Underdamped
$\zeta = 0$	Pure Imaginary	Undamped
$-1 < \zeta < 0$	Complex (Positive Real Part)	Negatively damped
$\zeta < -1$	Real (Positive)	Negatively damped

The solution of Eq. (2.3-13) then becomes

$$y(t) = k_1 e^{s_1 t} + k_2 e^{s_2 t} \quad (2.3-18)$$

where k_1 and k_2 are arbitrary constants determined from the initial conditions, usually $y(0)$ and $y'(0)$.

There are two time constants, $1/s_1$ and $1/s_2$, associated with the response of this system and the **settling time** is defined as four times the larger time constant.

Example 2.3-2 Real and Distinct Roots

Solve the following differential equation assuming the initial conditions $y(0) = 0$ and $y'(0) = 1$.

$$y''(t) + 3y'(t) + 2y(t) = 0$$

The characteristic equation is

$$s^2 + 3s + 2 = 0$$

and the roots are $s_1 = -1$ and $s_2 = -2$. For these real and distinct roots, the solution given by Eq. (2.3-18) is

$$y(t) = k_1 e^{-t} + k_2 e^{-2t}$$

Applying the initial conditions, results in the system of simultaneous equations

$$y(0) = 0 = k_1 + k_2, \text{ and } y'(0) = 1 = -k_1 - 2k_2$$

Solving for k_1 and k_2 , we get $k_1 = 1$ and $k_2 = -1$. Thus, the complete solution becomes

$$y(t) = (e^{-t} - e^{-2t})u(t)$$

Case 2: $\zeta=1$. In a **critically-damped** system, the real and equal characteristic roots are given by

$$s = s_1 = s_2 = -\omega_n \quad (2.3-19)$$

The solution of the differential equation then is

$$y(t) = k_1 e^{-\omega_n t} + k_2 t e^{-\omega_n t} = (k_1 + k_2 t) e^{-\omega_n t} \quad (2.3-20)$$

where again the constants k_1 and k_2 are determined from the initial conditions. The above solution can be checked by substituting it into the original equation to see if it satisfies the equation. For the critically-damped case, the system time constant is $1/\omega_n$ and the settling time $4/\omega_n$.

Example 2.3-3 Real and Repeated Roots

Solve the differential equation

$$y''(t) + 4y'(t) + 4y(t) = 0$$

The characteristic equation is

$$s^2 + 4s + 4 = 0$$

from which the real and repeated roots are

$$s_1 = s_2 = -2$$

The solution then becomes

$$y(t) = k_1 e^{-2t} + k_2 t e^{-2t}$$

where the constants k_1 and k_2 can be determined from initial conditions as in the previous example. The time constant for this system is 0.5 sec and the settling time is 2 sec.

Case 3: $0 < \zeta < 1$. In an **underdamped** system, the characteristic roots or eigenvalues are complex conjugates as given by Eq. (2.3-17). For this case, the solution of the differential equation is

$$y(t) = e^{-\zeta\omega_n t} [k_1 \cos\omega_n \sqrt{1-\zeta^2} t + k_2 \sin\omega_n \sqrt{1-\zeta^2} t] \quad (2.3-21)$$

To verify Eq.(2.3-21), assume the simpler notation for the pair of complex-conjugate roots, that is , suppose

$$s_1 = -a + jb, \text{ and } s_2 = -a - jb \quad (2.3-22)$$

The solution then becomes

$$y(t) = k_1 e^{(-a+jb)t} + k_2 e^{(-a-jb)t} = e^{-at} (k_1 e^{jbt} + k_2 e^{-jbt}) \quad (2.3-23)$$

Applying Euler's identity

$$e^{\pm jx} = \cos x \pm j \sin x \quad (2.3-24)$$

to Eq. (2.3-23), results in the solution

$$\begin{aligned}y(t) &= e^{-at}[k_1(\cos bt + j\sin bt) + k_2(\cos bt - j\sin bt)] \\&= e^{-at}[(k_1 + k_2)\cos bt + (jk_1 - jk_2)\sin bt]\end{aligned}\quad (2.3-25)$$

which, except for notation, is equivalent to Eq. (2.3-21).

In the second-order, underdamped case, the time constant of the system is given by

$$\text{Time constant} = \tau = \frac{1}{\zeta\omega_n} \quad (2.3-26)$$

Thus, the settling time is

$$\text{Settling time} = T_s = \frac{4}{\zeta\omega_n} \quad (2.3-27)$$

Example 2.3-4 Complex Roots

Consider the differential equation

$$y''(t) + 2y'(t) + 5y(t) = 0$$

The characteristic equation is

$$s^2 + 2s + 5 = 0$$

from which the complex roots are

$$s_1 = -1 + j2, \quad \text{and} \quad s_2 = -1 - j2$$

The solution of the differential equation becomes

$$y(t) = k_1 e^{-t} \cos 2t + k_2 e^{-t} \sin 2t$$

where k_1 and k_2 depend on the system initial conditions. Observe that for this system the time constant is 1 sec and the settling time 4 sec. The natural frequency ω_n is 2.236 rad/sec, the damping ratio ζ is 0.447 and the frequency of oscillations is $\omega_d = 2$ rad/sec. Notice that the inverse of ω_d is the period of oscillations, a very helpful quantity when plotting system responses.

Higher-Order Equations. For higher than second-order differential equations, the characteristic roots are found by solving the characteristic equation. These roots are grouped as real and distinct, real and repeated and complex. The corresponding solutions are then developed and combined to determine the solution of the homogeneous equation.

Example 2.3-5

Solve the equation

$$y'''(t) + 6y''(t) + 11y'(t) + 6y(t) = 0$$

The characteristic equation is

$$s^3 + 6s^2 + 11s + 6 = 0$$

and the real and distinct eigenvalues are

$$s_1 = -1, s_2 = -2, \text{ and } s_3 = -3$$

The solution then is

$$y(t) = k_1 e^{-t} + k_2 e^{-2t} + k_3 e^{-3t}$$

where k_1 , k_2 , and k_3 can be evaluated from three initial conditions.

Example 2.3-6

Determine the homogeneous solution of the differential equation whose characteristic equation is given by

$$s^3 + 6s^2 + 12s + 8 = 0$$

The real and repeated characteristic roots are

$$s_1 = s_2 = s_3 = -2$$

The solution then is

$$y(t) = k_1 e^{-2t} + k_2 t e^{-2t} + k_3 t^2 e^{-2t} = e^{-2t}(k_1 + k_2 t + k_3 t^2)$$

where the constants k_1 , k_2 , and k_3 are evaluated from initial conditions.

Example 2.3-7

Solve the differential equation whose eigenvalues are

$$s_1 = -3, s_2 = -1 + j2 \text{ and } s_3 = -1 - j2$$

The solution due to the combination of the real and complex roots is

$$y(t) = k_1 e^{-3t} + k_2 e^{-t} \cos 2t + k_3 e^{-t} \sin 2t$$

where again the constants k_1 , k_2 , and k_3 depend on initial conditions.

2.3-2 Solution of Nonhomogeneous Differential Equations

For a nonhomogeneous differential equation, the input or forcing function $x(t)$ is not zero. The total solution, or just the solution, of the differential equation is composed of two components, the complementary component $y_h(t)$ already discussed earlier and the **particular** component $y_p(t)$ discussed next; the particular solution is also known as **forced** or **steady-state**. Thus, the solution of a nonhomogeneous differential equation is

$$y(t) = y_h(t) + y_p(t) \quad (2.3-28)$$

The term particular implies that $y_p(t)$ satisfies the nonhomogeneous equation for a particular input $x(t)$. The term forced is used because $y_p(t)$ depends on the forcing function $x(t)$, while the term steady-state refers to the permanent nature of state $y_p(t)$ reaches with increasing time.

The complementary solution $y_h(t)$ is found by replacing $x(t)$ by zero in the nonhomogeneous equation, and, then, solving the resulting homogeneous equation. Note, however, that since the initial conditions apply to the total solution, the arbitrary constants $k_1, k_2, \dots, k_n$ associated with $y_h(t)$ must not be determined until the solution ($y=y_h+y_p$) is obtained.

In determining the complementary solution, we first factored the characteristic equation and used the resulting roots to obtain $y_h(t)$. The procedure for obtaining the particular solution $y_p(t)$ works in reverse. That is, from the terms on the right-hand side of the differential equation, we determine the implied roots from which a generalized particular solution $y_p(t)$ is constructed. Then, $y_p(t)$ is substituted into the original differential equation to determine the unknown constants.

The solution of nonhomogeneous differential equations is first illustrated using first-order and second-order systems with a unit-step function input. Next, the method is extended to higher-order systems with various types of inputs. The analysis of first-order and second-order systems is emphasized because such systems are the building blocks for higher-order systems. Furthermore, higher-order systems are often approximated by first- and/or second-order models.

Step Response of First-Order Systems. Assume that a first-order system is modeled by the differential equation

$$y'(t) + ay(t) = Kx(t) \quad (2.3-29)$$

where $1/a$ is the system time constant and K is an arbitrary constant. If the system is initially at rest, that is, if the initial conditions are zero, determine the response of the system to the unit-step function

$$x(t) = u(t) \quad (2.3-30)$$

The equation to be solved is

$$y'(t) + ay(t) = K, \text{ for } t \geq 0 \quad (2.3-31)$$

together with the initial condition $y(0) = 0$.

First, the complementary solution $y_h(t)$ is obtained using the procedure developed in the previous section. That is, from the characteristic equation

$$s + a = 0 \quad (2.3-32)$$

the real characteristic root is

$$s_1 = -a \quad (2.3-33)$$

The complementary solution becomes

$$y_h = k_1 e^{-at} u(t) \quad (2.3-34)$$

Next, to find the particular solution $y_p(t)$, we proceed as follows. Since the only term on the right-hand side of the equation is the constant K , the root contributed by this constant must be equal to zero. Call this root s'_1 , where the prime notation is used to distinguish the right-hand side from the left-hand side roots. Thus,

$$s'_1 = 0 \quad (2.3-35)$$

suggests the generalized particular solution

$$y_p(t) = A \quad (2.3-36)$$

where A is an arbitrary constant. Since $y_p(t)$ is a solution to Eq. (2.3-29), it should satisfy it. This gives

$$0 + aA = K, \text{ or } y_p(t) = A = K/a \quad (2.3-37)$$

The $y_p(t)$ of Eq. (2.3-37) applies for the general case if the right-hand side of the equation is constant. For such cases, the particular solution is the ratio of the right-hand side constant term K , to the right-most coefficient of the left-hand side, in this case a . Thus, the total solution becomes

$$y(t) = y_h(t) + y_p(t) = k_1 e^{-at} + K/a \quad (2.3-38)$$

The initial condition of zero is now applied to determine k_1 . That is,

$$y(0) = 0 = k_1 + K/a, \text{ or } k_1 = -K/a \quad (2.3-39)$$

The solution of Eq. (2.3-29) then becomes

$$y(t) = \frac{K}{a} (1 - e^{-at}) u(t) \quad (2.3-40)$$

The ratio K/a represents the steady-state or final value of the system as t approaches infinity. The unit-step response of the system is shown plotted in Fig. 2.3-2 for the unity-gain case, that is, for $K=a$.

The time constant of this system is $\tau=1/a$ sec and its settling time is $T_s=4/a$ sec, or four time constants. As expected, the response reaches 1.8 percent of its final or steady-state value (1 in this case) in about $4/a$ sec. If the slope of the step response at $t=0$ is maintained until it intercepts its final value, the interception occurs at $t=\tau=1/a$ sec, or in one time constant. At this time, the step response has reached 63.2 percent of its final value.

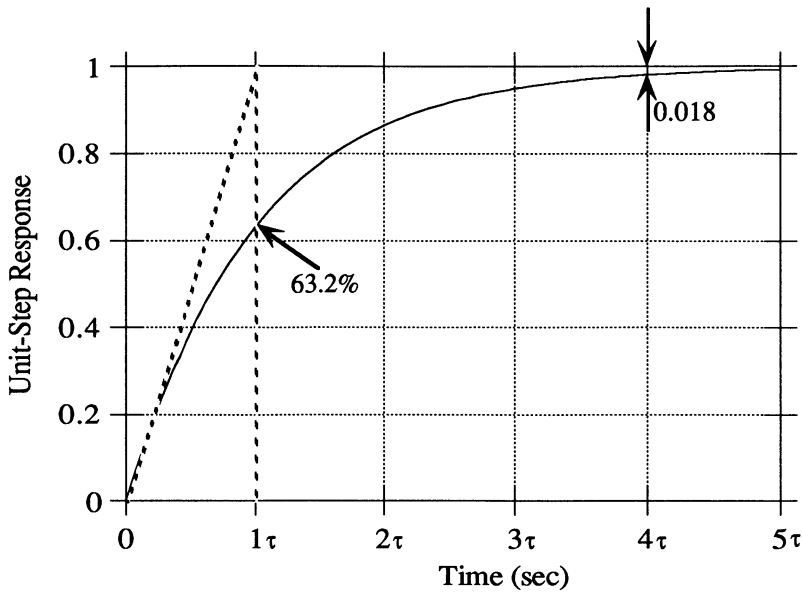


Fig. 2.3-2 Unit-step response of a first-order system.

For small values of a , the negative real root is near the origin, the time constant is large, and the system is "sluggish" or slow-responding. On the other hand, for large values of a , the time constant is small and the system responds quickly.

If a is positive, the exponential term in Eq. (2.3-40) or the transient component of the response decays with time and eventually dies out. If a is negative, however, the transient term increases with time. The following specific examples are presented to further illustrate and generalize these concepts.

Example 2.3-8

A first-order system is modeled by the differential equation

$$2y'(t) + 10y(t) = 20x(t)$$

Determine the unit-step response of the system, assuming zero initial conditions.

First, divide by 2 to normalize the equation. Since the input is a unit-step function, the equation can be written as

$$y'(t) + 5y(t) = 10, \text{ for } t \geq 0$$

The characteristic equation is

$$s + 5 = 0$$

The characteristic root is $s_1 = -5$ and the complementary solution becomes

$$y_h(t) = k_1 e^{-5t}$$

Next, we must find the particular solution. The right-hand side of the original equation consists of the constant 10, which implies that the root that contributed this constant is zero, or

$$s'_1 = 0$$

This root is real and distinct because it is different from the only other root, that is, $s_1 = -5$. Therefore, the particular solution is

$$y_p(t) = A$$

where A is an arbitrary constant. To find A , we substitute $y_p(t)$ and its derivative into the original equation. This gives

$$5A = 10, \text{ or } A = 2$$

The total solution becomes

$$y(t) = y_h(t) + y_p(t) = k_1 e^{-5t} + 2$$

To find k_1 , we must now apply the initial condition. Thus,

$$y(0) = 0 = k_1 + 2, \text{ or } k_1 = -2$$

and the complete solution is

$$y(t) = 2(1 - e^{-5t})u(t)$$

For this system, the relevant parameter values are

Time constant $\tau = \frac{1}{5} = 0.2$ sec, settling time $T_s = \frac{4}{5} = 0.8$ sec, and steady-state value = 2.

Example 2.3-9

Determine the unit-impulse response of the system used in Example 2.3-8. That is, solve the differential equation

$$2y'(t) + 10y(t) = 20\delta(t)$$

If the system is at rest, the response is zero for $t < 0$. Because the forcing function is zero except at $t = 0$, the solution for $t \geq 0$ consists only of the complementary component

$$y(t) = y_h(t) = k_1 e^{-5t}$$

where k_1 can be evaluated from the coefficients of the original equation by the following procedure.

The only way the equation can be satisfied is for the highest-order derivative term to contain an impulse at $t = 0$, which balances the impulse of the right side. Therefore, at $t = 0$ this implies

$$2[k_1\delta(t)] = 20\delta(t)$$

from which $k_1=10$ and the solution becomes

$$y(t) = 10e^{-5t}u(t)$$

Note that if the term $y(t)$ contains the impulse, then $y'(t)$ creates a doublet on the left-hand side. However, this is a contradiction because there is no doublet on the right-hand side of the equation.

Since, by definition, the impulse function can be derived by differentiating the step function, the above answer can also be obtained by differentiating the unit-step response expression of Example 2.3-8.

Example 2.3-10

Assuming the system at rest, solve the differential equation

$$y'(t) + y(t) = 2tu(t)$$

Similar to Example 2.3-9, the complementary component can be found to be

$$y_h(t) = k_1e^{-t}$$

The term $2t$ implies two repeated (equal) real roots of value zero. That is,

$$s'_1 = 0 \text{ and } s'_2 = 0$$

The generalized particular solution is

$$y_p(t) = A + Bt$$

Substituting $y_p(t)$ and its derivative into the original equation gives

$$B + A + Bt = 2t$$

Equating coefficients, results in the simultaneous equations

$$A + B = 0 \text{ and } B = 2$$

Thus, the total solution becomes

$$y(t) = k_1e^{-t} - 2 + 2t$$

Applying initial condition of zero gives

$$y(0) = 0 = k_1 - 2, \text{ or } k_1 = 2$$

The completely defined total solution is

$$y(t) = 2e^{-t} - 2 + 2t, \text{ for } t \geq 0, \text{ or } y(t) = (2e^{-t} - 2 + 2t)u(t)$$

Example 2.3-11

Solve the differential equation

$$y'(t) + y(t) = (1 + e^{-t})u(t)$$

assuming zero initial conditions. The complementary solution is

$$y_h(t) = k_1 e^{-t}$$

To find $y_p(t)$, consider the right-hand side, term by term, until all relevant roots are determined and thus a generalized $y_p(t)$ can be written. The first term is the constant 1 which implies the root

$$s'_1 = 0$$

and contributes the particular-solution component A, a constant. Next, consider the exponential term e^{-t} . The implied root is

$$s'_2 = -1$$

The root of -1, however, has already occurred as the eigenvalue of the system, $s_1 = -1$. Because of this, the term contributed to the particular solution by the repeated root -1 is

$$Bte^{-t}$$

where B is another arbitrary constant. The particular solution then is

$$y(t) = A + Bte^{-t}$$

Substitution of $y_p(t)$ into the entire equation yields the values of $A=1$ and $B=1$. The total solution then becomes

$$y(t) = k_1 e^{-t} + 1 + te^{-t}$$

Applying the initial conditions of zero results in the equality $k_1 = -1$. The completely defined total solution is

$$y(t) = (1 - e^{-t} + te^{-t})u(t)$$

Example 2.3-12

Assuming zero initial conditions, solve the equation

$$y'(t) + y(t) = 10e^{-t}\sin t$$

The complementary solution is the same as in the previous examples, that is,

$$y_h(t) = k_1 e^{-t}$$

The sinusoidal term implies the complex roots

$$s'_1 = -1 + j1 \quad \text{and} \quad s'_2 = -1 - j1$$

which in turn suggest the particular solution

$$y_p(t) = Ae^{-t}\cos t + Be^{-t}\sin t$$

Substituting $y_p(t)$ and $y'_p(t)$ in the original equation results in

$$e^{-t}(-A\cos t - A\sin t - B\sin t + B\cos t + A\cos t + B\sin t) = 10e^{-t}\sin t$$

Simplifying and then equating the coefficients of the $\sin t$ and $\cos t$ terms, leads to the constant values $A=-10$, and $B=0$. The total solution then is

$$y(t) = k_1e^{-t} - 10e^{-t}\cos t$$

After applying the initial conditions of zero, we find that $k_1=10$ and the desired solution function becomes

$$y(t) = (10e^{-t} - 10e^{-t}\cos t)u(t)$$

Step Response of Second-Order Systems. Consider the second-order equation in the generalized or standard form

$$y''(t) + 2\zeta\omega_n y'(t) + \omega_n^2 y(t) = K\omega_n^2 x(t) \quad (2.3-41)$$

where K represents the steady-state gain of the system, and ζ and ω_n are the damping ratio and natural frequency parameters introduced in Section 2.3-1. Suppose that the problem is to find the unit-step response of the system assuming zero initial conditions. Unfortunately, the unit-step response relationships are different for the underdamped, critically-damped, and overdamped cases and they must be considered separately.

For the underdamped case, $\zeta < 1$, the eigenvalues are

$$s_1 = -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2} \quad \text{and} \quad s_2 = -\zeta\omega_n - j\omega_n\sqrt{1-\zeta^2} \quad (2.3-42)$$

which suggest the complementary solution

$$y_h(t) = e^{-\zeta\omega_n t} [k_1 \cos \omega_n \sqrt{1-\zeta^2} t + k_2 \sin \omega_n \sqrt{1-\zeta^2} t] \quad (2.3-43)$$

For a unit-step input, the right-hand side of the original equation is a constant which suggests the particular solution

$$y_p(t) = A \quad (2.3-44)$$

Substituting $y_p(t)$ in the entire equation yields

$$y_p(t) = K \quad (2.3-45)$$

and the total solution becomes

$$y(t) = K + e^{-\zeta\omega_n t} [k_1 \cos \omega_n \sqrt{1-\zeta^2} t + k_2 \sin \omega_n \sqrt{1-\zeta^2} t] \quad (2.3-46)$$

Applying the initial conditions $y(0)=0$ and $y'(0)=0$ and solving the resulting simultaneous equations gives the following values for the constants

$$k_1 = -K, \text{ and } k_2 = -\frac{K\zeta}{\sqrt{1-\zeta^2}}$$

Then, the completely defined unit-step response of the second-order underdamped system becomes

$$y(t) = K\left\{1 - e^{-\zeta\omega_n t}\left[\cos\omega_n\sqrt{1-\zeta^2}t + \frac{\zeta}{\sqrt{1-\zeta^2}}\sin\omega_n\sqrt{1-\zeta^2}t\right]\right\}u(t) \quad (2.3-47)$$

A typical unit-step response of a second-order, underdamped system is shown plotted in Fig. 2.3-3. Note that the independent variable is $\omega_n t$, with $\omega_n = 1$ (normalized).

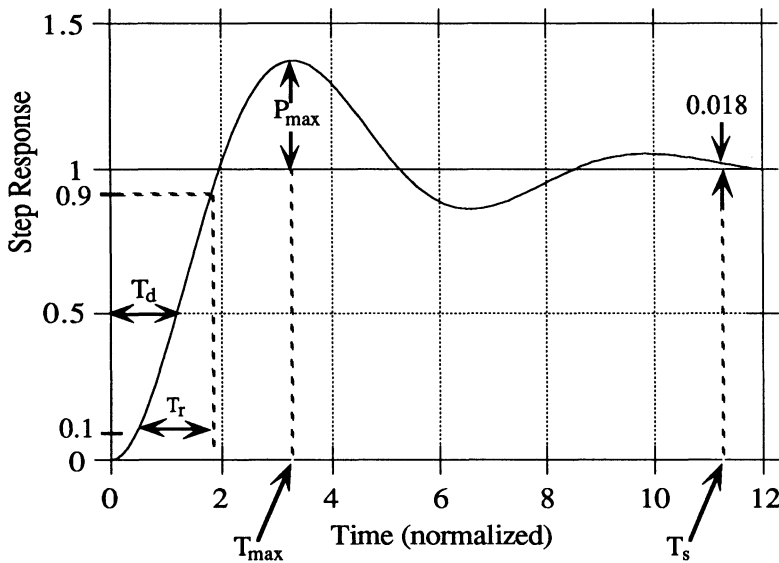


Fig. 2.3-3 Unit-step response (vs. $\omega_n t$) of a typical second-order system.

In Section 2.3-1, we defined the time constant of a second-order, underdamped system as $1/\zeta\omega_n$ and its settling time as $4/\zeta\omega_n$. Additional quantities that help describe the performance characteristics of such a system include the **delay time** T_d , **rise time** T_r , and **maximum overshoot** $P_{\max}$.

The delay time is the time required for the step response to reach 50 percent of its final value. The rise time is generally defined as the time required for the step response to rise from 10 percent to 90 percent of its final value. Maximum overshoot refers to the largest deviation of the response over its final value which occurs during the transient state.

To determine the time $T_{\max}$ at which the maximum overshoot occurs, we differentiate the expression for the unit-step response of the system given by Eq. (2.3-

47) with $K=1$ and set the result equal to zero. This gives the time at which the maximum overshoot occurs as

$$T_{\max} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} = \frac{\pi}{\omega_d} \quad (2.3-48)$$

and emphasizes its relationship to the damping ratio and natural frequency.

To determine the magnitude of the maximum overshoot $P_{\max}$, we substitute the above expression for $T_{\max}$ into Eq. (2.3-47). Assuming $K=1$, this gives the maximum response value $Y_{\max}$ as

$$Y_{\max} = 1 + e^{-\pi\zeta/\sqrt{1-\zeta^2}} \quad (2.3-49)$$

and the amount of maximum overshoot as

$$P_{\max} = e^{-\pi\zeta/\sqrt{1-\zeta^2}} \quad (2.3-50)$$

or, in percent,

$$P_{\max} = 100e^{-\pi\zeta/\sqrt{1-\zeta^2}} \quad (2.3-51)$$

To find the unit-step response of the critically-damped case, the value of $\zeta=1$ is substituted in Eq. (2.3-42). This results in the real and equal eigenvalues

$$s_1 = -\omega_n \quad \text{and} \quad s_2 = -\omega_n \quad (2.3-52)$$

which imply the complementary solution

$$y_h(t) = e^{-\omega_n t}(k_1 + k_2 t) \quad (2.3-53)$$

A unit-step input results again in the particular solution $y_p(t)=K$, the steady-state gain. Finally, evaluating k_1 and k_2 using the zero initial conditions results in the desired solution function

$$y(t) = K(1 - e^{-\omega_n t} - \omega_n t e^{-\omega_n t})u(t) \quad (2.3-54)$$

For the overdamped case or for $\zeta>1$, the real and distinct eigenvalues are

$$s_1 = -\zeta\omega_n + \omega_n\sqrt{\zeta^2-1} \quad \text{and} \quad s_2 = -\zeta\omega_n - \omega_n\sqrt{\zeta^2-1} \quad (2.3-55)$$

The particular solution is again equal to the steady-state gain K , and the total solution becomes

$$y(t) = K + k_1 e^{-\omega_n(\zeta+\sqrt{\zeta^2-1})t} + k_2 e^{-\omega_n(\zeta-\sqrt{\zeta^2-1})t}u(t) \quad (2.3-56)$$

The constants k_1 and k_2 are evaluated by applying the zero initial conditions. The completely defined total solution is

$$y(t) = K\left[1 + \frac{\zeta-\sqrt{\zeta^2-1}}{2\sqrt{\zeta^2-1}} e^{-\omega_n(\zeta+\sqrt{\zeta^2-1})t} - \frac{\zeta+\sqrt{\zeta^2-1}}{2\sqrt{\zeta^2-1}} e^{-\omega_n(\zeta-\sqrt{\zeta^2-1})t}\right]u(t) \quad (2.3-57)$$

The unit-step responses of a unity-gain, second-order system when the damping ratio ζ is varied from 0.3 to 1.5 are shown in Fig. 2.3-4. Note how values of ζ much smaller than 1 increase the maximum overshoot as well as the settling time. It is also evident that small values of ζ result in short rise and delay times. If the value of ζ is fixed, the time $T_{\max}$ varies inversely with ω_n . That is, a large ω_n means a small $T_{\max}$ and a quick-responding system. Conversely, a small ω_n means a large $T_{\max}$ and therefore a slower-responding system.

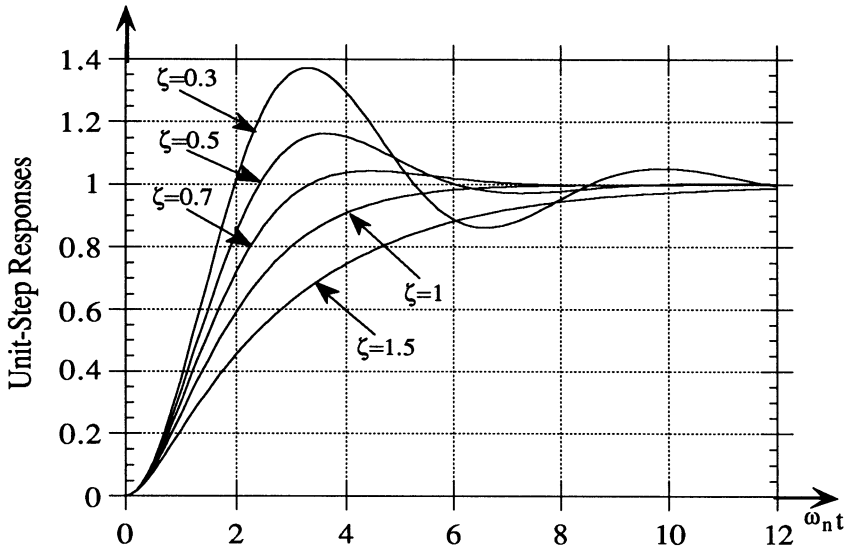


Fig. 2.3-4 Unit-step responses of second-order systems for various values of ζ .

Even though we can use the appropriate equations to calculate values for $P_{\max}$ and T_s , it is not possible to formulate exact analytical expressions for T_r and T_d as functions of ζ and ω_n . In other words, if the unit-step response function of Eq. (2.3-47) is set equal to 0.5, we can not determine the delay T_d analytically. We can, however, use computer methods to obtain approximate solutions numerically. Results obtained in this manner are shown tabulated in Table 2.3-2, and plotted in Fig. 2.3-5.

The tabular data can also be fitted by simple polynomial or exponential functions, which can then be used to calculate approximate values for T_r and T_d . For instance, in the region $0.3 < \zeta < 1.5$, the delay time can be approximated by the first-degree polynomial

$$T_d = \frac{0.87\zeta + 0.852}{\omega_n} \quad (2.3-58)$$

Table 2.3-2 Variation of T_d and T_r versus ζ

Damping Ratio (ζ)	Delay Time ($\omega_n T_d$)	Rise Time ($\omega_n T_r$)
0.0	1.05	1.02
0.1	1.09	1.10
0.2	1.13	1.20
0.3	1.18	1.32
0.4	1.24	1.46
0.5	1.29	1.64
0.6	1.36	1.85
0.7	1.43	2.13
0.8	1.50	2.47
0.9	1.59	2.89
1.0	1.68	3.36
1.1	1.78	3.86
1.2	1.88	4.37
1.3	1.99	4.88
1.4	2.11	5.37
1.5	2.22	5.86
1.6	2.35	6.34
1.7	2.47	6.81
1.8	2.60	7.29
1.9	2.73	7.76
2.0	2.86	7.85

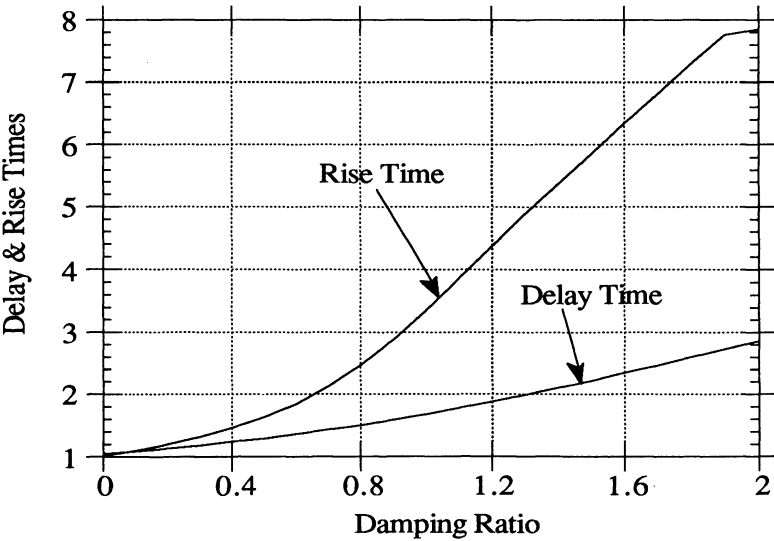


Fig. 2.3-5 Plots of rise time (T_r) and delay time (T_d) versus ζ .

and the rise time by the second-degree polynomial

$$T_r = \frac{1.942\zeta^2 + 0.442\zeta + 0.947}{\omega_n} \quad (2.3-59)$$

These concepts are illustrated by the following specific examples.

Example 2.3-13

Assuming zero initial conditions, solve the equation

$$y''(t) + 2y'(t) + 5y(t) = 5u(t)$$

Note that the complementary solution is identical to that obtained in Example 2.3-4. That is, this is an underdamped system for which

$$y_h(t) = k_1 e^{-t} \cos 2t + k_2 e^{-t} \sin 2t$$

The right-hand side term of 5 suggests the particular solution form

$$y_p(t) = \frac{5}{5} = 1$$

Thus, the complete solution becomes

$$y(t) = k_1 e^{-t} \cos 2t + k_2 e^{-t} \sin 2t + 1$$

The values of k_1 and k_2 can be determined by applying the initial conditions $y(0)=0$ and $y'(0)=0$, and then solving the resulting equations. This results in

$$y(t) = [1 - e^{-t}(\cos 2t + 0.5 \sin 2t)]u(t)$$

This can also be obtained by substituting the parameter values $K=1$, $\zeta=0.447$, and $\omega_n=2.236$ rad/sec in Eq. (2.3-47).

The time constant of this system is 1 sec, and its settling time is 4 sec. In other words, the oscillations, in this case, due to the transient part of the response, die out in about 4 sec. By this time, the response approaches its steady-state value of 1. The frequency of oscillations is 2 rad/sec, which suggests a period of 3.14159 sec, or a little more than one cycle of oscillations, before the response reaches steady state.

Example 2.3-14

Assuming the system is at rest, solve the differential equation

$$y''(t) + 3y'(t) + 2y(t) = (10 + e^{-t})u(t)$$

The complementary solution was determined in Example 2.3-2 as

$$y_h(t) = k_1 e^{-t} + k_2 e^{-2t}$$

The term 10 on the right side implies the root

$$s'_1 = 0$$

and contributes the constant A to the particular solution. The exponential term e^{-t} implies the root

$$s'_2 = -1$$

which is real and repeated because it has occurred already as an eigenvalue. The particular solution then becomes

$$y_p(t) = A + Bte^{-t}$$

which upon substitution in the original equation results in the relationship

$$2A + Be^{-t} = 10 + e^{-t}$$

Equating the constant and exponential terms, respectively, gives $A=5$ and $B=1$. The total solution is

$$y(t) = k_1 e^{-t} + k_2 e^{-2t} + 5 + te^{-t}$$

Finally, evaluating k_1 and k_2 for zero initial conditions yields the result

$$y(t) = (-11e^{-t} + 6e^{-2t} + 5 + te^{-t})u(t)$$

Example 2.3-15

Solve the differential equation

$$y'''(t) + 7y''(t) + 14y'(t) + 8y(t) = 10tu(t)$$

The characteristic equation is

$$s^3 + 7s^2 + 14s + 8 = 0$$

Its characteristic roots are

$$s_1 = -1, \quad s_2 = -2, \quad \text{and} \quad s_3 = -4$$

The complementary solution is

$$y(t) = k_1 e^{-t} + k_2 e^{-2t} + k_3 e^{-4t}$$

The right-hand side first-degree-polynomial term $10t$ implies the real and repeated roots

$$s'_1 = 0, \text{ and } s'_2 = 0$$

which result in the particular solution

$$y_p(t) = A + Bt$$

Substituting $y_p(t)$ into the original equation gives

$$14B + 8A + 8Bt = 10t$$

from which we obtain the values $B=1.25$ and $A=-2.1875$. The total solution becomes

$$y(t) = (k_1 e^{-t} + k_2 e^{-2t} + k_3 e^{-4t} - 2.1875 + 1.25t)u(t)$$

The constants k_1 , k_2 , and k_3 depend on initial conditions.

2.4 THE LAPLACE TRANSFORM

The Laplace transform is a fundamental mathematical tool used extensively for the analysis, modeling and design of systems. *The Laplace transform of a function $f(t)$ is defined by*

$$\mathcal{L}[f(t)] = F(s) = \int_0^{\infty} f(t)e^{-st}dt \quad (2.4-1)$$

The variable t represents time and s is the complex frequency variable

$$s = \sigma + j\omega = \sigma + j2\pi f \quad (2.4-2)$$

where σ is real, ω represents frequency in rad/sec, and f denotes frequency in Hz.

Eq. (2.4-1) is also known as the **one-sided** or **unilateral Laplace transform**, because it applies to $f(t)$ for $t \geq 0$ only. If $f(t)$ is nonzero for $t < 0$, then the **bilateral Laplace transform** with integration limits from $-\infty$ to $+\infty$ is used. Generally, this latter case is not common in applications involving physical systems. Eq. (2.4-1) transforms the function $f(t)$ from the time or t -domain to the frequency or s -domain. The time function $f(t)$ and its transform $F(s)$ are called a **transform pair**.

In order for the transform $F(s)$ to exist or, equivalently, for $f(t)$ to be Laplace transformable, the function $f(t)$ must be piecewise continuous and of **exponential order** for all $t > 0$. A function $f(t)$ is of exponential order if for some real number a ,

$$\lim_{t \rightarrow \infty} f(t)e^{at} = 0 \quad (2.4-3)$$

An equivalent interpretation is that a function is of exponential order if it can be fitted under an exponential function. Fortunately, functions that do not satisfy this property are very seldom encountered in the physical sciences.

Equally important to finding $F(s)$ from $f(t)$ is the inverse problem of finding $f(t)$ from its transform $F(s)$. *The inverse Laplace expression is given by the complex inversion integral*

$$\mathcal{L}^{-1}[F(s)] = f(t) = \frac{1}{2\pi j} \int_{\sigma_1 - j\infty}^{\sigma_1 + j\infty} F(s)e^{st}ds \quad (2.4-4)$$

where the path of integration is along the vertical line $s=\sigma_1$ from $-j\infty$ to $+j\infty$.

Evaluation of $f(t)$ from $F(s)$ using the inversion integral is tedious and impractical. This approach becomes unnecessary, because of the uniqueness property of the Laplace transform. *The uniqueness property states that for every $F(s)$ there is only one $f(t)$ and suggests the formation of a table of transform pairs starting with $f(t)$ and finding the corresponding $F(s)$ using Eq. (2.4-1).*

The table of transform pairs is used to find $f(t)$ from a given $F(s)$. Sometimes it may be necessary to manipulate and rearrange $F(s)$ into a form that appears in the table. Obtaining the inverse of $F(s)$ from Laplace transform tables is analogous to obtaining an original number from its antilogarithm using logarithmic tables or a calculator.

2.4-1 Laplace Transform Theorems

The following theorems are useful when evaluating Laplace transforms.

Theorem 1. Linearity

If $f_1(t)$ and $f_2(t)$ are both Laplace transformable and if a and b are arbitrary constants, then

$$\mathcal{L}[af_1(t) + bf_2(t)] = aF_1(s) + bF_2(s) \quad (2.4-5)$$

Proof: The proof of this theorem follows directly from the definition of the Laplace transform, that is

$$\begin{aligned} \mathcal{L}[af_1(t) + bf_2(t)] &= \int_0^{\infty} [af_1(t) + bf_2(t)]e^{-st}dt \\ &= a \int_0^{\infty} f_1(t)e^{-st}dt + b \int_0^{\infty} f_2(t)e^{-st}dt = aF_1(s) + bF_2(s) \end{aligned}$$

Theorem 2. Differentiation

If the Laplace transform of $f(t)$ is $F(s)$ and if $f'(t)$ is Laplace transformable, then

$$\mathcal{L}[f'(t)] = sF(s) - f(0) \quad (2.4-6)$$

Proof: Applying the Laplace transform definition to the function $f'(t)$ gives

$$\mathcal{L}[f'(t)] = \int_0^{\infty} f'(t)e^{-st}dt \quad (2.4-7)$$

which can be evaluated using integration by parts, that is,

$$\int_a^b u dv = uv \Big|_a^b - \int_a^b v du \quad (2.4-8)$$

Substituting $u=e^{-st}$, $du=-se^{-st}dt$, $v=f(t)$, and $dv=f'(t)dt$ in Eq. (2.4-7), we obtain

$$\begin{aligned}\mathcal{L}[f'(t)] &= f(t)e^{-st} \Big|_0^{\infty} + s \int_0^{\infty} f(t)e^{-st} dt \\ &= \lim_{t \rightarrow \infty} [f(t)e^{-st}] - f(0) + s \int_0^{\infty} f(t)e^{-st} dt = sF(s) - f(0)\end{aligned}$$

The assumption made is that $f(t)$ is of exponential order, that is,

$$\lim_{t \rightarrow \infty} f(t)e^{-st} = 0$$

For a second-order derivative term, it can be shown that

$$\mathcal{L}[f''(t)] = s^2F(s) - sf(0) - f'(0)$$

and, for a general, n th-order derivative term the expression becomes

$$\mathcal{L}[f^{(n)}(t)] = s^n F(s) - \sum_{k=1}^n s^{n-k} f^{(k-1)}(0) \quad (2.4-9)$$

Theorem 3. Integration

If the Laplace transform of $f(t)$ is $F(s)$ and if the definite integral of $f(t)$ is Laplace transformable, then

$$\mathcal{L}\left[\int_0^t f(t)dt\right] = \frac{F(s)}{s} \quad (2.4-10)$$

For the indefinite integral case, on the other hand, the integration theorem is

$$\mathcal{L}\left[\int f(t)dt\right] = \frac{F(s)}{s} + \frac{f^{-1}(0+)}{s} \quad (2.4-11)$$

where $f^{-1}(0+) = \lim_{t \rightarrow 0} \int_0^t f(t)dt$, for $t > 0$.

Theorem 4. Translation in Time

If the Laplace transform of $f(t)$ is $F(s)$ and T is any positive real number, then

$$\mathcal{L}[f(t-T)u(t-T)] = e^{-sT}F(s) \quad (2.4-12)$$

In other words, the Laplace transform of a function shifted or delayed by T sec in the time domain is the Laplace transform of the unshifted function multiplied by the exponential term e^{-sT} in the frequency domain. The term e^{-sT} is known as the **shift** or **delay operator**; T denotes the amount of shift or delay in seconds.

Proof: The proof follows from the basic definition. That is

$$\mathcal{L}[f(t-T)u(t-T)] = \int_0^{\infty} f(t-T)u(t-T)e^{-st}dt = e^{-sT} \int_0^{\infty} f(\tau)e^{-s\tau}d\tau = e^{-sT}F(s)$$

where the substitution $\tau=t-T$ was made and the integration limit adjusted accordingly.

The remaining theorems are presented without proof and may be used as exercises for the student.

Theorem 5. Translation in Frequency (Multiplication by e^{-at})

If the Laplace transform of $f(t)$ is $F(s)$ and a is an arbitrary constant, either real or complex, then

$$\mathcal{L}[e^{-at}f(t)] = F(s+a) \quad (2.4-13)$$

In other words, the transform of the product $e^{-at}f(t)$ can be obtained from $F(s)$ by replacing s by $(s+a)$.

Theorem 6. Complex Differentiation (Multiplication by t)

If the Laplace transform of $f(t)$ is $F(s)$, then

$$\mathcal{L}[tf(t)] = -\frac{d}{ds} F(s) \quad (2.4-14)$$

Thus, multiplication by t in the time domain is equivalent to differentiation with respect to s in the frequency domain.

Theorem 7. Complex Integration (Division by t)

If the Laplace transform of $f(t)$ is $F(s)$ and the limit of $f(t)/t$ as $t \rightarrow 0$ exists, then

$$\mathcal{L}\left[\frac{f(t)}{t}\right] = \int_s^{\infty} F(s)ds \quad (2.4-15)$$

In other words, division by t in the time domain is equivalent to integration with respect to s in the frequency domain.

Theorem 8. Initial Value Theorem

If $f(t)$ and its derivative are Laplace transformable and if the limit of $sF(s)$ as s approaches infinity exists, then

$$\lim_{t \rightarrow 0} f(t) = \lim_{s \rightarrow \infty} sF(s) \quad (2.4-16)$$

where $F(s)$ is the transform of $f(t)$. The initial value theorem states that the behavior of $f(t)$ near $t=0$ is related to the behavior of $sF(s)$ as s approaches infinity.

Theorem 9. Final Value Theorem

If $f(t)$ and its derivative are Laplace transformable and if the limit of $f(t)$ as t approaches infinity exists, then

$$\lim_{t \rightarrow \infty} f(t) = \lim_{s \rightarrow 0} sF(s) \quad (2.4-17)$$

where $F(s)$ is the transform of $f(t)$.

The final value theorem states that the behavior of $f(t)$ as t approaches infinity is related to the behavior of $sF(s)$ near $s=0$. Eq. (2.4-17) requires that all poles of $sF(s)$, or roots of its denominator, have negative real parts, except for a pole at the origin. Otherwise, the theorem cannot be applied.

Theorem 10. Convolution Integral

If the functions $f_1(t)$ and $f_2(t)$ are Laplace transformable with transforms $F_1(s)$ and $F_2(s)$, respectively, then the transform $F(s)$ composed of the product of $F_1(s)$ and $F_2(s)$ has the inverse transform $f(t)$ given by

$$f(t) = f_1(t) * f_2(t) = \mathcal{L}^{-1}[F_1(s)F_2(s)] = \int_0^t f_1(\tau)f_2(t-\tau)d\tau \text{ or } \int_0^t f_2(\tau)f_1(t-\tau)d\tau \quad (2.4-18)$$

The notation $f_1(t) * f_2(t)$ denotes the **convolution** of $f_1(t)$ and $f_2(t)$. The integral is known as the convolution integral; the parameter τ is a dummy variable.

The convolution integral often provides a direct solution to the inverse Laplace transform problem of $F(s)=F_1(s)F_2(s)$, if the inverse transforms of $F_1(s)$ and $F_2(s)$ are known.

Theorem 11. Periodic Functions

A function $f(t)$ is said to be periodic with period T if $f(t) = f(t+nT)$ for any integer n , positive or negative. The Laplace transform of a periodic function is given by

$$F(s) = \frac{1}{1-e^{-sT}} \int_0^T f(t)e^{-stdt} \quad (2.4-19)$$

2.4-2 TABLES OF LAPLACE TRANSFORMS

The Laplace transform definition given by Eq. (2.4-1) is now applied to develop a brief table of common Laplace transform pairs.

Example 2.4-1 The unit-step function $u(t)$

Given $f(t)=u(t)$, the problem is to find its associated Laplace transform $F(s)$. From Eq. (2.4-1), the Laplace transform of the unit-step function is found as

$$F(s) = \mathcal{L}[u(t)] = \int_0^{\infty} u(t)e^{-st}dt = \int_0^{\infty} e^{-st}dt = -\frac{1}{s}e^{-st}\Big|_0^{\infty} = -\frac{1}{s}(0-1) = \frac{1}{s}$$

Example 2.4-2 The exponential function e^{-at}

If $f(t) = e^{-at}$, then

$$F(s) = \mathcal{L}[e^{-at}] = \int_0^{\infty} e^{-at}e^{-st}dt = \int_0^{\infty} e^{-(s+a)t}dt = -\frac{1}{s+a}e^{-(s+a)t}\Big|_0^{\infty} = \frac{1}{s+a}$$

Note that we can obtain the same answer by applying Theorem 5 to the result of Example 2.4-1. That is, when $f(t)=u(t)$, its corresponding transform is $F(s)=1/s$. Consequently, when $f(t)$ is multiplied by e^{-at} , its transform is obtained by replacing s by $(s+a)$ in $F(s)=1/s$.

Example 2.4-3 The exponential function e^{at}

Since a in Example 2.4-2 is an arbitrary constant, it can be replaced by $-a$, another constant. The result is

$$\mathcal{L}[e^{at}] = \frac{1}{s-a}$$

Example 2.4-4 The ramp function $tu(t)$

We can use the basic definition of Eq. (2.4-1) or we can apply Theorem 2. This suggests substituting

$$f(t) = tu(t), \text{ and } f'(t) = u(t)$$

in the expression

$$\mathcal{L}[f'(t)] = sF(s) - f(0)$$

Since

$$\begin{aligned}\mathcal{L}[u(t)] &= \frac{1}{s} \\ \mathcal{L}[tu(t)] &= -\frac{d}{ds} \mathcal{L}[u(t)] = \frac{1}{s^2}\end{aligned}$$

Example 2.4-5 $t^2u(t)$

If in Example 2.4-4 we substitute

$$f(t) = t^2u(t)$$

then, $f'(t) = 2tu(t)$, and $f(0) = 0$

Thus, using Theorem 2 again, the expression

$$\mathcal{L}[f'(t)] = sF(s) - f(0)$$

gives

$$\mathcal{L}[2tu(t)] = s\mathcal{L}[t^2u(t)] - 0$$

or

$$\frac{2}{s^2} = s\mathcal{L}[t^2u(t)]$$

The desired result then becomes

$$\mathcal{L}[t^2u(t)] = \frac{2}{s^3}$$

Example 2.4-6 $t^n u(t)$

Generalizing the last two examples leads to the result

$$\mathcal{L}[t^n u(t)] = \frac{n!}{s^{n+1}}$$

where n is an integer >0 .

Example 2.4-7 $te^{-at}u(t)$

From Example 2.4-4 and Theorem 5, we can write

$$\mathcal{L}[te^{-at}u(t)] = -\frac{d}{ds} \mathcal{L}[e^{-at}u(t)] = -\frac{d}{ds} \frac{1}{(s+a)} = \frac{1}{(s+a)^2}$$

The same result can be obtained by using Theorem 6 with $f(t)=e^{-at}$.

Example 2.4-8 $\sin bt$

From the Laplace transform definition,

$$\mathcal{L}[\sin bt] = F(s) = \int_0^{\infty} \sin bt \, e^{-st} dt$$

By adding and subtracting the two equations in Euler's identity

$$e^{jbt} = \cos bt + j\sin bt \quad \text{and} \quad e^{-jbt} = \cos bt - j\sin bt$$

we obtain,

$$\cos bt = \frac{e^{jbt} + e^{-jbt}}{2}$$

and

$$\sin bt = \frac{e^{jbt} - e^{-jbt}}{2j}$$

Substituting the last relationship in the integral equation yields

$$\mathcal{F}[\sin bt] = \frac{1}{2j} \int_0^{\infty} (e^{jbt} - e^{-jbt})e^{-st} dt = \frac{1}{2j} \left[\int_0^{\infty} e^{-(s-jb)t} dt - \int_0^{\infty} e^{-(s+jb)t} dt \right]$$

or

$$\mathcal{F}[\sin bt] = \frac{1}{2j} \left[\frac{1}{s-jb} - \frac{1}{s+jb} \right] = \frac{b}{s^2+b^2}$$

Example 2.4-9 $\cos bt$

Following a similar procedure, we can show that

$$\mathcal{F}[\cos bt] = \frac{s}{s^2+b^2}$$

Example 2.4-10 $e^{-at}\sin bt$

From Example 2.4-8 and Theorem 5, we obtain

$$\mathcal{F}[e^{-at}\sin bt] = \frac{b}{(s+a)^2+b^2}$$

Example 2.4-11 $e^{-at}\cos bt$

From Example 2.4-9 and Theorem 5,

$$\mathcal{F}[e^{-at}\cos bt] = \frac{s+a}{(s+a)^2+b^2}$$

Example 2.4-12 The unit-impulse function $\delta(t)$

Consider a unit-area rectangular pulse of duration or width T sec. The analytic expression for the pulse function is

$$f(t) = \frac{u(t) - u(t-T)}{T}$$

and its transform becomes

$$F(s) = \frac{1 - e^{-sT}}{Ts}$$

If the width T of the pulse is decreased, its height $1/T$ increases but the area under the pulse (its strength) remains unity. As T approaches zero, the limit of $f(t)$ is termed the **unit-impulse** function, usually denoted by $\delta(t)$. To obtain the Laplace transform of the impulse function $\delta(t)$, we evaluate the limit of $F(s)$ as s approaches 0. This leads to $0/0$,

an indeterminate result. Hence, applying L'Hospital's rule to the above function gives the Laplace transform of $\delta(t)$ as 1. That is,

$$\mathcal{L}[\delta(t)] = \lim_{T \rightarrow 0} \frac{1 - e^{-sT}}{Ts} = 1$$

This result can also be obtained from the basic Laplace transform definition, that is,

$$\mathcal{L}[\delta(t)] = \int_0^{\infty} \delta(t) e^{-st} dt = e^{-st} \Big|_{t=0} = 1$$

since the last integral represents the area under the $\delta(t)$ curve. Equivalently, the unit-impulse function "picks out" the value 1 of the function e^{-st} at the instant $t=0$ when the impulse occurs.

The Laplace transform pairs developed so far are included in Table 2.4-1. The use of the Table is illustrated by means of examples.

Example 2.4-13

Determine the function $f(t)$, if its transform $F(s)$ is given by

$$F(s) = \frac{10}{(s+2)}$$

Using Table 2.4-1, we obtain

$$f(t) = 10e^{-2t}u(t)$$

Example 2.4-14

Determine $f(t)$ if its transform is given by

$$F(s) = \frac{s+23}{s^2+6s+25}$$

When the denominator polynomial is of degree 2 or higher, we first factor the polynomial and find its roots. For this example, the roots are the complex-conjugate pair

$$s_1 = -3+j4 \quad \text{and} \quad s_2 = -3-j4$$

Table 2.4-1 *Laplace Transform Pairs*

Item	$f(t)$	$F(s)$
1	$\delta(t)$	1
2	$u(t)$	$\frac{1}{s}$
3	t	$\frac{1}{s^2}$
4	t^n	$\frac{n!}{s^{n+1}}$
5	$e^{\pm at}$	$\frac{1}{(s \mp a)}$
6	$t^n e^{\pm at}$	$\frac{n!}{(s \mp a)^{n+1}}$
7	$\sin bt$	$\frac{b}{(s^2 + b^2)}$
8	$\cos bt$	$\frac{s}{(s^2 + b^2)}$
9	$e^{\pm at} \sin bt$	$\frac{b}{[(s \mp a)^2 + b^2]}$
10	$e^{\pm at} \cos bt$	$\frac{(s \mp a)}{[(s \mp a)^2 + b^2]}$
11	$q(t-T)u(t-T)$	$e^{-sT}Q(s)$
12	$q^{(n)}(t)$	$s^n Q(s) - \sum_{k=1}^n s^{n-k} q^{(k-1)}(0)$

$F(s)$ can then be rearranged in the standard form

$$F(s) = \frac{s+23}{(s+3)^2+4^2}$$

so that the denominator can be found in Table 2.4-1. To find the appropriate $f(t)$, however, we must also match its numerator. To accomplish this, observe that the term s in the numerator can be picked up by the factor $(s+3)$ and that the rest of the numerator is constant. This suggests writing $F(s)$ in the form

$$F(s) = \frac{s+3}{(s+3)^2+4^2} + 5 \frac{4}{(s+3)^2+4^2}$$

The corresponding $f(t)$ is obtained from the table as

$$f(t) = (e^{-3t}\cos 4t + 5e^{-3t}\sin 4t)u(t)$$

Example 2.4-15

Determine the inverse transform $f(t)$ of the function

$$F(s) = \frac{10}{s(s+5)}$$

Notice that the denominator of $F(s)$ does not exist in Table 2.4-1. However, we can obtain the inverse transform $f(t)$ from $F(s)$ by using the convolution integral. From Eq. (2.4-18), the transform $F(s)$ can be written as the product of the transforms $F_1(s)$ and $F_2(s)$, where

$$F_1(s) = \frac{10}{s} \quad \text{and} \quad F_2(s) = \frac{1}{(s+5)}$$

The convolution integral of $f_1(t)$ and $f_2(t)$ gives

$$\begin{aligned} f(t) &= f_1(t) * f_2(t) = \int_0^t f_2(\tau) f_1(t-\tau) d\tau = 10 \int_0^t e^{-5\tau} u(t-\tau) d\tau \\ &= 10 \left[-\frac{e^{-5\tau}}{5} \right]_0^t = 10 \left[-\frac{e^{-5t}}{5} + \frac{1}{5} \right] = 2(1-e^{-5t})u(t) \end{aligned}$$

Example 2.4-16

Determine the Laplace transform of a pulse train with period T , pulse amplitude 1, and pulse width of a , as shown in Fig. 2.4-1a.

From Example 2.4-12, the Laplace transform of one cycle of the pulse function of Fig. 2.4-1a is

$$\frac{(1-e^{-as})}{s}$$

From Eq. (2.4-19) - Theorem 11, the transform of the periodic pulse train becomes

$$F(s) = \frac{1 - e^{-as}}{s(1 - e^{-sT})}$$

Example 2.4-17

Determine the Laplace transform of the unit-impulse train of period T , as shown in Fig. 2.4-1b.

From Example 2.4-12, the Laplace transform of the unit impulse is 1. Applying Theorem 11, we obtain the transform of the impulse train as

$$F(s) = 1 + e^{-sT} + e^{-2sT} + \dots = \sum_{n=0}^{\infty} e^{-nsT} = \frac{1}{1 - e^{-sT}}$$

which is an important result. The infinite series and closed-form expressions in the equation follow from the binomial theorem.

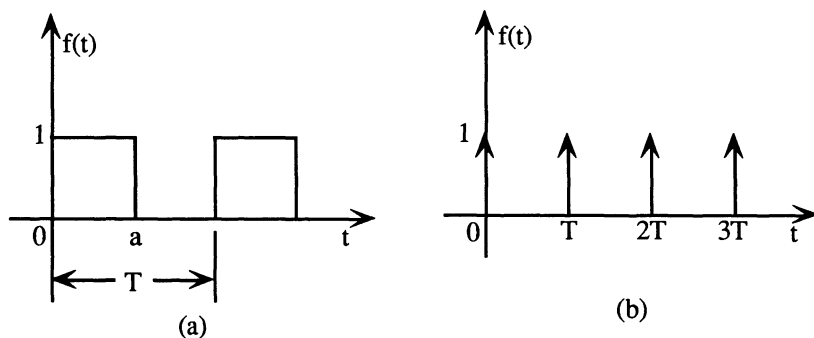


Fig. 2.4-1 Periodic pulse train (a) and impulse train (b) functions.

2.4-3 Partial Fraction Expansion

In this section, a class of Laplace transform pairs is introduced for which the general form of $F(s)$ can be expressed as the ratio of two polynomials $A(s)$ and $B(s)$ of degree m and n , respectively. That is,

$$F(s) = \frac{A(s)}{B(s)} = \frac{a_0 s^m + a_1 s^{m-1} + \dots + a_{m-1} s + a_m}{s^n + b_1 s^{n-1} + \dots + b_{n-1} s + b_n} \quad (2.4-20)$$

where m and n are positive integers and the a and b coefficients are real constants.

In the study of physical systems, $F(s)$ is normally a **proper function**, that is, the degree of its numerator polynomial is at least one less than that of its denominator or, equivalently, $m < n$. If this is not the case, the numerator is divided by the denominator as

many times as necessary to produce a polynomial in s plus a rational polynomial remainder which satisfies the proper function condition.

If $F(s)$ is a proper function, the first step is to factor the denominator polynomial $B(s)$ into first-order terms. Thus, $F(s)$ is written as

$$F(s) = \frac{A(s)}{(s-p_1)(s-p_2) \dots (s-p_n)} \quad (2.4-21)$$

where the roots $p_1, p_2, \dots, p_n$ may be real or complex. The roots of the denominator represent the zeros of the equation $B(s)=0$ and can be determined by numerical methods.

The next step is to expand the transform $F(s)$ into a series of partial fractions which can be easily inverted using Laplace transform tables. The form of the expansion depends on the type of roots of the denominator $B(s)$. The roots can be real and distinct, real and repeated, or complex. Simple complex roots are common, while repeated complex roots are rarely encountered in practical applications.

A method of partial fraction expansion generally referred to as the **Heaviside** partial fraction expansion is presented next.

Heaviside Expansion Theorem I: Real and Distinct Roots. Assume that the n denominator roots of $F(s)$ are real and distinct. Then, the partial fraction expansion is expressed as

$$F(s) = \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} + \dots + \frac{k_n}{s-p_n} \quad (2.4-22)$$

where the constants $k_1, k_2, \dots, k_n$ are called the **residues** of $F(s)$ at the corresponding roots. The inverse transform of $F(s)$ is given by

$$f(t) = k_1 e^{p_1 t} + k_2 e^{p_2 t} + \dots + k_n e^{p_n t} \quad (2.4-23)$$

The constants or residues $k_1, k_2, \dots, k_n$ can be determined as follows. To find k_n , multiply both sides of Eq. (2.4-22) by the term under k_n , namely $(s-p_n)$, and cancel the common factors. Then, replace s by p_n and obtain k_n by evaluating the expression

$$k_n = \frac{A(s)(s-p_n)}{B(s)} \Big|_{s=p_n} \quad (2.4-24)$$

The other residues can be determined in the same manner. This is referred to as Heaviside Expansion Theorem I.

Example 2.4-18

Determine $f(t)$ if $F(s)$ is given by

$$F(s) = \frac{3s+7}{s^3+6s^2+11s+6} = \frac{3s+7}{(s+1)(s+2)(s+3)}$$

The partial fraction expansion form of $F(s)$ is

$$F(s) = \frac{k_1}{s+1} + \frac{k_2}{s+2} + \frac{k_3}{s+3}$$

Using Heaviside's Theorem, we obtain

$$k_1 = \frac{3s+7}{(s+2)(s+3)} \Big|_{s=-1} = \frac{4}{2} = 2$$

$$k_2 = \frac{3s+7}{(s+1)(s+3)} \Big|_{s=-2} = \frac{1}{-1} = -1$$

and

$$k_3 = \frac{3s+7}{(s+1)(s+2)} \Big|_{s=-3} = \frac{-2}{2} = -1$$

Thus,

$$F(s) = \frac{2}{s+1} - \frac{1}{s+2} - \frac{1}{s+3}$$

From Table 2.4-1, the inverse transform of $F(s)$ is

$$f(t) = (2e^{-t} - e^{-2t} - e^{-3t})u(t)$$

Heaviside Expansion Theorem II: Real and Repeated Roots. If the n denominator roots of $F(s)$ are real and repeated, the partial fraction expansion form is

$$F(s) = \frac{A(s)}{(s-p)^n} = \frac{k_1}{s-p} + \frac{k_2}{(s-p)^2} + \dots + \frac{k_n}{(s-p)^n} \quad (2.4-25)$$

and the corresponding inverse transform

$$f(t) = e^{pt}(k_1 + k_2 t + \dots + k_n t^{n-1}) \quad (2.4-26)$$

The residue k_n can be evaluated in the same way as if the roots were real and distinct. That is, multiplying both sides of Eq. (2.4-25) by $(s-p)^n$ we obtain the equation

$$A(s) = k_1(s-p)^{n-1} + k_2(s-p)^{n-2} + \dots + k_n \quad (2.4-27)$$

from which k_n becomes

$$k_n = A(s) \Big|_{s=p} \quad (2.4-28)$$

However, the remaining residues cannot be evaluated in a similar manner, because this would lead to indeterminate values. Instead, each of the remaining residues is determined by successive differentiation and then evaluating the derivatives that result at $s=p$. The general expression for the j th residue is

$$k_j = \frac{1}{(n-j)!} \frac{d^{n-j}}{ds^{n-j}} A(s)(s-p)^n \Big|_{s=p} \quad (2.4-29)$$

where $j = n, n-1, \dots, 2$ and 1 . This method of partial fraction expansion is known as Heaviside Theorem II and is illustrated next by means of an example.

Example 2.4-19

Find the inverse transform $f(t)$ of the function

$$F(s) = \frac{s+1}{(s+2)^2}$$

The partial fraction expansion is

$$F(s) = \frac{k_1}{s+2} + \frac{k_2}{(s+2)^2}$$

Multiplying both sides of the equation by the term $(s+2)^2$ and simplifying gives the equation

$$s+1 = k_1(s+2) + k_2$$

which when evaluated for $s=-2$ yields the residue value $k_2=-1$. Differentiating the last equation with respect to 's' results in $k_1=1$. Thus, the transform is

$$F(s) = \frac{1}{s+2} - \frac{1}{(s+2)^2}$$

from which we obtain the inverse transform

$$f(t) = (e^{-2t} - te^{-2t})u(t)$$

Complex Roots. Complex roots always occur in complex-conjugate pairs whose real part may be either positive or negative. For the simple complex root case, $F(s)$ can be written as

$$F(s) = \frac{As+B}{(s+a)^2+b^2} \quad (2.4-30)$$

where A , B , a , and b are real constants. A possible partial fraction expansion form is given by

$$F(s) = \frac{As+B}{(s+a+jb)(s+a-jb)} = \frac{k_1}{s+a+jb} + \frac{k_2}{s+a-jb} \quad (2.4-31)$$

where the residues k_1 and k_2 are complex-conjugate constants. The residues can be evaluated using Heaviside's Theorem I and complex arithmetic.

A more practical scheme which avoids complex arithmetic is to expand $F(s)$ in the form

$$F(s) = \frac{As+B}{(s+a)^2+b^2} = \frac{k_3(s+a)}{(s+a)^2+b^2} + \frac{k_4b}{(s+a)^2+b^2} \quad (2.4-32)$$

The constants k_3 and k_4 are real numbers which are evaluated by equating the numerator coefficients. The result is

$$k_3 = A \quad \text{and} \quad k_4 = \frac{B-aA}{b} \quad (2.4-33)$$

From the transform table, the inverse transform is written as

$$f(t) = k_3 e^{-at} \cos bt + k_4 e^{-at} \sin bt \quad (2.4-34)$$

The general procedure is illustrated by the following example.

Example 2.4-20

Determine the inverse transform of

$$F(s) = \frac{2s+40}{(s+5)^2+10^2}$$

The transform $F(s)$ can be expanded in the form

$$F(s) = \frac{2(s+5)}{(s+5)^2+10^2} + \frac{3*10}{(s+5)^2+10^2}$$

Using the table, we obtain

$$f(t) = 2e^{-5t} \cos 10t + 3e^{-5t} \sin 10t$$

For repeated complex roots, a similar procedure developed for repeated real roots can be followed.

A combination of more than one of the previous partial fraction expansion techniques may be necessary to expand $F(s)$ into a form which is easy to invert using Laplace transform tables. The general approach is illustrated by examples.

Example 2.4-21

Find $f(t)$ if $F(s)$ is given by

$$F(s) = \frac{125}{s[(s+5)^2+10^2]} = \frac{k_1}{s} + \frac{k_2 s + k_3}{(s+5)^2+10^2}$$

The residue k_1 is readily obtained from

$$k_1 = \frac{125}{(s+5)^2+10^2} \Big|_{s=0} = \frac{125}{25+100} = 1$$

To evaluate the coefficients k_2 and k_3 , the right side of the above equation is placed under the common denominator of the left side. Then, the numerators of both sides are equated to give

$$125 = s^2 + 10s + 125 + k_2 s^2 + k_3 s$$

By equating similar coefficients, we get

$$1 + k_2 = 0, \text{ or } k_2 = -1$$

and $10+k_3 = 0$, or $k_3 = -10$

Thus, the transform function becomes

$$F(s) = \frac{1}{s} - \frac{s+10}{(s+5)^2+10^2} = \frac{1}{s} - \frac{s+5}{(s+5)^2+10^2} - \frac{0.5*10}{(s+5)^2+10^2}$$

Finally, the inverse transform is

$$f(t) = (1 - e^{-5t}\cos 10t - 0.5e^{-5t}\sin 10t)u(t)$$

Example 2.4-22

Determine $f(t)$ if

$$F(s) = \frac{3s+5}{(s+1)^2(s+2)}$$

Performing partial fraction expansion gives

$$F(s) = \frac{k_1}{s+1} + \frac{k_2}{(s+1)^2} + \frac{k_3}{s+2}$$

The coefficients k_3 and k_2 then are

$$k_3 = \left. \frac{3s+5}{(s+1)^2} \right|_{s=-2} = \frac{-6+5}{1} = -1$$

$$k_2 = \left. \frac{3s+5}{s+2} \right|_{s=-1} = \frac{-3+5}{1} = 2$$

The coefficient k_1 can be obtained by differentiation, as suggested by Eq. (2.4-29), or algebraically by generalizing the method illustrated in Example 2.4-21. If applied from the beginning, this latter approach leads to the following system of simultaneous algebraic equations

$$2k_1 + 2k_2 + k_3 = 5$$

$$3k_1 + k_2 + 2k_3 = 3$$

$$k_1 + 0k_2 + k_3 = 0$$

In general, similar equations can be set up and solved simultaneously to evaluate the residues for any transform function $F(s)$. However, for this example $k_3=-1$ and the value of k_1 is readily obtained from the last equation as $k_1=1$. This yields

$$F(s) = \frac{1}{s+1} + \frac{2}{(s+1)^2} - \frac{1}{s+2}$$

and the inverse transform

$$f(t) = (e^{-t} + 2te^{-t} - e^{-2t})u(t)$$

Example 2.4-23

Determine $f(t)$ if its transform $F(s)$ is given by

$$F(s) = \frac{1}{(s^2 + \omega)^2}$$

We would be hard pressed if we tried to solve this problem by partial fraction expansion techniques. Try it!

As an alternative, however, we can use the convolution theorem, see Eq. (2.4-18). First, express $F(s)$ in the form

$$F(s) = \frac{1}{\omega} \frac{\omega}{(s^2 + \omega^2)} \frac{1}{\omega} \frac{\omega}{(s^2 + \omega^2)} = \frac{1}{\omega^2} \frac{\omega}{(s^2 + \omega^2)} \frac{\omega}{(s^2 + \omega^2)}$$

Then, using convolution,

$$\begin{aligned} \mathcal{F}^{-1}\left[\frac{1}{(s^2 + \omega)^2}\right] &= \frac{1}{\omega^2} \int_0^t \sin \omega(t - \tau) \sin \omega \tau d\tau = \frac{1}{2\omega^2} \left[\int_0^t \cos(2\omega\tau - \omega t) d\tau - \cos \omega t d\tau \right] \\ &= \frac{1}{2\omega^2} \left[\frac{\sin(2\omega\tau - \omega t)}{2\omega} - \cos \omega \tau \right] \Big|_0^t = \frac{1}{2\omega^2} \left[\frac{\sin \omega t}{\omega} - t \cos \omega t \right] \end{aligned}$$

Thus, if $\omega=1$,

$$\mathcal{F}^{-1}\left[\frac{1}{(s^2 + 1)^2}\right] = \frac{1}{2} (\sin t - t \cos t)$$

2.5 TIME RESPONSE USING TRANSFORM METHODS

As we have seen, solving differential equations by the classical method involves a number of separate steps. The Laplace transform method for solving differential equations, on the other hand, is an operational method that offers certain advantages over the classical method.

The main advantage of the Laplace transform method is that it is systematic. Also, the Laplace transform method gives the total or general solution, that is, complementary plus particular solution in one operation. Finally, the initial conditions are automatically included in the transformed equation.

For a linear, time-invariant system with input $x(t)$ and output $y(t)$, the Laplace transform method for solving a general differential equation consists basically of the following steps:

1. Finding the input transform $X(s)$ from $x(t)$.
2. Finding the output transform $Y(s)$ by Laplace transforming the differential equation.
3. Substituting initial conditions in the expression for $Y(s)$.

4. Finding the inverse transform $y(t)$ from $Y(s)$ using the Laplace transform tables.

To illustrate this method, consider a single-input-single-output (SISO), linear, time-invariant system given by

$$y'(t) + ay(t) = Kx(t) \quad (2.5-1)$$

where a and K are arbitrary constants. Taking the Laplace transform of both sides of the equation gives

$$sY(s) - y(0) + aY(s) = KX(s) \quad (2.5-2)$$

Therefore, the output transform $Y(s)$ becomes

$$Y(s) = \frac{y(0) + KX(s)}{s+a} \quad (2.5-3)$$

Assuming the system at rest, that is, assuming zero initial conditions, the generalized expressions for the unit-impulse, unit-step and unit-pulse responses for the system can be determined. This is discussed below.

Unit-Impulse Response. Substituting $y(0)=0$ in Eq. (2.5-3) gives the output transform

$$Y(s) = \frac{K}{s+a} X(s) \quad (2.5-4)$$

A unit-impulse input implies that $X(s)=1$. Eq. (2.5-4) then gives

$$Y(s) = \frac{K}{s+a} \quad (2.5-5)$$

and, from Table 2.4-1, the unit-impulse response becomes

$$y(t) = Ke^{-at}u(t) \quad (2.5-6)$$

Unit-Step Response. For a unit-step function input, the input transform is $X(s)=1/s$. Using this in Eq. (2.5-4) results in the output transform

$$Y(s) = \frac{K}{s(s+a)} \quad (2.5-7)$$

Partial fraction expansion of Eq. (2.5-7) gives

$$Y(s) = \frac{k_1}{s} + \frac{k_2}{s+a} = \frac{K}{a} \left[\frac{1}{s} - \frac{1}{s+a} \right] \quad (2.5-8)$$

From Table 2.4-1, the inverse transform $f(t)$ is

$$y(t) = \frac{K}{a} (1 - e^{-at})u(t) \quad (2.5-9)$$

which represents the unit-step response of the system.

The **steady-state value** of $y(t)$ in Eq. (2.5-1), also referred to as the **steady-state gain**, can be obtained by letting t approach infinity in Eq. (2.5-9). This causes the transient component of the unit-step response, in this case the exponential term, to vanish and leaving the value of K/a . The same result can be obtained by applying the final value theorem given by Eq. (2.4-17) to the output transform of Eq. (2.5-7). The latter approach is preferable, because it does not require solving for the step response of the system.

The behavior of the system at $t=0$ can be obtained by letting $t=0$ in Eq. (2.5-9) or by applying the initial value theorem of Eq. (2.4-16) to the output transform of Eq. (2.5-7). In either case, the initial value is $y(0)=0$.

Unit-Pulse Response. A unit-pulse function input can be expressed as

$$x(t) = \frac{[u(t) - u(t-T)]}{T} \quad (2.5-10)$$

where T denotes the duration or width of the pulse. The transform of the unit pulse is

$$X(s) = \frac{(1 - e^{-sT})}{sT} \quad (2.5-11)$$

To find the unit-pulse response of the first-order system represented by Eq. (2.5-1), substitute Eq. (2.5-11) into Eq. (2.5-4) and expand $Y(s)$ into partial fractions. This gives

$$Y(s) = \frac{K}{T} \frac{(1 - e^{-sT})}{s(s+a)} = \frac{K}{aT} \left[\frac{1}{s} - \frac{1}{s+a} - \left(\frac{e^{-sT}}{s} - \frac{e^{-sT}}{s+a} \right) \right] \quad (2.5-12)$$

Using Table 2.4-1, the unit-pulse response of the system can be written as

$$y(t) = \frac{K}{aT} [(1 - e^{-at})u(t) - (1 - e^{-a(t-T)})u(t-T)] \quad (2.5-13)$$

It is worth noting that for practical considerations the unit-pulse response of Eq. (2.5-13) approximates the ideal unit-impulse response of Eq. (2.5-6) if the width of the pulse is "narrow enough", say by a factor of 10 or more, compared to the time constant of the system. In general, for higher-order systems the width of the pulse must be narrow enough compared to the smallest time constant of the system. This guideline makes the laboratory use of impulse functions practical for testing purposes.

Consider the standard-form, second-order system introduced in Section 2.3-2. Its differential equation model is

$$y''(t) + 2\zeta\omega_n y'(t) + \omega_n^2 y(t) = K\omega_n^2 x(t) \quad (2.5-14)$$

Assuming zero initial conditions, the unit-impulse response for the underdamped case is determined from

$$Y(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (2.5-15)$$

It can be shown that the unit-impulse response is

$$y(t) = \frac{K\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin\omega_n[\sqrt{1-\zeta^2}] t \quad (2.5-16)$$

Analogous expressions can be derived for the critically-damped and overdamped cases. For the underdamped case with zero initial conditions, the unit-step response of the system represented by Eq. (2.5-14) is obtained from the transform

$$\begin{aligned} Y(s) &= \frac{K\omega_n^2}{s(s^2+2\zeta\omega_n s+\omega_n^2)} = \frac{k_1}{s} + \frac{k_2 s+k_3}{s^2+2\zeta\omega_n s+\omega_n^2} \\ &= \frac{K}{s} - K \left[\frac{s+\zeta\omega_n}{(s+\zeta\omega_n)^2+(\omega_n\sqrt{1-\zeta^2})^2} + \frac{\zeta}{\sqrt{1-\zeta^2}} \frac{\omega_n\sqrt{1-\zeta^2}}{(s+\zeta\omega_n)^2+(\omega_n\sqrt{1-\zeta^2})^2} \right] \end{aligned} \quad (2.5-17)$$

Taking the inverse transform of $Y(s)$, the unit-step response is found to be

$$y(t) = K \left\{ 1 - e^{-\zeta\omega_n t} \left[\cos\omega_n\sqrt{1-\zeta^2} t + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin\omega_n\sqrt{1-\zeta^2} t \right] \right\} u(t) \quad (2.5-18)$$

This agrees with Eq. (2.3-47) derived earlier using classical methods. Similar unit-step responses can be derived for the critically-damped and overdamped cases. Results should match those given by Eq. (2.3-54) and Eq. (2.3-57), respectively.

Even though time response characteristics of first-order and second-order systems have been emphasized, the analysis of higher-order systems can often be carried out by assuming that such systems have a response dominated by a pair of complex denominator roots near the origin. This pair of complex roots is known as a **dominant pair** and the corresponding system is said to have a **quadratic mode**.

In order to perform a complete quantitative analysis of a higher-than-second-order system, that is, in order to determine such quantities as overshoot, delay time and rise time, it becomes desirable and sometimes necessary to use a digital computer. The method of solving differential equations by Laplace transforms is illustrated further by examples.

Example 2.5-1

Solve the following differential equation using the Laplace transform method. Assume zero initial conditions.

$$y'(t) + y(t) = 2tu(t)$$

Taking the Laplace transform on both sides of the equation we obtain

$$sY(s) + Y(s) = \frac{2}{s^2}, \quad \text{or } Y(s) = \frac{2}{s^2(s+1)}$$

Using partial fraction expansion gives

$$Y(s) = \frac{k_1}{s} + \frac{k_2}{s^2} + \frac{k_3}{s+1} = -\frac{2}{s} + \frac{2}{s^2} + \frac{2}{s+1}$$

From Table 2.4-1, the solution is

$$y(t) = (-2 + 2t + 2e^{-t})u(t)$$

The above result agrees with that obtained in Example 2.3-3.

Example 2.5-2

Solve the following differential equation using Laplace transforms. Assume zero initial conditions.

$$y'(t) + y(t) = (1 + e^{-t})u(t)$$

Laplace transforming the equation gives

$$sY(s) + Y(s) = \frac{1}{s} + \frac{1}{s+1}$$

from which

$$Y(s) = \frac{2s+1}{s(s+1)^2} = \frac{-1}{s+1} + \frac{1}{(s+1)^2} + \frac{1}{s}$$

The solution is

$$y(t) = (-e^{-t} + te^{-t} + 1)u(t)$$

which is the same as the result obtained in Example 2.3-4.

Example 2.5-3

Solve the differential equation

$$y'(t) + y(t) = 10e^{-t}\sin t$$

assuming zero initial conditions. Laplace transforming the equation and rearranging gives

$$Y(s) = \frac{10}{(s+1)(s^2+2s+2)} = \frac{k_1}{s+1} + \frac{k_2s+k_3}{(s+1)^2+1} = \frac{10}{s+1} - \frac{10(s+1)}{(s+1)^2+1}$$

and

$$y(t) = 10(e^{-t} - e^{-t}\cos t)u(t)$$

The same result was obtained by classical methods in Example 2.3-12.

Example 2.5-4

A first-order system is described by the differential equation

$$y'(t) + y(t) = x(t)$$

Determine its response to a pulse of unit area and width T sec. Assume zero initial conditions. Plot the unit-impulse response of the system with the pulse responses obtained for $T=0.1$ sec (i.e., a pulse width 10 times narrower than the system time constant), and $T=1$ sec. If we substitute the values $a=1$ and $K=1$ in Eq. (2.5-12), we obtain the pulse response

$$y(t) = \frac{[(1-e^{-t})u(t) - (1-e^{-(t-T)})u(t-T)]}{T}$$

The unit-impulse response plots are shown in Fig. 2.5-1.

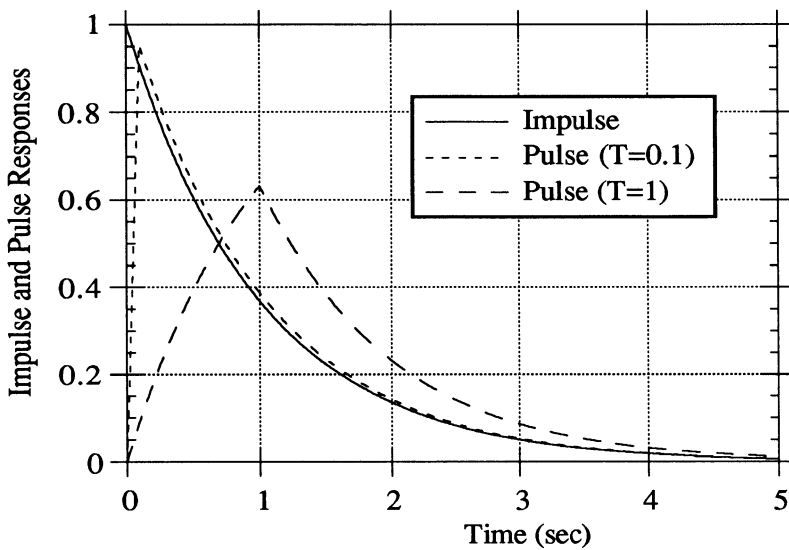


Fig. 2.5-1 Unit-impulse and unit-area pulse responses for the system of Example 2.5-4.

Example 2.5-5

Determine and plot the unit-step response of the system modeled by the equation

$$y''(t) + 100y'(t) + 5000y(t) = 5000x(t)$$

Assume zero initial conditions. Laplace transforming the equation and rearranging gives the output transform as

$$Y(s) = \frac{5000}{s(s^2 + 100s + 5000)} = \frac{5000}{s[(s+50)^2 + 50^2]}$$

Using partial fraction expansion gives

$$Y(s) = \frac{1}{s} - \frac{s+100}{(s+50)^2+50^2} = \frac{1}{s} - \frac{s+50}{(s+50)^2+50^2} - \frac{50}{(s+50)^2+50^2}$$

From Table 2.4-1, the unit-step response becomes

$$y(t) = [1 - e^{-50t}\cos 50t - e^{-50t}\sin 50t]u(t)$$

For this system, $\omega_n=70.71$ rad/sec, $\zeta=0.707$, $\tau=0.02$ sec, $T_s=0.08$ sec and $\omega_d=50$ rad/sec. Thus, the period of oscillations is $2\pi/\omega_d=0.126$ sec, which implies that steady state is reached before one cycle of oscillations is completed. The plot of the response is shown in Fig. 2.5-2.

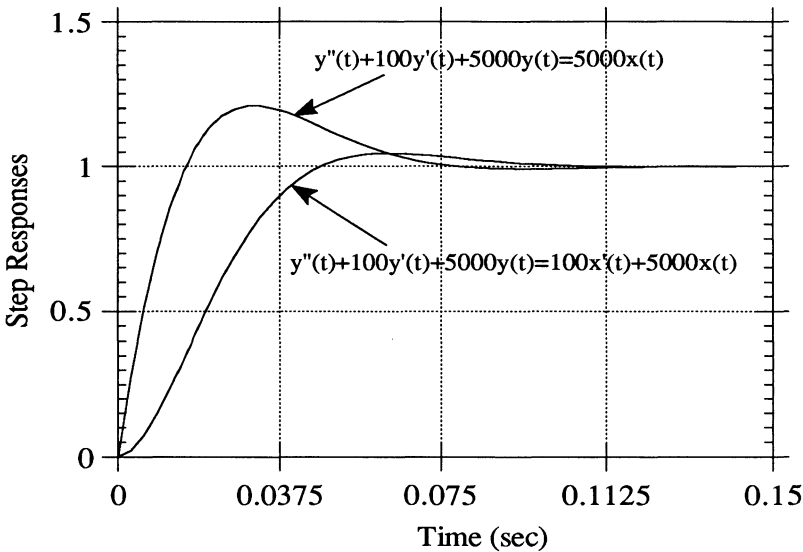


Fig. 2.5-2 Unit-step responses for the second-order systems used in Example 2.5-5 and Example 2.5-6.

Example 2.5-6

A system is described by the equation

$$y''(t) + 100y'(t) + 5000y(t) = 100x'(t) + 5000x(t)$$

Determine and plot the unit-step response of the system assuming zero initial conditions. For a unit-step input, the output transform is

$$Y(s) = \frac{100s+5000}{s(s^2+100s+5000)} = \frac{100s+5000}{s[(s+50)^2+50^2]} = \frac{1}{s} - \frac{s+50}{(s+50)^2+50^2} + \frac{50}{(s+50)^2+50^2}$$

The unit-step response becomes

$$y(t) = (1 - e^{-50t}\cos 50t + e^{-50t}\sin 50t)u(t)$$

The unit-step response is shown plotted in Fig. 2.5-2. Note the early peak in this response, which is due to the system zero.

Example 2.5-7

Determine and plot the unit-step response of the system represented by the equation

$$y'''(t) + 40y''(t) + 800y'(t) + 8000y(t) = 8000x(t)$$

assuming zero initial conditions. The output transform is

$$\begin{aligned} Y(s) &= \frac{8000}{s(s^3 + 40s^2 + 800s + 8000)} = \frac{8000}{s(s+20)[(s+10)^2 + 17.32^2]} \\ &= \frac{1}{s} - \frac{1}{s+20} - \frac{1.1547 \cdot 17.32}{(s+10)^2 + 17.32^2} \end{aligned}$$

The unit-step response is given by

$$y(t) = (1 - e^{-20t} - 1.1547e^{-10t} \sin 17.32t)u(t)$$

and is shown in Fig. 2.5-3. For this system, there are two time constants, $\tau_1 = 1/20 = 0.05$ sec and $\tau_2 = 1/\zeta\omega_n = 1/10 = 0.1$ sec. Thus, $T_s = 0.4$ sec and the period of oscillation is $2\pi/\omega_d = 0.363$ sec; in other words, it takes a little less than one cycle of oscillations before the step response reaches steady state.

2.6 TRANSFER FUNCTIONS AND BLOCK DIAGRAMS

The transfer function of a linear, time-invariant system is defined as the ratio of the output transform $Y(s)$ to the input transform $X(s)$, under the assumption that all initial conditions are zero. The transfer function, usually denoted by $G(s)$, is given by

$$G(s) = \frac{Y(s)}{X(s)} \quad (2.6-1)$$

Eq. (2.6-1) is a relationship between one output and one input as shown in the block diagram of Fig. 2.6-1.

The transfer function is a frequency domain model which, like its time domain counterpart - the differential equation, describes the relationship between a system output and a system input. The transfer function does not provide any information concerning the physical structure or internal make-up of the system it represents. For instance, an electrical RLC network may have the same transfer function as a mechanical spring-mass system, etc. However, unlike the differential equation which is used primarily for time-domain analysis, the transfer function is suitable for system analysis and design in both time and frequency domains.

A transfer function may be given or it may have to be derived from models including differential equations, block diagrams, or signal flow graphs.

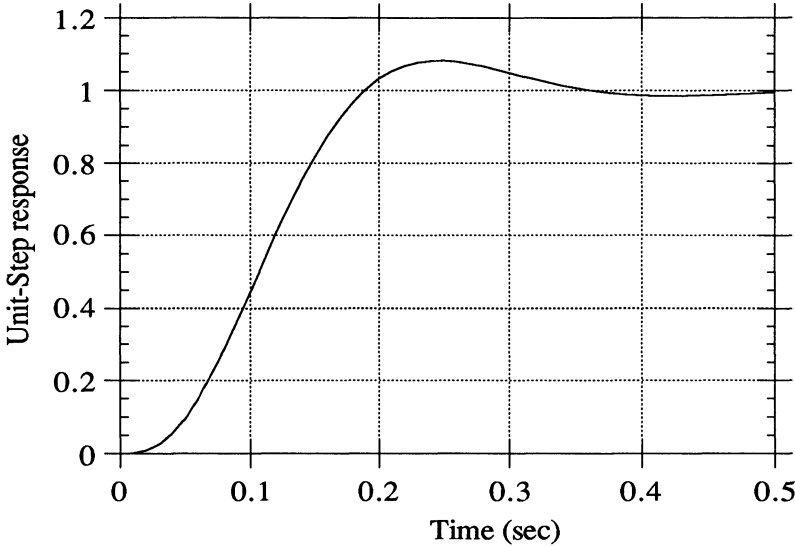


Fig. 2.5-3 Unit-step response of the 3rd-order system used in Example 2.5-7.

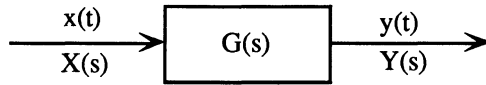


Fig. 2.6-1 Block diagram representation of a SISO system.

2.6-1 Transfer Functions and Differential Equations

Consider an n th-order, linear, time-invariant differential equation

$$\begin{aligned}
 y^{(n)}(t) + b_1 y^{(n-1)}(t) + \dots + b_{n-1} y'(t) + b_n y(t) \\
 = a_0 x^{(m)}(t) + a_1 x^{(m-1)}(t) + \dots + a_{m-1} x'(t) + a_m x(t)
 \end{aligned}
 \quad (2.6-2)$$

where $y(t)$ is the output and $x(t)$ is the input of the system.

The corresponding transfer function model of the system is obtained by Laplace transforming both sides of Eq. (2.6-2), assuming zero initial conditions. This gives

$$G(s) = \frac{Y(s)}{X(s)} = \frac{a_0 s^m + a_1 s^{m-1} + \dots + a_{m-1} s + a_m}{s^n + b_1 s^{n-1} + \dots + b_{n-1} s + b_n}
 \quad (2.6-3)$$

where $m \leq n$.

If the transfer function is known, the corresponding differential equation can be obtained by first cross-multiplying terms in Eq. (2.6-3). This gives

$$(s^n + b_1 s^{n-1} + \dots + b_{n-1} s + b_n)Y(s) = (a_0 s^m + a_1 s^{m-1} + \dots + a_{m-1} s + a_m)X(s) \quad (2.6-4)$$

Replacing 's' by the derivative operator d/dt , s^2 by d^2/dt^2 , etc., results in the differential equation given by Eq. (2.6-2).

From Eq. (2.6-3), the output transform of a system can be expressed as

$$Y(s) = G(s)X(s) \quad (2.6-5)$$

where $G(s)$ is the **transfer function**, also referred to as the **system function**. The time response of the system due to a known input $x(t)$ can then be determined from the above equation using the method discussed in Section 2.5. Note, for instance, that if the input is the unit impulse function, then $X(s)=1$ and the unit-impulse response of the system is

$$y(t) = \mathcal{L}^{-1}[G(s)] = g(t) \quad (2.6-6)$$

2.6-2 Transfer Functions of Electrical Networks

Transfer function models for the electrical, mechanical, and control systems considered in Section 2.2 can be determined by simply Laplace transforming their respective differential equation models with zero initial conditions.

It is more practical to work with transformed variables, rather than time variables, and manipulate the resulting algebraic equations. For the RC electrical network, for instance, the transformed diagram is shown in Fig. 2.6-2a, where the upper-case notation denotes transformed variables. The relevant equations are

$$V_1(s) = I(s)R + V_2(s) \quad (2.6-7)$$

and

$$I(s) = sCV_2(s) \quad (2.6-8)$$

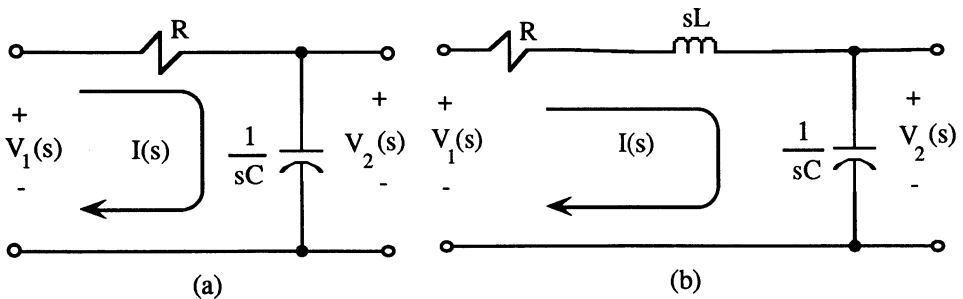


Fig. 2.6-2 Electrical network diagrams representing transformed variables.

Substituting $I(s)$ from the second equation into the first and rearranging leads to the transfer function model

$$\frac{V_2(s)}{V_1(s)} = \frac{1}{sCR+1} \quad (2.6-9)$$

which agrees with that in Eq. (2.2-2). For the RLC network shown in Fig 2.6-2b, the algebraic equations are

$$V_1(s) = I(s)(R + sL) + V_2(s) \quad (2.6-10)$$

$$\text{and} \quad I(s) = sCV_2(s) \quad (2.6-11)$$

Eliminating $I(s)$ leads to the network voltage transfer function

$$\frac{V_2(s)}{V_1(s)} = \frac{1}{s^2LC+sCR+1} \quad (2.6-12)$$

which is the same as Eq. (2.2-4) - obtained earlier.

As another example, consider the so-called **twin-T** or **notch** filter network shown in Fig. 2.6-3. This is a practical system used in many signal-conditioning applications. Its voltage transfer function can be determined by using either nodal analysis or mesh analysis. It can be shown that the transfer function relating $V_2(s)$ and $V_1(s)$ is

$$\frac{V_2(s)}{V_1(s)} = \frac{s^2C^2R^2+1}{s^2C^2R^2+s4CR+1} \quad (2.6-13)$$

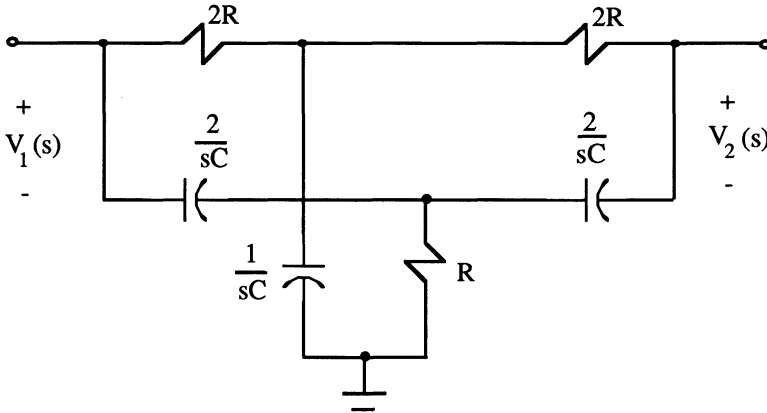


Fig. 2.6-3 Twin-T or notch filter network.

2.6-3 Transfer Functions of Simple Block Diagrams

In general, dynamic systems can be mathematically described by a set of simultaneous differential equations. Laplace transforming these equations results in a set of algebraic equations each of which can be solved for a different variable and, subsequently, represented in block diagram form.

A **block diagram model** of a system is a graphical or pictorial representation of the functions performed by each component or subsystem. The block diagram preserves the mathematical description of the system and uses arrows to indicate the direction of signal flow. A block diagram model can be developed, for example, for the RC network of Section 2.6-2. The procedure involves first solving Eq. (2.6-7) for $I(s)$, the variable to be eliminated. That is,

$$I(s) = \frac{[V_1(s) - V_2(s)]}{R} \quad (2.6-14)$$

Appropriate blocks are then connected in a diagram that satisfies the above equation, as shown in Fig. 2.6-4a. Next, substituting the intermediate variable $I(s)$ in Eq. (2.6-8), we express the output variable $V_2(s)$ as

$$V_2(s) = \frac{I(s)}{sC} \quad (2.6-15)$$

The completed block diagram is shown in Fig. 2.6-4b.

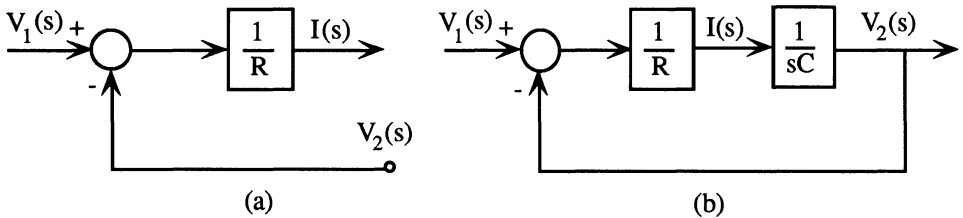


Fig. 2.6-4 Block diagram model of the RC network.

Following the same procedure, we can use Eq. (2.6-10) and Eq. (2.6-11) to develop block diagram models for an RLC network. One such model is shown in Fig. 2.6-5a. An equivalent model is shown in Fig. 2.6-5b.

If a block-diagram representation of a system is available, then appropriate mathematical models - transfer functions and differential equations - can be derived by block diagram reduction or manipulation techniques. Block diagrams can also be used for system simulation studies using a digital computer.

Consider next the **standard (canonical)** form block diagram model for a control system with negative feedback shown in Fig. 2.6-6. More complicated block diagrams can be reduced to this canonical form by the method discussed in the next section. The quantities shown in the block diagram of Fig. 2.6-6 are defined as follows:

$R(s)$ = Reference (command or setpoint) input

$C(s)$ = Controlled variable (output)

$E(s)$ = Error (actuating) signal

$G_c(s)$ = Controller (control law or control algorithm) transfer function

$G_p(s)$ = Plant (process) transfer function

$G(s) = G_c(s)G_p(s)$ = Forward path transfer function

$M(s)$ = Manipulation variable (controller output)

$D(s)$ = Disturbance (noise or load) input

$H(s)$ = Feedback transfer function

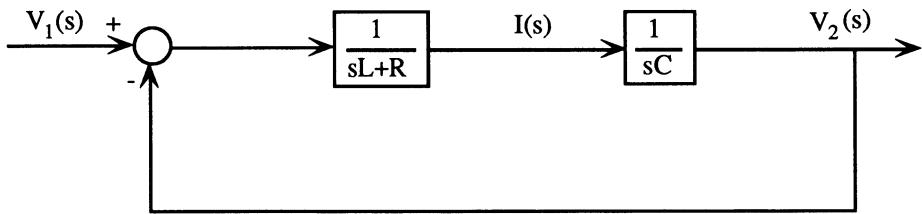
$B(s)$ = Primary feedback signal

$C(s)/R(s)$ = Closed-loop transfer function (control ratio)

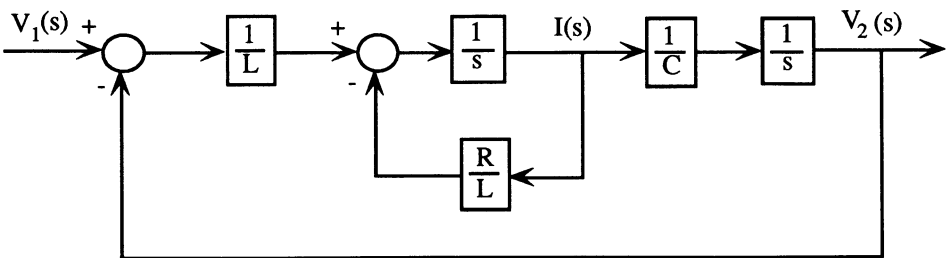
$B(s)/E(s) = G(s)H(s)$ = Open-loop (loop) transfer function

$E(s)/R(s)$ = Error (error ratio) transfer function

$C(s)/D(s)$ = Output-to-disturbance transfer function.



(a)



(b)

Fig. 2.6-5 Block diagram models of the RLC network.

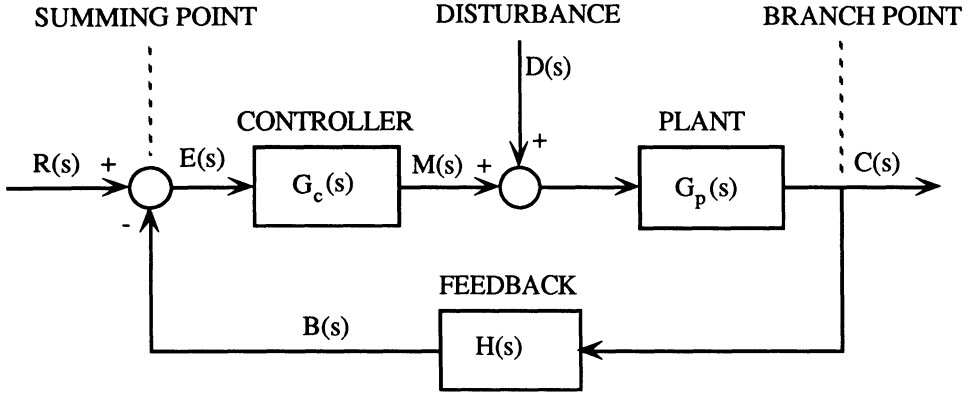


Fig. 2.6-6 Standard (canonical) form block diagram model for a control system.

The signal $B(s)$ is fed back to the summing point and compared with the reference input $R(s)$. The error detector produces a signal which is the difference between the reference and feedback signals.

In order for the output $C(s)$ to be compared with the reference $R(s)$ at the summing point, the units of the two signals must be the same. Any needed conversion or possible modification of the output signal is accomplished by the feedback element whose transfer function is denoted by $H(s)$.

The system shown in Fig. 2.6-6 is subjected to a disturbance and has two inputs, $R(s)$ and $D(s)$. To develop the relevant output-to-input transfer functions, we must consider each input separately with the system output $Y(s)$. Thus, assume first that the external disturbance input $D(s)$ is zero. Then, from Fig. 2.6-6,

$$E(s) = R(s) - B(s) = R(s) - H(s)C(s) \quad (2.6-16)$$

and
$$C(s) = E(s)G(s), \text{ or } E(s) = \frac{C(s)}{G(s)} \quad (2.6-17)$$

Equating the expressions for $E(s)$ and rearranging results in the expression for the closed-loop transfer function

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1+H(s)G(s)} \quad (2.6-18)$$

Since $C(s) = E(s)G(s)$, the error-to-input transfer function becomes

$$\frac{E(s)}{R(s)} = \frac{1}{1+H(s)G(s)} \quad (2.6-19)$$

If the system has positive instead of negative feedback, the plus sign in the denominators of Eq. (2.6-18) and Eq. (2.6-19) becomes minus.

The **characteristic equation** of a system in canonical form is given by

$$1 + H(s)G(s) = 0 \quad (2.6-20)$$

Assume next that the reference input $R(s)$ is zero. The output-to-disturbance transfer function is

$$\frac{C(s)}{D(s)} = \frac{G_p(s)}{1+H(s)G(s)} \quad (2.6-21)$$

which has the same denominator as the main system transfer function.

Example 2.6-1

Determine the corresponding transfer function models for the control systems shown in block diagram form in Fig. 2.2-4.

The systems can be re-drawn in block diagram form as shown in Fig. 2.6-7a and Fig. 2.6-7b. Using Eq. (2.6-18), the transfer function of the first-order system is

$$\frac{C(s)}{R(s)} = \frac{K}{s+K}$$

For the system of Fig. 2.6-7b, we apply Eq. (2.6-18) first to the inner loop and then to the outer loop. The resulting transfer function is

$$\frac{C(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

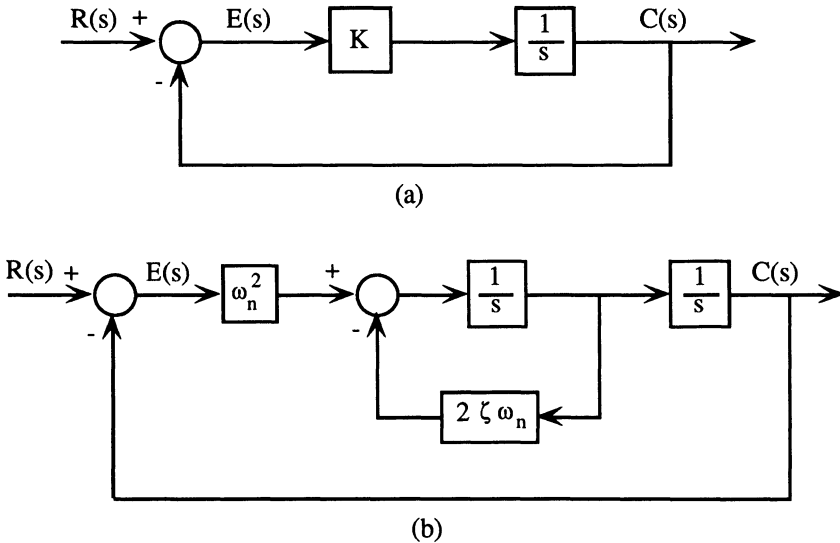


Fig. 2.6-7 First-order (a) and second-order (b) feedback control systems.

2.6-4 Block Diagram Reduction

When a complicated block diagram involving multiple feedback loops is encountered, it is often desirable to reduce it to standard form so that the relationships developed in Section 2.6-3 can be used. To accomplish this objective, fundamental rules of block diagram algebra are employed. Some of these rules are shown in Table 2.6-1. More comprehensive tables of such rules, if needed, can be found in the literature.

The first rule in the table concerns blocks, or components, connected in series (cascade). In such a configuration, the output of one block becomes the input to the next block. The cascaded blocks can be replaced by a single block whose transfer function is the product of the individual transfer functions. This implies that the flow of signals is only in the direction of the arrow and, thus, the output of any one block is not affected or "loaded" by the input of the following block. Thus, if $A(s)$ is the output of $G_1(s)$, and the input to $G_2(s)$, then

$$\frac{Y(s)}{X(s)} = \frac{A(s)}{X(s)} \frac{Y(s)}{A(s)} = G_1(s)G_2(s) \quad (2.6-22)$$

The second entry in Table 2.6-1 represents the rule for transfer function blocks connected in parallel. This rule states that if two or more blocks are connected in parallel, they can be replaced by a single block whose transfer function is equal to the algebraic sum of the individual transfer functions. That is,

$$\frac{Y(s)}{X(s)} = G_1(s) + G_2(s) \quad (2.6-23)$$

The third and fourth table entries define the rules for exchanging adjacent summing points and moving a summing point around a block, respectively. The fifth entry in the table depicts the rule for moving a branch point or node around a block. The validity of these rules can be readily established by writing equations for the corresponding output transforms and showing that they are equivalent.

Block diagram manipulation is illustrated next by means of examples. The analysis of practical systems represented by block diagrams is emphasized.

Example 2.6-2

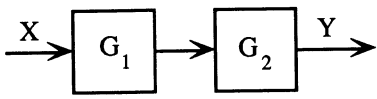
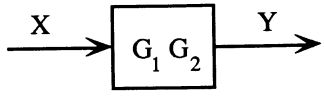
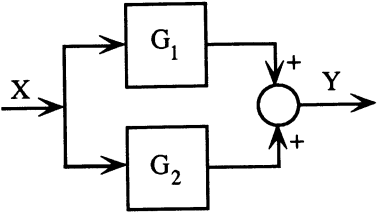
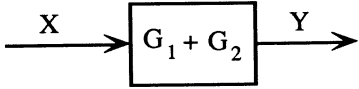
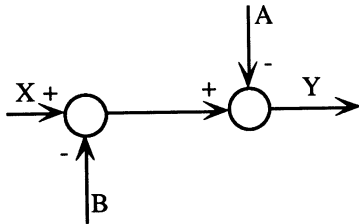
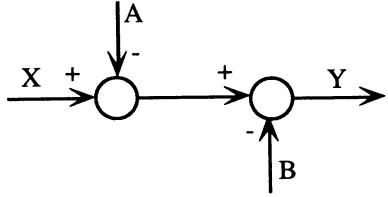
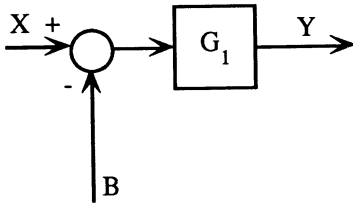
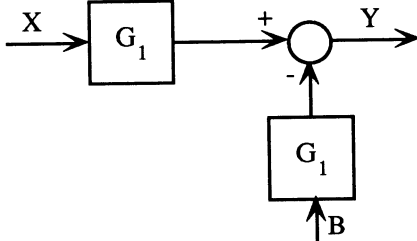
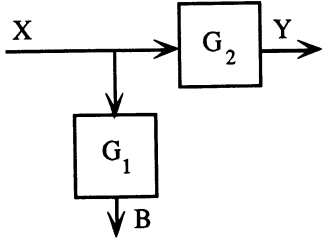
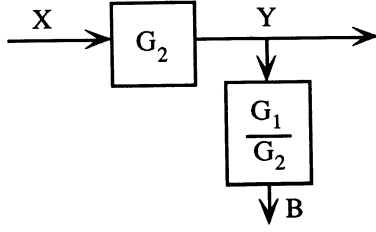
Reduce the block diagram of Fig. 2.6-8a to a single-loop standard/canonical form block diagram. Then, calculate the transfer function $C(s)/R(s)$ using Eq. (2.6-18).

First, move the branch point located between blocks G_2 and G_3 around block G_3 according to rule 5 of Table 2.6-1. The result is shown in Fig. 2.6-8b. Next, replace the inner feedback loop by its transfer function according to Eq. (2.6-18), and combine the blocks connected in series. This step yields the canonical form representation shown in Fig. 2.6-8c. Using Eq. (2.6-18), we obtain the closed-loop transfer function as

$$\frac{C(s)}{R(s)} = \frac{G_1(s)G_2(s)G_3(s)}{1 + H_1(s)G_2(s)G_3(s) + H_2(s)G_1(s)G_2(s)}$$

An alternative approach is to move the inner summing point to the left and the branch point to the right in Fig. 2.6-8a. The canonical form block diagram is then obtained by combining the resulting parallel feedback blocks. Carrying out these steps

Table 2.6-1. Brief Table of algebra rules for block diagram manipulation

Rule	Block Diagram	Equivalent Block Diagram
1		
2		
3		
4		
5		

shows that this canonical form is different from the one shown in Fig. 2.6-8c, but the corresponding $C(s)/R(s)$ transfer functions are the same. This verifies the "uniqueness" property of transfer functions and the non-uniqueness property of block diagrams.

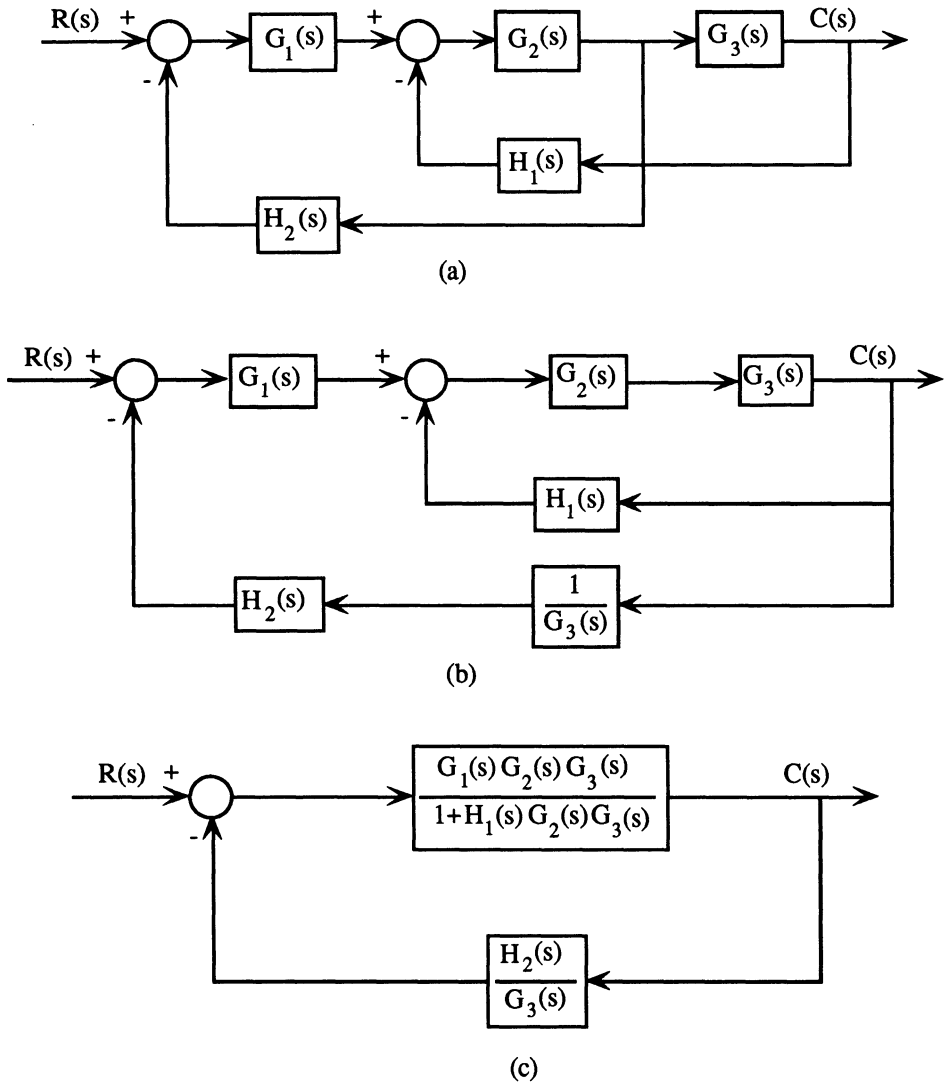


Fig. 2.6-8 Example of block diagram reduction.

Example 2.6-3

For the systems (a) and (b) shown in Fig. 2.6-9, determine the closed-loop transfer function $C(s)/R(s)$, the output-to-disturbance transfer function $C(s)/D(s)$, and the open-loop transfer function $H(s)G(s)$. Also, for each system, determine and plot the unit step response if $K=20$, $K_P=5$, and $K_I=250$.

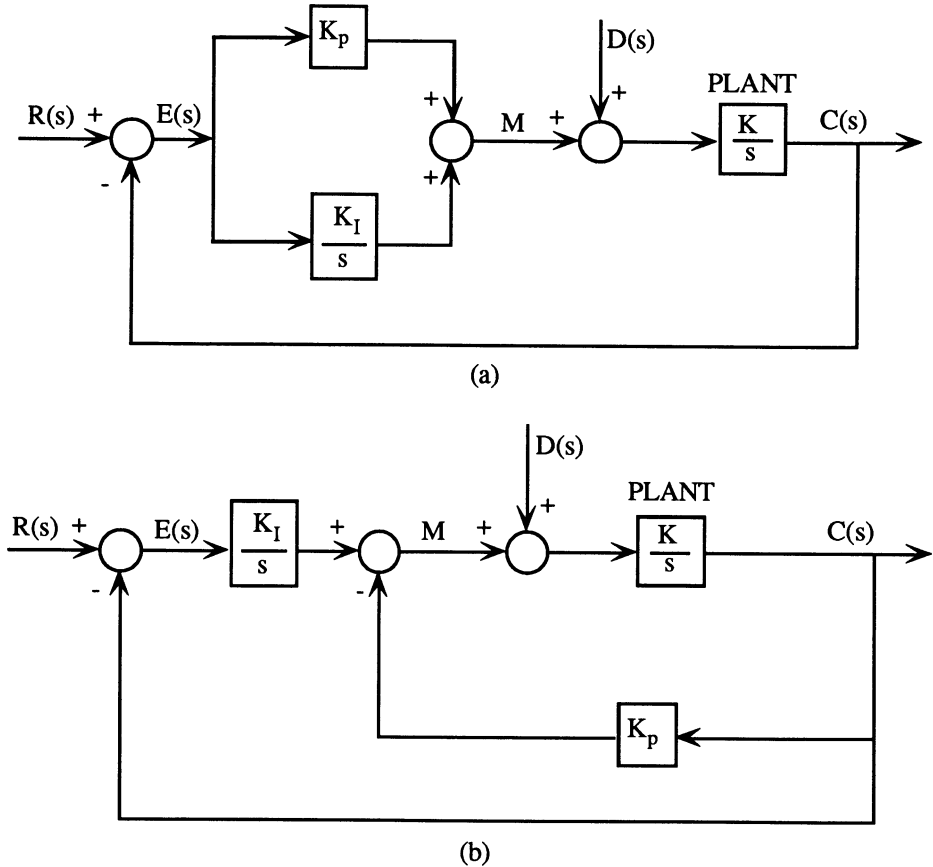


Fig. 2.6-9 Block diagram models for the PID (a) and PDF (b) control systems used in Example 2.6-3.

The plant to be controlled acts as an integrator with gain K , an arbitrary constant. The control law (controller) shown in Fig. 2.6-9a is a modified version of a popular controller known as proportional-integral-derivative, or just PID. The parameters K_P and K_I represent the proportional and integral controller gains, respectively. The derivative controller gain K_D is not shown in either block diagram, because it is not required in this example.

To distinguish the control system shown in Fig. 2.6-9b from that of Fig. 2.6-9a, we refer to the controller of system (b) as PDF, rather than PID, because the proportional (K_p) and, if present, the derivative (K_D) controller gains are located in the feedback path. However, the integral controller gain (K_I) is located in the forward path for both PID and PDF controllers. More about these and other control laws in Chapters 6 and 7.

For system (a):

The open-loop transfer function is

$$H(s)G(s) = \frac{K(K_p s + K_I)}{s^2}$$

Then, from Eq. (2.6-18), the closed-loop transfer function can be written as

$$\frac{C(s)}{R(s)} = \frac{KK_p s + KK_I}{s^2 + KK_p s + KK_I}$$

and from Eq. (2.6-21) , the output-to-disturbance transfer function is

$$\frac{C(s)}{D(s)} = \frac{Ks}{s^2 + KK_p s + KK_I}$$

For the given values $K=20$, $K_p=5$, and $K_I=250$, the transfer functions of the PID control system are

$$\frac{C(s)}{R(s)} = \frac{100s + 5000}{s^2 + 100s + 5000}$$

$$\frac{C(s)}{D(s)} = \frac{20s}{s^2 + 100s + 5000}$$

and

$$H(s)G(s) = \frac{100s + 5000}{s^2}$$

The step response for the system of Eq. (2.6-30) was determined in Example 2.5-6.

For system (b):

Assuming $D(s)=0$, the inner-loop transfer function is $K/(s+KK_p)$. Then, the open-loop transfer function becomes

$$H(s)G(s) = \frac{KK_I}{s(s+KK_p)}$$

From Eq. (2.6-18),

$$\frac{C(s)}{R(s)} = \frac{KK_I}{s^2 + KK_p s + KK_I}$$

With $R(s)=0$, the forward transfer function is that of the plant, that is, $G_p(s)= K/s$, and the feedback transfer function $H(s)$ is the sum of the two feedback transfer functions

$$H(s) = K_P + \frac{K_I}{s}$$

Thus, the open-loop transfer function becomes

$$H(s)G(s) = \frac{KK_Ps + KK_I}{s^2}$$

and, using Eq. (2.6-21),

$$\frac{C(s)}{D(s)} = \frac{Ks}{s^2 + KK_Ps + KK_I}$$

For the given parameter values, the transfer functions for the PDF system become

$$\frac{C(s)}{R(s)} = \frac{5000}{s^2 + 100s + 5000}$$

and

$$H(s)G(s) = \frac{5000}{s(s+100)}$$

The above $C(s)/R(s)$ transfer function suggests a standard-form second-order system with a damping ratio of 0.707 and a natural frequency of 70.71 rad/sec. Its corresponding unit-step response was determined in Example 2.5-5.

Note that both the PID and PDF control systems have the same denominators, but the response of the PID system exhibits a pronounced early peak due to the zero in the numerator. In general, the magnitude of the peak in the step response of such a system is inversely proportional to the distance of this zero from the origin ($s=0$) and the $j\omega$ -axis. In the PID system, the location of the zero is fixed automatically by the location of its denominator zeros and, thus, its controller gains cannot be adjusted independently in order to change the response of the system.

Observe, also, that both control systems have the same output-to-disturbance transfer function $C(s)/D(s)$. From the final value theorem, it can be easily seen that a step change in disturbance will result in a steady-state value of zero for the output $c(t)$. In other words, both systems will remove any disturbance that may enter the control loop in a stepwise fashion. The step response of $C(s)/D(s)$ shows this feature.

Example 2.6-4

Determine the transfer functions $C(s)/R(s)$, $C(s)/D(s)$, and $C(s)/E(s)$ for the system shown in Fig. 2.6-10. Also, determine and plot the unit step response of the system, assuming $K=500$, $K_P=1.6$, $K_I=16$, and $K_D=0.08$.

Note that the plant consists of a double integrator K/s^2 . Simplifying the inner loop, first, leads to the open-loop transfer function

$$\frac{C(s)}{E(s)} = H(s)G(s) = \frac{KK_I}{s(s^2 + KK_Ds + KK_P)}$$

The closed-loop transfer function then becomes

$$\frac{C(s)}{R(s)} = \frac{KK_I}{s^3 + KK_D s^2 + KK_P s + KK_I}$$

With $R(s)=0$, the output-to-disturbance transfer function is

$$\frac{C(s)}{D(s)} = \frac{Ks}{s^3 + KK_D s^2 + KK_P s + KK_I}$$

From the final value theorem, the last equation implies a final value of zero to step function inputs, which is the desirable characteristic. For the given parameter values, the transfer functions are

$$H(s)G(s) = \frac{8000}{s(s^2 + 40s + 800)}$$

$$\frac{C(s)}{R(s)} = \frac{8000}{s^3 + 40s^2 + 800s + 8000}$$

and

$$\frac{C(s)}{D(s)} = \frac{500s}{s^3 + 40s^2 + 800s + 8000}$$

The step response of this system was determined in Example 2.5-7.

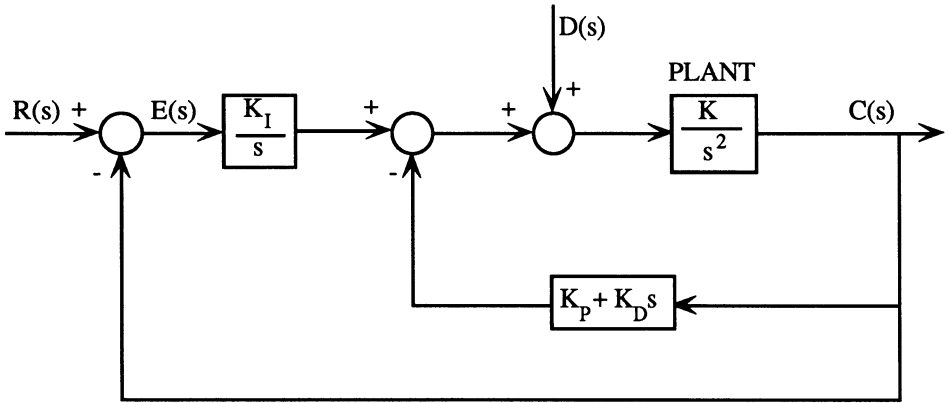


Fig. 2.6-10 Block diagram model of the control system used in Example 2.6-4.

2.6-5 Steady-State Error Analysis

In control system applications, the ability of a system to follow or track a given input with a minimum or zero steady-state error is significant. *The steady-state error is a measure of the difference between a system's input and its output, under steady state conditions.* In the case of control systems, the steady-state error quantity is said to measure the accuracy of the system and depends on the **type** of the open-loop transfer function $H(s)G(s)$. *The term "type" refers to the order of the pole of $H(s)G(s)$ at $s=0$.* For instance, if the open-loop transfer function is given by

$$H(s)G(s) = \frac{K(1+T_1s)(1+T_2s) \dots (1+T_ms)}{s^q(1+\tau_1s)(1+\tau_2s) \dots (1+\tau_ns)} \quad (2.6-24)$$

then the system is of type q , where $q=0, 1, 2, \dots$.

From the error ratio defined by Eq. (2.6-19), the error function transform becomes

$$E(s) = R(s) - C(s)H(s) = \frac{R(s)}{1+H(s)G(s)} \quad (2.6-25)$$

From the final value theorem, the steady-state error of the system is

$$E_{ss} = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} \frac{sR(s)}{1+H(s)G(s)} \quad (2.6-26)$$

If the input is a step function of amplitude R_o , that is, if $r(t)=R_o u(t)$, the steady-state error is

$$E_{ss}(\text{step}) = \lim_{s \rightarrow 0} \frac{R_o}{1+H(s)G(s)} = \frac{R_o}{1+C_p} \quad (2.6-27)$$

where

$$C_p = \lim_{s \rightarrow 0} H(s)G(s) \quad (2.6-28)$$

C_p is usually referred to as the **positional or displacement error constant**.

If the input is a ramp function of slope R_o , that is, if $r(t)=R_o t u(t)$, then the corresponding steady-state error is given by

$$E_{ss}(\text{ramp}) = \lim_{s \rightarrow 0} \frac{R_o}{s[1+H(s)G(s)]} = \frac{R_o}{\lim_{s \rightarrow 0} sH(s)G(s)} = \frac{R_o}{C_v} \quad (2.6-29)$$

where

$$C_v = \lim_{s \rightarrow 0} sH(s)G(s) \quad (2.6-30)$$

C_v is referred to as the **velocity error constant**.

Finally, if the input is the parabolic function $(R_o t^2/2)u(t)$, then the steady-state error becomes

$$E_{ss}(\text{parabolic}) = \lim_{s \rightarrow 0} \frac{R_o}{s^2[1+H(s)G(s)]} = \frac{R_o}{C_a} \quad (2.6-31)$$

where

$$C_a = \lim_{s \rightarrow 0} s^2H(s)G(s) \quad (2.6-32)$$

C_a is referred to as the **acceleration error constant**. The steady-state error concepts are illustrated by means of examples.

Example 2.6-5

Determine the error constants C_p , C_v , and C_a for the standard-form second-order system shown in Fig. 2.6-7b. Its open-loop transfer function is

$$H(s)G(s) = \frac{\omega_n^2}{s(s+2\zeta\omega_n)} = \frac{\omega_n}{2\zeta} \frac{1}{[s(1+\frac{s}{2\zeta\omega_n})]}$$

which indicates a type 1 system with K given by

$$K = \frac{\omega_n}{2\zeta}$$

The steady-state error coefficients defined above are

$$C_p = \infty$$

$$C_v = \frac{\omega_n}{2\zeta}$$

and

$$C_a = 0$$

The corresponding steady-state-error values then become

$$E_{ss}(\text{step}) = 0$$

$$E_{ss}(\text{ramp}) = \frac{2\zeta R_0}{\omega_n}$$

and

$$E_{ss}(\text{parabolic}) = \infty$$

Thus, this system can track step inputs with zero steady-state error, ramp inputs with finite steady-state error, but it cannot follow parabolic inputs.

Example 2.6-6

Determine the error constants C_p , C_v , and C_a as well as the respective steady-state errors for the PID and PDF control systems shown in Fig. 2.6-9.

The PID control system is a type 2 system which can follow step as well as ramp inputs with zero steady-state error. The PDF control system is a type 1 system which can follow step inputs with zero steady-state error and ramp inputs with finite steady-state error. Using the open-loop transfer function relationships developed in Example 2.6-3, we can obtain the error constants and steady-state-error values shown in Table 2.6-2.

Table 2.6-2 *Error constants and steady-state error values for PID and PDF control systems used in Example 2.6-6.*

System	C_p	C_v	C_a	$E_{ss}(\text{step})$	$E_{ss}(\text{ramp})$	$E_{ss}(\text{parabolic})$
PID	∞	∞	KK_I	0	0	$\frac{R_o}{KK_I}$
PDF	∞	$\frac{K_I}{K_P}$	0	0	$\frac{R_o K_P}{K_I}$	∞

For the case specified by the values $K=20$, $K_P=5$, and $K_I=250$, the above table implies that the PID system can track step and ramp inputs with zero steady-state error, and parabolic inputs with $0.0002R_o$ steady-state error. On the other hand, the PDF control system can follow step inputs with zero steady-state error, ramp inputs with $0.02R_o$ steady-state error, but it cannot follow parabolic inputs.

Example 2.6-7

Repeat Example 2.6-6 for the PDF control system shown in Fig. 2.6-4.

This is a type 1 system and, thus, it can follow step inputs with zero steady-state error. It can follow ramp inputs with finite steady-state error given by

$$E_{ss}(\text{ramp}) = \frac{R_o}{C_v} = \frac{R_o K_P}{K_I}$$

For the parameter values $K=500$, $K_P=1.6$, $K_I=16$, and $K_D=0.08$, the steady-state error is $10R_o$ percent, where R_o denotes the slope of the ramp. Note that the steady-state error for parabolic inputs is infinite.

2.7 TRANSFER FUNCTIONS AND SIGNAL FLOW GRAPHS

A **signal flow graph**, like a block diagram, is a graphical representation of a set of simultaneous algebraic equations. It consists of a diagram in which **nodes** represent system variables or signals that are connected by directed branches. A **branch** connected between two nodes acts as a signal multiplier, with the multiplication factor, also called **transmittance** or **transfer function**, indicated near the arrow of the branch. The arrow indicates the one-way direction of signal flow.

For a given system, a signal flow graph model contains essentially the same information as a block diagram and, like a block diagram, it is not unique. That is, different signal flow graph models can be drawn by rearranging the algebraic equations.

An important reason for using signal flow graphs is the availability of a gain formula, called **Mason's gain formula**, which provides the transfer function relationship between a pair of output-input system variables, without requiring any reduction or manipulation of the signal flow graph.

2.7-1 Signal Flow Graph Definitions

To illustrate the basic concepts of signal flow graphs, consider the block diagram and its equivalent signal flow graph shown in Fig. 2.7-1. The four variables or signals are $R(s)$, $E(s)$, $M(s)$, and $C(s)$. These variables are shown as nodes (small circles) in the signal flow graph. The value of each variable is equal to the sum of all variables entering the node on incoming branches, and this value is transmitted to all outgoing branches.

The branches show the relationship between the variables and include the appropriate transmittances or transfer functions. Notice that the negative summation is designated by associating the negative sign with the corresponding transfer function, in this case $H(s)$. The following definitions are relevant to the discussion of signal flow graphs:

Input Nodes (Sources). *Input nodes represent independent variables and have only outgoing branches.* In Fig. 2.7-1b, node $R(s)$ is a source or input node.

Output Nodes (Sinks). *Output nodes represent dependent variables and have only incoming branches.* In Fig. 2.7-1b, the last node $C(s)$ is an output node.

Mixed Nodes. *Mixed nodes have both incoming and outgoing branches.* In Fig. 2.7-1b, the $E(s)$ node and the left-most $C(s)$ node are mixed nodes. A mixed node may be treated as an output node by adding an outgoing branch of unity transmittance, as has been done with node $C(s)$ in Fig. 2.7-1b. Such a sink (output) node is sometimes called a dummy node. In general, any noninput node of a signal flow graph can be made an output node by this operation.

Path. *A path is a continuous sequence of branches traversed in the same direction.* The simple signal flow graph of Fig. 2.7-1b has many paths.

Forward Path. *A forward path is a path between an input node and an output node which does not meet or cross any node more than once.* In Fig. 2.7-1b, the path $R(s)E(s)M(s)C(s)C(s)$ is a forward path between the nodes $R(s)$ and $C(s)$.

Path Gain. *A path gain is the product of the branch transmittances of the path.* In Fig. 2.7-1b, the gain of the forward path is $(1)G_c(s)G_p(s)(1)=G_c(s)G_p(s)$.

Loop. *A loop is a closed path which originates and terminates on the same node and along the path no node is encountered more than once.* In Fig. 2.7-1b, $E(s)M(s)C(s)E(s)$ is a loop, in this case a feedback loop.

Loop Gain. *The loop gain is the product of the branch transmittances of a loop.* In Fig. 2.7-1b, the gain of the feedback loop is $-H(s)G_c(s)G_p(s)$.

Nontouching Loops. *Two loops are said to be nontouching if they have no nodes in common.*

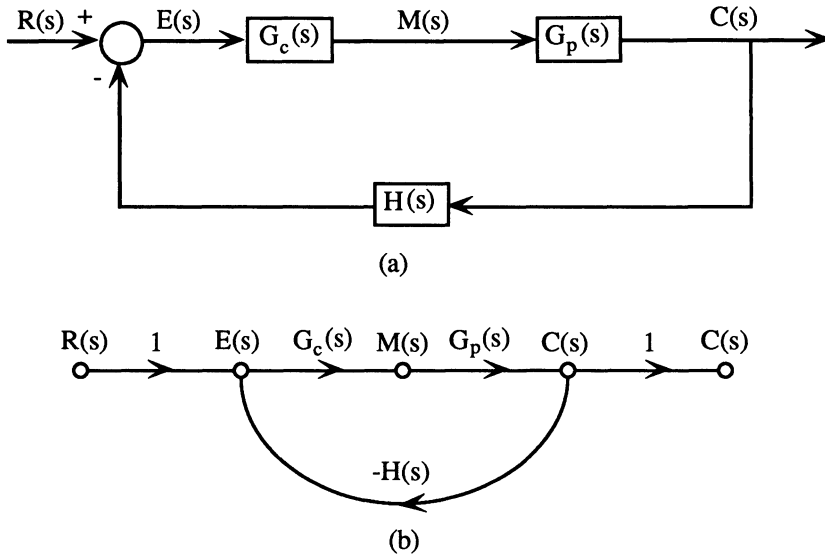


Fig. 2.7-1 Block diagram (a) and equivalent signal flow graph (b) models.

2.7-2 Mason's Gain Formula

In many practical applications, we are often interested in determining the transfer function description of a system for which a signal flow graph model is available or relatively easy to derive. Solving the problem analytically is usually a tedious task. Fortunately, the transfer function between an input variable $X(s)$ and an output variable $Y(s)$ is readily obtained using **Mason's gain formula**. The formula is given by

$$\frac{Y(s)}{X(s)} = \frac{\sum_i P_i(s) \Delta_i(s)}{\Delta(s)} \quad (2.7-1)$$

where

$P_i(s)$ = Path gain of i th forward path

$\Delta(s)$ = Graph determinant

= 1 - (sum of all different loop gains)

+ (sum of gain products of all combinations of two nontouching loops)

- (sum of gain products of all combinations of three nontouching loops)

+ ...

$\Delta_i(s)$ = Cofactor of the i th forward path determinant with loops touching the i th path removed.

For our case, the summations are taken over all possible paths between the input $X(s)$ and output $Y(s)$. Also, the graph determinant is the system characteristic polynomial; equivalently, the characteristic equation of the system is obtained by setting $\Delta(s)$ equal to zero. Mason's gain formula is illustrated by means of examples.

Example 2.7-1

Determine the closed-loop transfer function $C(s)/R(s)$ for the signal flow graph of Fig. 2.7-1b using Mason's gain formula.

The only forward path between the input $R(s)$ and output $C(s)$ is

$$P_1(s) = (1)G_c(s)G_p(s)(1) = G_c(s)G_p(s)$$

There is only one loop, call it L_1 . Then,

$$L_1(s) = -H(s)G_c(s)G_p(s)$$

The system (graph) determinant is

$$\Delta(s) = 1 - L_1(s) = 1 + H(s)G_c(s)G_p(s)$$

The only loop L_1 touches path P_1 and thus

$$\Delta_1(s) = 1 - 0 = 1$$

From Eq. (2.7-1) or Mason's formula, we obtain the transfer function as

$$\frac{C(s)}{R(s)} = \frac{P_1(s)\Delta_1(s)}{\Delta(s)} = \frac{G_c(s)G_p(s)}{1+H(s)G_c(s)G_p(s)}$$

Example 2.7-2

Draw the equivalent signal flow graph for the block diagram of Fig. 2.6-8a. Then, determine the transfer function $C(s)/R(s)$ using Mason's gain formula.

The equivalent signal flow graph model is shown in Fig. 2.7-2. The only forward path is

$$P_1(s) = (1)G_1(s)G_2(s)G_3(s)(1) = G_1(s)G_2(s)G_3(s)$$

There are two loops:

$$L_1(s) = -H_2(s)G_1(s)G_2(s), \text{ and } L_2(s) = -H_1(s)G_2(s)G_3(s)$$

There are no nontouching loops, because L_1 and L_2 have common nodes. Thus, the determinant is

$$\Delta(s) = 1 - [L_1(s) + L_2(s)] = 1 + H_2(s)G_1(s)G_2(s) + H_1(s)G_2(s)G_3(s)$$

Since P_1 touches loops L_1 and L_2 , the cofactor determinant is $\Delta(s) = 1$. From Mason's gain formula, then

$$\frac{C(s)}{R(s)} = \frac{P_1(s)\Delta_1(s)}{\Delta(s)} = \frac{G_1(s)G_2(s)G_3(s)}{1+H_2(s)G_1(s)G_2(s)+H_1(s)G_2(s)G_3(s)}$$

which agrees with Eq. (2.6-43), obtained using block-diagram reduction.

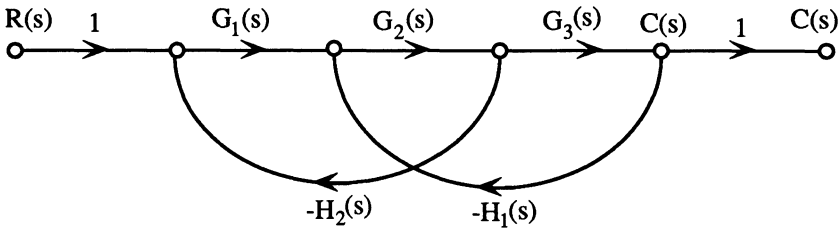


Fig. 2.7-2 Signal flow graph representation of the system shown in Fig. 2.6-8a.

Example 2.7-3

Draw the equivalent signal flow graph and then use Mason's gain formula to determine the transfer functions $C(s)/R(s)$ and $C(s)/D(s)$ for the control system shown in Fig. 2.7-3a. Also, determine and plot the unit-step responses of the control system specified by the values $K=20$, $K_P=5$, $K_I=250$ and for the cases $K_F=K_P$, $K_F=0.707K_P$, and $K_F=0.547K_P$.

The control scheme of Fig. 2.7-3a differs from the PDF control scheme of Fig. 2.6-9b by the so-called feedforward control gain K_F . We refer to this type of control scheme as PDFF, where the second F denotes the feedforward gain K_F . The equivalent signal flow graph is shown in Fig. 2.7-3b. From the signal flow graph,

$$P_1(s) = (1)(K_I)(s^{-1})(1)(K)(s^{-1})(1) = \frac{KK_I}{s^2}$$

$$P_2(s) = (K_F)(1)(K)(s^{-1})(1) = \frac{KK_F}{s}$$

$$L_1(s) = (K_I)(s^{-1})(1)(K)(s^{-1})(-1) = -\frac{KK_I}{s^2}$$

and

$$L_2(s) = (1)(K)(s^{-1})(-K_P) = -\frac{KK_P}{s}$$

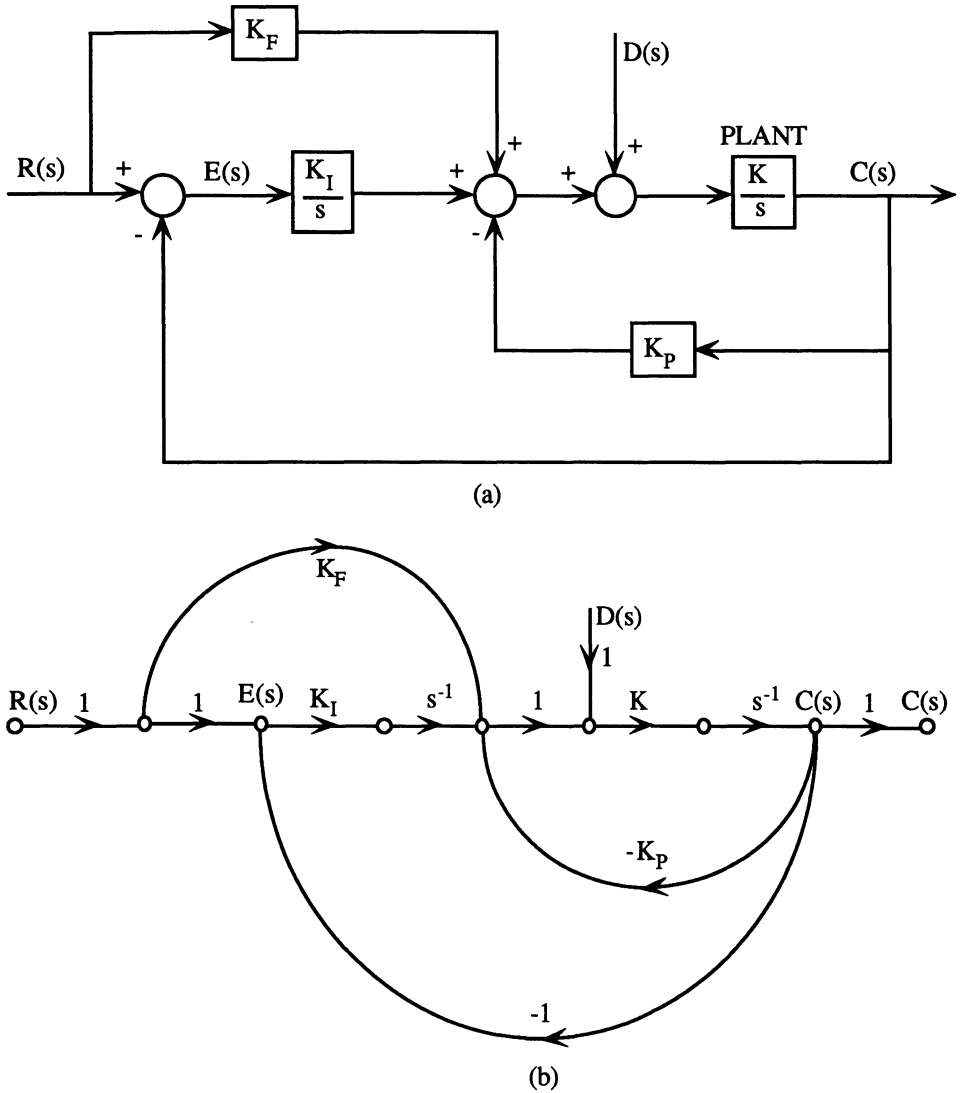


Fig. 2.7-3 (a) Block diagram and (b) signal flow graph models of the PDF control system used in Example 2.7-3.

Therefore, the system determinant and cofactors are

$$\Delta(s) = 1 - [L_1(s) + L_2(s)] = 1 + \frac{KK_P s + KK_I}{s^2}$$

and

$$\Delta_1(s) = \Delta_2(s) = 1$$

The closed-loop transfer function then becomes

$$\frac{C(s)}{R(s)} = \frac{P_1(s)\Delta_1(s) + P_2(s)\Delta_2(s)}{\Delta(s)} = \frac{KK_F s + KK_I}{s^2 + KK_P s + KK_I} \quad (2.7-2)$$

It is important to note that the K_F term has added a zero to the transfer function of the standard PDF control scheme of Example 2.6-3. The zero is at

$$s = -\frac{K_I}{K_F} \quad (2.7-3)$$

This zero can therefore be placed anywhere, independently of the denominator zeros to achieve desirable response characteristics. For instance, if $K_F = K_P$, the closed-loop transfer function of the PDF control system becomes identical to the transfer function of the PID control system used in Example 2.6-3. If the zero of the PDF control system is moved further to the left by simply increasing the gain K_F , the early peak in the step response of the system can be decreased.

In order to find the transfer function $C(s)/D(s)$, we must find all paths from $D(s)$ to $C(s)$. In the present diagram, there is only one path from the input $D(s)$ to the output $C(s)$. Hence, the path gain is

$$P'_1(s) = (1)(K)(s^{-1})(1) = \frac{K}{s}$$

Since $\Delta'_1(s) = 1$ and the system determinant $\Delta(s)$ remains the same,

$$\frac{C(s)}{D(s)} = \frac{P'_1(s)\Delta'_1(s)}{\Delta(s)} = \frac{Ks}{s^2 + KK_P s + KK_I} \quad (2.7-4)$$

This transfer function is the same as that obtained for the PID and PDF control schemes.

For the parameter values $K=20$, $K_P=5$, and $K_I=250$, the characteristic equation results in an underdamped system with damping ratio of 0.707 and natural frequency of 70.71 rad/sec. If K_F is set equal to K_P , $0.707K_P$ and $0.547K_P$, the corresponding zero locations are at -50 rad/sec (PID response), -70.71 rad/sec (zero at natural frequency), and -91.42 rad/sec, respectively.

The unit-step response of the PDF system is determined from the output transform $C(s)$. Using partial fractions

$$C(s) = \frac{20K_F s + 5000}{s(s^2 + 100s + 5000)} = \frac{1}{s} - \frac{(s+50) + (1-0.4K_F)50}{(s+50)^2 + 50^2} \quad (2.7-5)$$

From Laplace transform tables

$$c(t) = [1 - e^{-50t}\cos 50t - (1-0.4K_F)e^{-50t}\sin 50t]u(t) \quad (2.7-6)$$

The unit-step responses evaluated for different values of K_F (i.e., 5, 3.535, and 2.735) is shown in Fig 2.7-4.

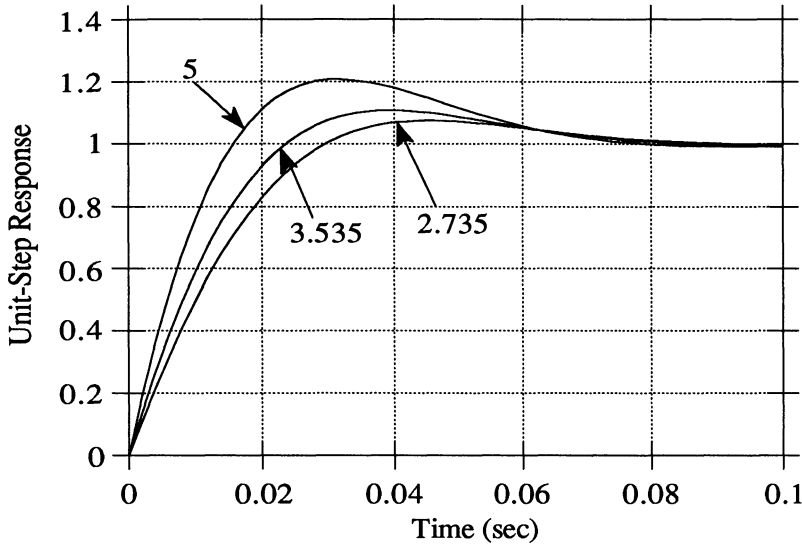


Fig. 2.7-4 Unit-step response of the system in Example 2.7-3 for various K_F values.

Example 2.7-4

Determine the steady-state error performance of the PDFF control system of Example 2.7-3.

The proportional term K_F does not change the type of the control system. This is a type 1 system and type 1 systems follow step inputs with zero steady-state error. For ramp inputs, we can apply the final value theorem to estimate the steady-state error. Since

$$E(s) = C(s) - R(s) = \frac{R(s)(s^2 + KK_Ps - KK_Fs)}{s^2 + KK_Ps + KK_I} \quad (2.7-7)$$

the steady-state error to ramp inputs is

$$E_{ss}(\text{ramp}) = \frac{R_o(K_P - K_F)}{K_I} \quad (2.7-8)$$

Therefore, for a PDFF controlled integrator plant, increasing the gain K_F decreases the steady-state error for ramp function inputs.

Following the previous procedure, one can show that the PDFF control of the double integrator plant of Example 2.6-4 exhibits performance characteristics similar to those of the PDFF control of the single integrator plant of Example 2.6-3. That is, the K_F term adds the zero given by Eq. (2.7-3) and the steady-state error to ramp inputs is given by Eq. (2.7-8). The controller gains for the two systems, however, are usually different.

2.8 CONTINUOUS SYSTEMS: POLES, ZEROS, AND STABILITY

The transfer function model of a typical n th-order system given by Eq. (2.6-3) can be expressed in factored form as

$$G(s) = \frac{Y(s)}{X(s)} = \frac{a_0(s-z_1)(s-z_2) \dots (s-z_m)}{(s-p_1)(s-p_2) \dots (s-p_n)} \quad (2.8-1)$$

where the numerator $Y(s)$ and the denominator $X(s)$ represent the transforms of the output $y(t)$ and input $x(t)$, respectively.

Poles and Zeros. The m roots of the numerator, $z_1, z_2, \dots, z_m$, are called the **zeros**, and the n roots of the denominator, $p_1, p_2, \dots, p_n$, are called the **poles** of the system or of its transfer function $G(s)$. Both zeros and poles may be real or complex.

It can be seen from Eq. (2.8-1) that a system can be completely specified by its poles, zeros, and the multiplying constant a_0 . Comparing the transfer function of Eq. (2.8-1) with the differential equation of Eq. (2.6-2), we see that the poles of the transfer function are the same as the characteristic roots or eigenvalues obtained from the differential equation. Consequently, the characteristic equation of a system can be obtained by simply setting the denominator of $G(s)$ equal to zero.

Stability. When classifying the performance of a system, one of the most important characteristics is its stability status. In the case of system analysis, for instance, first and foremost we want to know whether the system is stable or not. In the case of system design, on the other hand, we generally take it for granted that the system must be stable.

Even though there are several definitions of stability, the bounded-input-bounded-output (BIBO) definition is commonly used. According to this criterion

*a linear, time-invariant system is said to be bounded-input-bounded-output (BIBO) stable, or just **stable**, if every bounded input always produces a bounded output. A system that is not BIBO stable is said to be **unstable**.*

The stability of a system can be determined from the location of its characteristic roots (poles) in the complex s -plane. A system is BIBO stable, if all of its poles are located in the left-half s -plane. If at least one of the poles is not in the left-half s -plane, the system is unstable. For example, the first-order transfer function

$$G(s) = \frac{1}{s+a}, \quad a>0 \quad (2.8-2)$$

is stable, because its characteristic root or pole, $s=-a$, is located in the left-half s -plane. Thus, any bounded input, such as a step function, will produce a bounded output.

The second-order system described by the transfer function

$$G(s) = \frac{1}{s-b}, \quad b>0 \quad (2.8-3)$$

is an unstable system, since its only pole, $s=b$, is located in the right-half s -plane and its step response, for instance, will be unbounded.

A special situation arises when a system has no right-half s -plane poles but has simple poles, real or complex, on the $j\omega$ -axis. Such systems are often referred to as

marginally stable or **critically stable**, even though they are BIBO unstable. For instance, the single-integrator transfer function

$$G(s) = \frac{1}{s} \quad (2.8-4)$$

has a simple, real pole at the origin ($s=0$). This $G(s)$ represents an unstable system, because a step function input, which is bounded, produces a ramp function output, which is unbounded.

As another example, consider the transfer function

$$G(s) = \frac{1}{s^2+1} \quad (2.8-5)$$

It has simple, complex poles, $s=\pm j$, on the $j\omega$ -axis. Even though a step-function input, which is bounded, produces a bounded output, the input $\sin t$, which is also bounded, however, produces an unbounded output. Thus, the system is unstable.

The stability of a system can also be determined from its corresponding impulse response. For the general case represented by Eq. (2.8-1), the input is $x(t)$, the output is $y(t)$, and the impulse response is $g(t)$. Then, using the convolution integral relationship represented by Eq. (2.4-18), we can express the system output as

$$y(t) = \int_0^{\infty} x(t-\tau)g(\tau)d\tau \quad (2.8-6)$$

Taking the absolute value on both sides of the equation, we obtain

$$|y(t)| = \left| \int_0^{\infty} x(t-\tau)g(\tau)d\tau \right| \quad (2.8-7)$$

The above equation can be written as

$$|y(t)| \leq \int_0^{\infty} |x(t-\tau)||g(\tau)|d\tau \quad (2.8-8)$$

If the magnitude of the input $x(t)$ is bounded, that is, if

$$|x(t)| \leq P < \infty \quad (2.8-9)$$

where P is a finite, positive number, then

$$|y(t)| \leq P \int_0^{\infty} |g(\tau)|d\tau \quad (2.8-10)$$

If the output $y(t)$ is to be bounded, that is, if

$$|y(t)| \leq Q < \infty \quad (2.8-11)$$

where Q is a finite, positive number, it follows that

$$\int_0^{\infty} |\lg(\tau)| d\tau \leq R < \infty \quad (2.8-12)$$

for any finite, positive number R . This leads to a related definition of BIBO stability. That is,

for a system to be BIBO stable, or just stable, the area under its impulse response curve must be finite.

If the impulse response function itself is considered instead of the area under the impulse response, an equivalent stability criterion is the following:

for a BIBO stable system, the impulse response goes to zero as t approaches infinity. That is,

$$\lim_{t \rightarrow \infty} g(t) = 0 \quad (2.8-13)$$

Conversely, if

$$\lim_{t \rightarrow \infty} g(t) = \infty \quad (2.8-14)$$

then the system is unstable. Finally, if

$$\lim_{t \rightarrow \infty} g(t) = P \quad (2.8-15)$$

where P is a finite, nonzero value or a bounded oscillation, then the system is marginally stable.

The impulse response of $G(s)=1/(s+a)$, e^{-at} , satisfies Eq. (2.8-13) for a stable system, while the impulse response of $G(s)=1/(s-b)$, e^{bt} , satisfies Eq. (2.8-14) for an unstable system. Furthermore, the impulse response of the integrator $G(s)=1/s$ is equal to 1, while the impulse response of the system of $G(s)=1/(s^2+1)$, $\sin t$, results in a bounded oscillation. Thus, the last two systems are marginally stable, or BIBO unstable, because their impulse responses satisfy Eq. (2.8-15). For BIBO stability, it does not matter whether the input is the conventional function $x(t)$ depicted in Eq. (2.8-1), or an external disturbance.

If all inputs are zero, the system response due to initial conditions (internal energy) only is referred to as **zero-input response**. Thus,

*a linear, time-invariant system is said to be **zero-input stable**, if its zero-input response goes to zero as t approaches infinity; otherwise, the system is unstable*

is another definition of stability. Zero-input stability, also known as **asymptotic stability**, implies that the roots of the characteristic equation must be located in the left-half s -plane. Since this is also a requirement for BIBO stability, it follows that a BIBO stable system is also zero-input or asymptotically stable.

The previous definitions of stability address **absolute stability**. Absolute stability answers the fundamental question whether the system is stable or unstable. For a stable system, however, it is often necessary to indicate how stable it is or, equivalently, how close it is to becoming unstable. This is called **relative stability**.

One measure of relative stability is the location of the system poles in relation to the $j\omega$ -axis. For instance, the poles for a standard-form, second-order system are given by

$$s_{1,2} = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} \quad (2.8-16)$$

where ζ and ω_n represent the damping ratio and the (undamped) natural frequency, respectively. Pole locations, denoted by X, for some typical cases are shown in Fig. 2.8-1. For the underdamped case $\zeta < 1$, the distance of the complex poles from the origin is the natural frequency ω_n . Also, from the figure,

$$\zeta = \cos\theta \quad (2.8-17)$$

where θ is the angle the vector ω_n makes with the negative real axis. Thus, as θ increases (ζ decreases), the system poles move closer to the $j\omega$ -axis and the system becomes less stable. When the poles become purely imaginary ($\zeta=0$) or cross over to the right-half s -plane ($\zeta < 0$), the system becomes unstable.

Using typical values of ζ , the unit-step responses for the second-order system are shown in Fig. 2.8-2. Notice that for $\zeta=0$, the response exhibits sustained oscillations and that as the damping increases, the oscillations diminish. When $\zeta=1$, the damping is critical and the oscillations and overshoot disappear. For $\zeta > 1$, the response becomes overdamped and relatively "sluggish" or slow-responding, and for $\zeta < 0$, the response is unstable.

In general, most practical systems do not have poles or zeros in the right-half s -plane. A system that has no poles or zeros in the right-half s -plane is called a **minimum-phase** system. If there are poles and/or zeros in the right-half s -plane, the system is called **nonminimum-phase**. Most of the systems considered in this book are of the minimum-phase type.

The poles or zeros of a system can be determined by factoring the corresponding polynomial in ' s '. For higher than second-order systems, this task is routinely carried out on the digital computer. If the main concern is to simply answer the question whether a system is stable or unstable, an attractive alternative to finding the characteristic roots is to use Routh's criterion.

2.8-1 Routh's Stability Criterion

Routh's stability criterion is a simple method for determining whether or not a polynomial equation has roots with positive real parts, without actually having to solve for them. In the case of a characteristic equation, the criterion enables us to determine system stability, because characteristic roots with positive real parts indicate instability.

Consider the familiar characteristic equation

$$s^n + b_1s^{n-1} + \dots + b_{n-1}s + b_n = 0 \quad (2.8-18)$$

where the b 's are real coefficients. Assume that any zero root has been removed from the equation, that is, b_n is not zero. In addition, assume all coefficients are present and of the same sign. Otherwise, there are pure imaginary roots, or roots with positive real parts and the system is unstable.

Next, let the elements of Routh's table or array be denoted by the two-dimensional variable $T_{i,j}$, where the subscripts i and j represent rows and columns, respectively. Then, for a 5th-order characteristic equation, Routh's table is arranged as shown in Table 8.8-1.

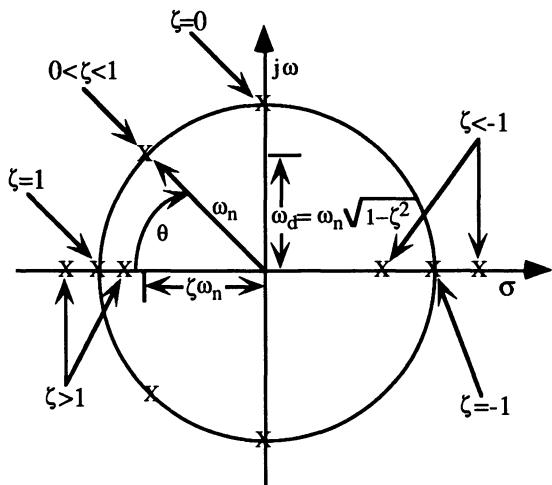


Fig. 2.8-1 Characteristic root (pole) locations, marked by X, and corresponding damping ratio (ζ) values for the standard-form, second-order system.

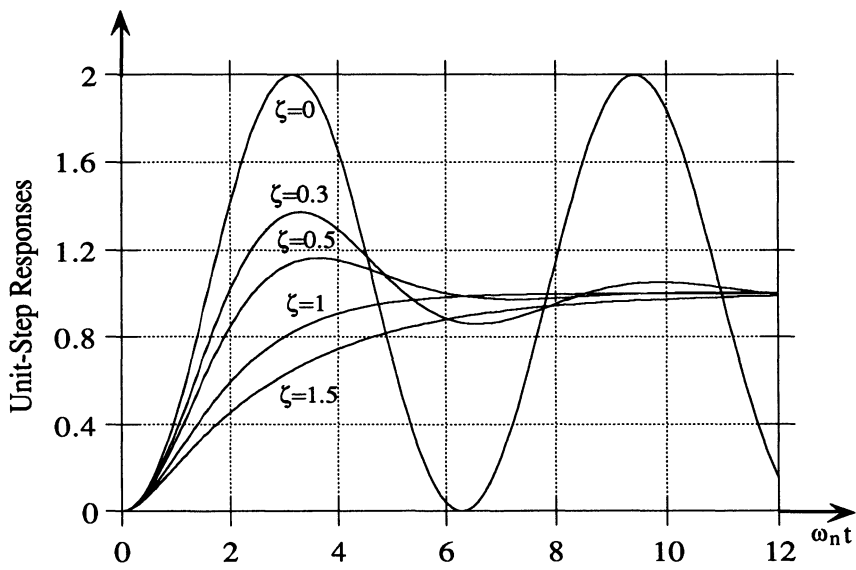


Fig. 2.8-2 Unit-step responses of a second-order system using typical values of ζ .

Table 2.8-1. Routh's Table of Coefficients

s^5	1	b_2	b_4
s^4	b_1	b_3	b_5
s^3	$T_{3,1}$	$T_{3,2}$	
s^2	$T_{4,1}$	$T_{4,2}$	
s^1	$T_{5,1}$		
s^0	$T_{6,1}$		

The first two rows of the table are formed from the coefficients of the characteristic equation, and the entries left blank represent zero elements. The rest of the entries are evaluated from the known coefficients using the relationships:

$$T_{3,1} = \frac{b_1 b_2 - b_3}{b_1} \quad (2.8-19)$$

$$T_{3,2} = \frac{b_1 b_4 - b_5}{b_1} \quad (2.8-20)$$

$$T_{4,1} = \frac{b_3 T_{3,1} - b_1 T_{3,2}}{T_{3,1}} \quad (2.8-21)$$

$$T_{4,2} = \frac{b_5 T_{3,1} - 0 b_1}{T_{3,1}} = b_5 \quad (2.8-22)$$

$$T_{5,1} = \frac{T_{4,1} T_{3,2} - T_{3,1} T_{4,2}}{T_{4,1}} \quad (2.8-23)$$

and

$$T_{6,1} = \frac{T_{4,2} T_{5,1} - 0 T_{4,1}}{T_{5,1}} = T_{4,2} \quad (2.8-24)$$

For the general case, the array T has $(n+1)$ rows and k columns, where the integer value of k depends on n as follows. If n is odd, then

$$k = \frac{n+1}{2} \quad (2.8-25)$$

and if n is even,

$$k = \frac{n+2}{2} \quad (2.8-26)$$

Routh's table can then be generated from the coefficients of the characteristic equation using the BASIC program segment given below.

```

100 IF (-1)^N < 0 THEN K=(N+1)/2 ELSE K=(N+2)/2
110 FOR I=1 TO 2
120   L=I-1
130   FOR J=1 TO K
140     IF I=1 AND J=1 THEN 160
150     T(I,J)=B(L)
160     L=L+2
170   NEXT J
180 NEXT I
190 T(1,1)=1
200 FOR I=3 TO N+1
210   FOR J=1 TO K
220     T(I,J)=(T(I-1,1)*T(I-2,J+1)-T(I-2,1)*T(I-1,J+1))/T(I-1,1)
230   NEXT J
240 NEXT I
250 FOR I=1 TO N+1
260   FOR J=1 TO K
270     PRINT T(I,J),
280   NEXT J
290 PRINT
300 NEXT I

```

Routh's stability criterion states that the number of sign changes in the first column of the Routh table is equal to the number of roots of the characteristic equation with positive real parts. Once Routh's array is formed, Routh's criterion can be applied to determine the stability of the system. The method is illustrated next by examples.

Example 2.8-1

Consider the characteristic equation

$$s^3 + 2s^2 + 4s + 10 = 0$$

The Routh table is

s^3	1	4
s^2	2	10
s^1	-1	
s^0	10	

There are two sign changes when the first column is scanned from top to bottom (i.e., 2 to -1, and -1 to 10). Hence, from Routh's criterion, there are two roots with positive real parts and the system is unstable.

Example 2.8-2

A system is described by the characteristic equation

$$s^4 + 10s^3 + 35s^2 + 50s + 24 = 0.$$

Determine whether the system is stable or not.

The Routh table is

s^4	1	35	24
s^3	10	50	
s^2	30	24	
s^1	42		
s^0	24		

There are no sign changes in the first column of the table. Thus, from Routh's criterion, there are no roots with positive real parts and the system is stable.

When constructing Routh's table, there are two cases which merit special consideration. The first case occurs when the first term in a row of the table is zero, but the remaining terms are not zero. One way to handle this case, is to replace the zero term by a small number ϵ and complete the table by taking the limit as $\epsilon \rightarrow 0$. Any sign changes in the first column can then be found. To illustrate this, consider the following example.

Example 2.8-3

The characteristic equation of a system is

$$s^4 + s^3 + s^2 + s + 10 = 0$$

The Routh table is

s^4	1	1	10
s^3	1	1	
s^2	ϵ	10	
s^1	$(\epsilon-10)/\epsilon$		
s^0	10		

The first term of the third row is zero and, thus, replaced by ϵ . Taking the limit as $\epsilon \rightarrow 0$ results in the first column sequence of terms: 1, 1, 0, $-\infty$, and 10. The two sign changes (0 to $-\infty$ and $-\infty$ to 10) imply two roots with positive real parts. Thus, the system is unstable.

The second special case occurs when all the terms in a row of the table are zero. This condition indicates that the characteristic polynomial has a factor with terms of even power. The roots of an even factor polynomial occur in pairs and are of equal magnitude and opposing signs. Thus, such roots may be real with equal magnitude but opposite signs, or they may be imaginary conjugates. In either case, the corresponding system is unstable, and the fundamental question about stability has been answered.

Additional information can be obtained from the Routh table. The coefficients of the even factor polynomial are the terms of the row just above the all-zero row. This polynomial is a factor of the characteristic equation and the coefficients of its derivative may be used in place of the zero row. The Routh table can be completed as illustrated by the following example.

Example 2.8-4

A system is described by the characteristic equation

$$s^3 + 3s^2 + 2s + K = 0$$

where the coefficient K is adjustable. The Routh table is

s^3	1	2
s^2	3	K
s^1	$(6-K)/3$	
s^0	K	

From the table, we see that the system is stable if $0 < K < 6$. However, if $K=6$, the Routh table becomes

s^3	1	2
s^2	3	6
s^1	0	
s^0	6	

In this case, an entire row is zero, even though the row consists of only one element. This suggests that the even factor polynomial, $3(s^2+2)$, from the row above the all-zero row, is a factor of the characteristic equation. Thus, we can write

$$s^3 + 3s^2 + 2s + 6 = (s^2 + 2)(s - p_1)$$

from which we obtain $p_1 = -3$ by equating coefficients. The characteristic roots or poles of the system are

$$p_1 = -3, \text{ and } p_{2,3} = \pm j\sqrt{2}$$

This means that the system is marginally or critically stable. It is obvious from the Routh table that the system is unstable if $K > 6$, because two of the roots are in the right-half s -plane. The system is also unstable if $K < 0$.

2.8-2 Root Locus Diagram

In general, when the stability of a system depends on the value of some adjustable parameter K , such as an amplifier gain, the system is said to be **conditionally stable**. *The mapping (trajectory) of the roots of the characteristic equation as K varies is known as the **root locus diagram**.*

The stability of a system can be readily evaluated from the mapping of its characteristic roots. In addition, the root locus diagram is a valuable design tool which can be constructed either analytically or by using digital computer methods or programs. The basic scheme for plotting a root locus diagram is illustrated by the following examples. A more systematic procedure as well as relevant computer programs can be found in the literature.

Example 2.8-5

Determine the root locus diagram of a control system whose open-loop transfer function is given by

$$H(s)G(s) = \frac{K}{s(s+2)}$$

Assume that K represents a proportional controller gain and that $0 < K < \infty$. The characteristic equation is

$$1 + H(s)G(s) = s^2 + 2s + K = 0$$

and the characteristic roots are

$$s_{1,2} = -1 \pm \sqrt{1-K}$$

For $K=0$, the roots are 0 and -2. These are also the poles of the open-loop transfer function $G(s)$. For $K=1$, the roots are both -1, and for $0 < K < 1$, the roots are real and distinct.

If $K=0.75$, for instance, the roots are -0.5 and -1.5 . For $K>1$, the roots become complex and if $K=5$, the roots are $-1+j2$, and $-1-j2$. A computer-drawn plot of the roots for $0<K<80$, the root locus diagram, is shown in Fig. 2.8-3.

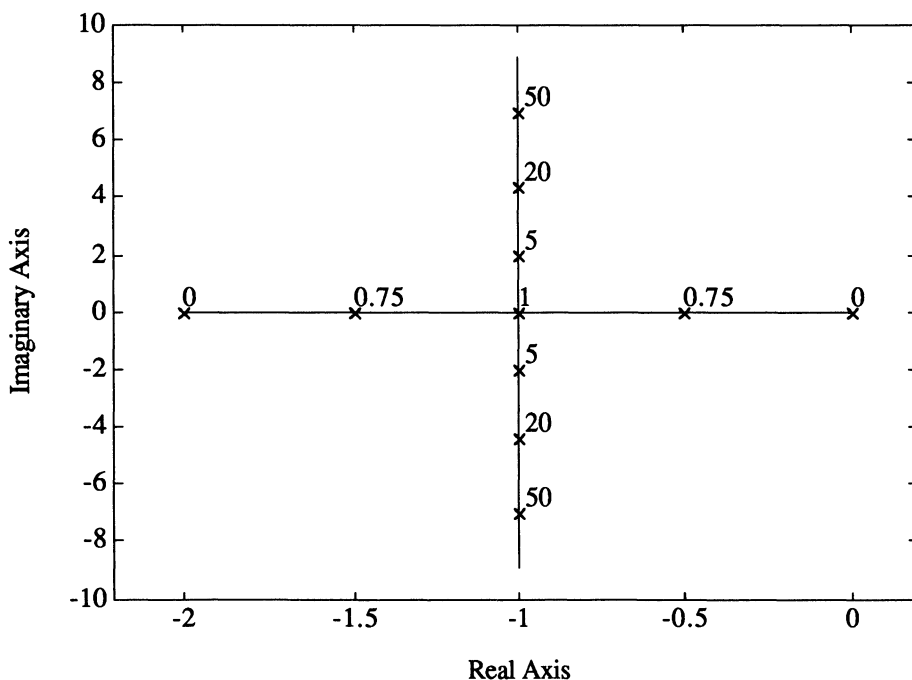


Fig. 2.8-3 The root locus diagram for the characteristics equation $s^2+2s+K = 0$; positive K ($0 < K < 80$).

Example 2.8-6

Determine the root locus diagram of a control system whose open-loop transfer function is given by

$$H(s)G(s) = \frac{K}{s(s+1)(s+2)}$$

The characteristic equation is

$$1 + H(s)G(s) = s^3 + 3s^2 + 2s + K = 0$$

which is identical to the characteristic equation considered in Example 2.8-4. From Example 2.8-4, the following information is available:

- If $K=0$, the roots are the same as the poles of $G(s)$, that is, 0, -1, and -2.
- If $K=6$, the roots are -3 and $\pm j\sqrt{2}$; thus the system is marginally/critically stable.

Additional information can be obtained from the characteristic equation by varying K and solving for the roots. The root locus diagram for $0 < K < 20$ is shown in Fig. 2.8-4.

Example 2.8-7

Determine the root locus diagram of the PDF control system used in Example 2.6-4 as the integral control gain K_I varies. The open-loop transfer function is given by

$$H(s)G(s) = \frac{500K_I}{s(s^2 + 40s + 800)}$$

The characteristic equation is

$$1 + H(s)G(s) = s^3 + 40s^2 + 800s + 500K_I = 0$$

If $K_I=0$, the characteristic roots are located at 0, $-20 \pm j20$. As K_I increases, the real root moves on the negative real axis, away from the origin; the complex pair of poles moves toward the imaginary axis. If $K_I=16$, for instance, the roots are located at -20, and $-10 \pm j17.3205$; the system is stable. If $K_I=64$, the complex roots become purely imaginary, resulting in a marginally stable system; finally, if $K_I=80$, the complex pair of roots moves to the right-half s -plane, making the system unstable. The root locus diagram is shown in Fig. 2.8-5.

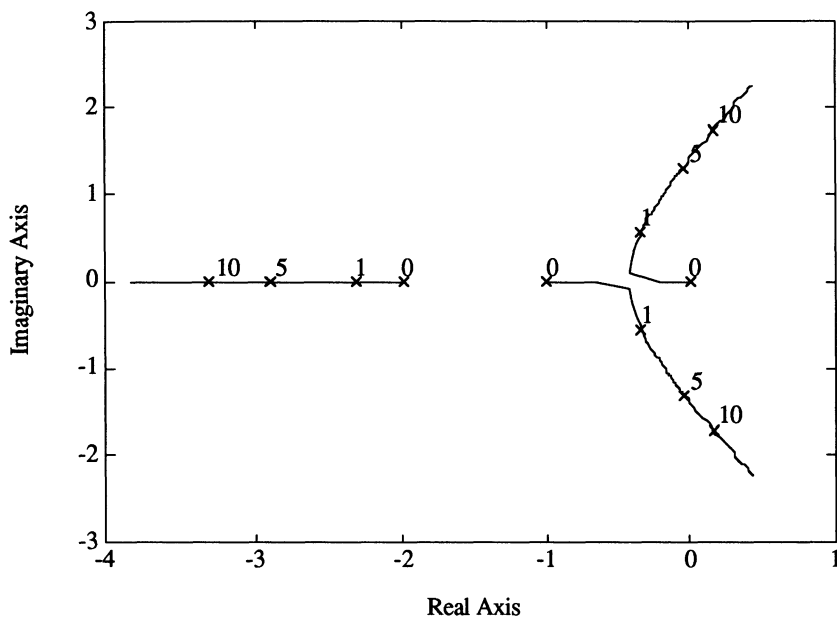


Fig. 2.8-4 The root locus diagram for $s^3 + 3s^2 + 2s + K = 0$; positive K ($0 < K < 20$).

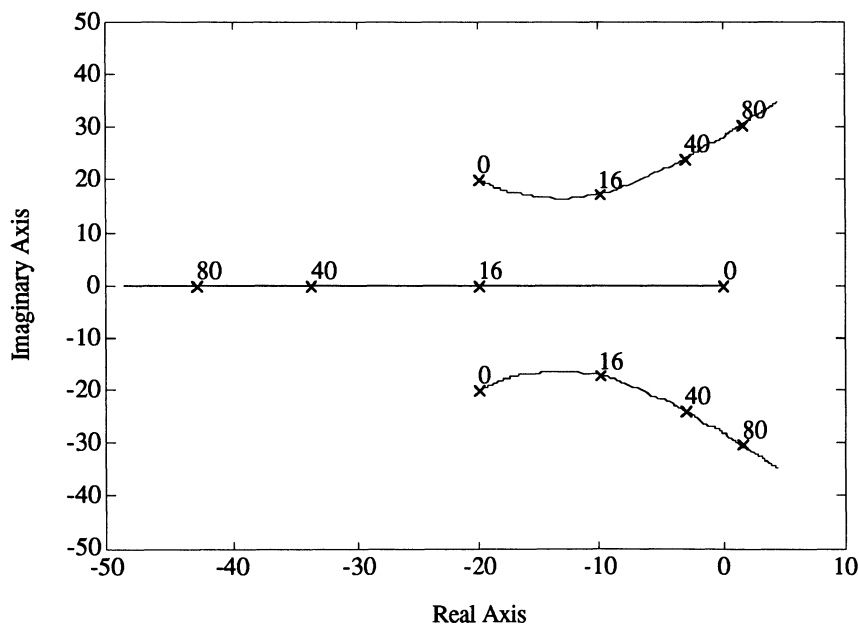


Fig. 2.8-5 The root locus diagram for the system used in Example 2.8-7; $0 < K_I < 120$.

2.9 FREQUENCY RESPONSE CONCEPTS

The term frequency response refers to the steady-state response of a system due to a sinusoidal input. The response of a system to an input of various frequencies can be expressed graphically by a plot of the system transfer function versus frequency, usually ω . Such a plot is known as the **frequency response** of the system.

Since the transfer function of a system is a complex quantity with real and imaginary components, it is common to represent frequency response information by plotting the magnitude (gain) and phase parts of the transfer function versus frequency. As we will see in the following sections, frequency response plots can be drawn using various methods and scales.

Like time domain methods, frequency domain methods can be used for both analysis as well as design of physical systems. One advantage of the frequency domain approach is that it is relatively easy to determine the frequency response of a system experimentally. This is attractive in applications where adequate mathematical models for some or all components of a system are not available.

In addition to being practical, frequency domain methods provide the designer with important information concerning the filtering characteristics of a system. In the case of feedback control systems, for instance, this is advantageous because it permits the determination of appropriate passband characteristics so that a system may be designed to track desirable inputs and reject noise and other disturbances.

Some general concepts relevant to the analysis and design of systems are presented and illustrated next. One of the main objectives is to determine the correlation, direct or indirect, that exists between transient and frequency responses.

2.9-1 Frequency Response Plots

The first step in the procedure for obtaining frequency response data analytically is to replace s by $j\omega$ in the system transfer function $G(s)$. That is, if a system is described by its transfer function $G(s)=C(s)/R(s)$, then the complex form is written in rectangular coordinates as

$$G(j\omega) = G(s) \Big|_{s=j\omega} = \text{Re}(\omega) + j\text{Im}(\omega) \quad (2.9-1)$$

where $\text{Re}(\omega)$ is the real part of $G(j\omega)$ and $\text{Im}(\omega)$ its imaginary part. The previous equation can be expressed in the polar form

$$G(j\omega) = |G(j\omega)| \angle \phi(\omega) \quad (2.9-2)$$

$|G(j\omega)|$ represents the magnitude of the output-to-input ratio and is given by

$$|G(j\omega)| = \sqrt{[\text{Re}(\omega)]^2 + [\text{Im}(\omega)]^2} \quad (2.9-3)$$

and $\phi(\omega)$ is the phase angle given by

$$\phi(\omega) = \tan^{-1} \left[\frac{\text{Im}(\omega)}{\text{Re}(\omega)} \right] \quad (2.9-4)$$

The frequency response of a system with transfer function $G(s)$ is characterized completely by the magnitude ratio and phase angle expressions - summarized by Eq. (2.9-3) and Eq. (2.9-4), respectively. Eq. (2.9-1) can be programmed on a digital computer in order to obtain relevant response data and plots.

A frequency analysis program called FREQC (FREQuency Continuous) has been used to obtain data for this book. The program requires that the transfer function $G(s)$ is in unfactored, rational polynomial form. The output data of the program can be printed out, or stored in a file for plotting by spreadsheet or other graphics programs.

An alternative but more descriptive method for determining the frequency response of a system analytically assumes $G(j\omega)$ is in the factored form

$$G(j\omega) = \frac{K e^{-j\omega T} (1+j\omega/z_1)(1+j\omega/z_2) \dots (1+j\omega/z_m)}{(j\omega)^q (1+j\omega/p_1)(1+j\omega/p_2) \dots (1+j\omega/p_n)} \quad (2.9-5)$$

The parameters $-z_1, -z_2, \dots, -z_m$ represent the zeros, $-p_1, -p_2, \dots, -p_n$ the poles, K the steady-state gain, and q the "type" of the system, with $q=0, 1, 2$, etc. Then, the magnitude ratio is

$$|G(j\omega)| = \frac{K |1+j\omega/z_1| |1+j\omega/z_2| \dots |1+j\omega/z_m|}{|j\omega|^q |1+j\omega/p_1| |1+j\omega/p_2| \dots |1+j\omega/p_n|} \quad (2.9-6)$$

where, for an arbitrary constant a , $|1+j(\omega/a)| = \sqrt{1+(\omega/a)^2}$. The phase angle, in degrees, is given by

$$\phi(\omega) = [\theta_1(\omega) + \theta_2(\omega) + \dots + \theta_m(\omega)] - [\psi_1(\omega) + \psi_2(\omega) + \dots + \psi_n(\omega)] - 90q - \omega T \quad (2.9-7)$$

where $\theta_1(\omega) = \tan^{-1}(\omega/z_1)$, $\psi_1(\omega) = \tan^{-1}(\omega/p_1)$, etc.

The use of Eq. (2.9-6) and Eq. (2.9-7) to obtain frequency responses is illustrated by general and specific examples. Of the several graphical representations available, the practical **logarithmic** or **Bode plots** are emphasized.

The basic factors present in the transfer function of Eq. (2.9-5) and the corresponding logarithmic plots are considered first. Composite asymptotic and exact plots for the general transfer function can be drawn with relative ease by simply adding the individual curves graphically.

First-Order Factor: $\frac{a}{s+a}$

Consider a first-order system described by its transfer function

$$G(s) = \frac{a}{s+a} = \frac{1}{1+s/a} \quad (2.9-8)$$

where a is an arbitrary real constant. Letting $s=j\omega$ results in

$$G(j\omega) = \frac{1}{1+j\omega/a} = \frac{1 \angle \phi(\omega)}{|1+j\omega/a|} \quad (2.9-9)$$

from which the magnitude and phase relationships become

$$|G(j\omega)| = \frac{1}{\sqrt{1+(\omega/a)^2}}$$

(2.9-10)

and

$$\phi(\omega) = -\tan^{-1}(\omega/a)$$

(2.9-11)

Rectangular Plots. The magnitude and phase expressions represented by Eq. (2.9-10) and Eq. (2.9-11) can be plotted vs. frequency on linear scales. They are usually referred to as **rectangular plots**. To sketch these functions approximately, the brief information shown in the first three columns of Table 2.9-1 is adequate. The complete set of magnitude and frequency responses is shown in Fig. 2.9-1.

Table 2.9-1 Frequency data for first-order factor

ω	$ G(j\omega) $	$\phi(\omega)$ in degree	$ G(j\omega) $ in db
$\ll a$	1.0	0.0	0.0
$0.1a$	0.995	-5.7	-0.04
a	0.707	-45.0	-3.0
$10a$	0.099	-84.3	-20.08
$\gg a$	0.0	-90.0	-20.0

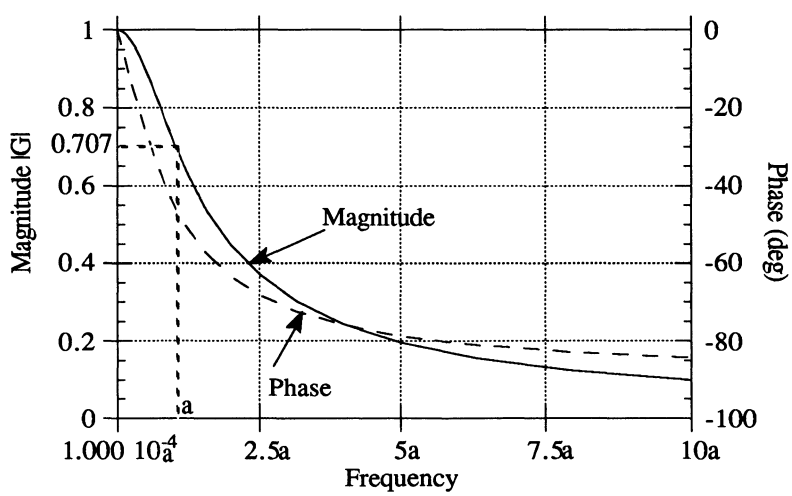


Fig. 2.9-1 Rectangular frequency plots of the first-order system $G(s)=a/(s+a)$.

Bode (Logarithmic) Plots. An alternative method of displaying frequency response information is the **Bode** or **logarithmic plot**, which uses logarithmic rather than linear scales. One advantage of the logarithmic scales is the ability to present much larger ranges of frequency and magnitude information on a given size plot. Also, the

logarithmic method is simple and systematic, because the multiplication and division of magnitudes implied by Eq. (2.9-6) are transformed into addition and subtraction. Thus, the frequency response curves can be sketched graphically, instead of having to evaluate them analytically.

The standard representation for Bode plots is on semi-log paper, with the vertical linear scale representing magnitude, or phase, and the horizontal log scale representing frequency. The usual unit for magnitude is the **decibel** (db), defined as

$$|G(j\omega)| \text{ in db} = 20\log|G(j\omega)| \quad (2.9-12)$$

where, in general, **log** denotes logarithm base 10, and **ln** logarithm base e, with $e=2.718$. The usual unit for phase is degrees, and for frequency it may be rad/sec, or Hz, depending on whether the independent variable is ω or f .

Two common units used to represent frequency intervals or bands are the decade and the octave. A **decade** represents a frequency range from a to b , where $b=10a$ or the ratio $b/a=10$. For instance, the band from 1 to 10 rad/sec is 1 decade wide. The number of decades from a to b is given by

$$(\text{Decades in the band from } a \text{ to } b) = \log(b/a) \quad (2.9-13)$$

An **octave** is a band from a to b where $b=2a$, or $b/a=2$. For instance, the frequency band from 10 to 20 rad/sec is 1 octave wide. The number of octaves in the frequency band from a to b is given by

$$(\text{Octaves in the band from } a \text{ to } b) = \frac{\log(b/a)}{\log 2} \quad (2.9-14)$$

To obtain a quick sketch of the Bode plots for the transfer function of Eq. (2.9-8), the following asymptotic approximation scheme is used. For our system, the magnitude ratio in db is given by Eq. (2.9-9) as

$$|G(j\omega)| \text{ in db} = -20\log\sqrt{1+(\omega/a)^2} \quad (2.9-15)$$

From the fourth column in Table 2.9-1, in the low-frequency range, that is, for $\omega \leq 0.1a$, the magnitude ratio is approximately zero, and it can be represented by a straight-line asymptote of zero slope. In the high-frequency range, that is, for $\omega \geq 10a$, the magnitude ratio is approximately -20 db, and it can be represented by a straight-line asymptote with slope of -20 db/decade. It can be shown that -20 db/decade = -6 db/octave.

The frequency $\omega=a$ at which the two asymptotes meet is called the **corner frequency**; this frequency is also known as the **break frequency** or the **break point** and is the inverse of the time constant of the system.

The magnitude ratio at the break frequency $\omega=a$ is -3 db ($20\log 0.707$). The exact magnitude curve can be drawn by locating a few more points on either side of the break point. The phase angle varies from 0 to -90 degrees, as can be seen from the third column in Table 2.9-1. The Bode plots for the first-order factor are shown in Fig. 2.9-2.

If the first-order factor is a zero instead of a pole, the slope of the high-frequency asymptote would be +20 db/decade, and the phase angle would vary from 0 to +90 degrees. Also, the magnitude ratio at the break frequency would be +3 db, instead of -3 db. The rectangular plots for such a zero would be affected in an analogous fashion.

Observe that moving the break frequency to the left or right does not change the shape of the magnitude and phase curves. However, if the steady-state gain of the system is not unity, the magnitude curve gets shifted up for $K>1$, and down if $K<1$.

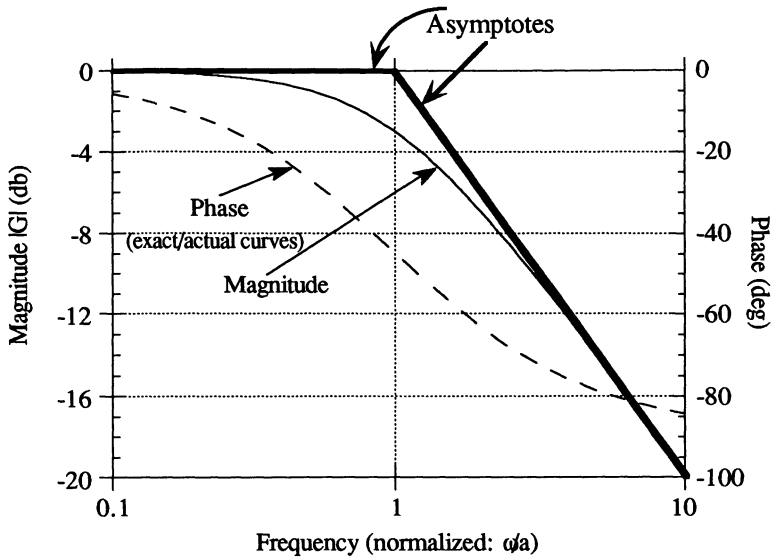


Fig. 2.9-2 Bode (logarithmic) plots for the first-order factor $a/(s+a)$.

Constant Factor: K

For a constant term K , its value in db is given by

$$20\log K \quad (2.9-16)$$

which is represented by a line parallel to the 0 db axis. The line is above the 0 db axis if $K > 1$, and below the 0 db axis if $K < 1$. The phase is zero.

Integral Factor: $\frac{1}{s}$

For a system acting as an integrator, its transfer function is given by

$$G(s) = \frac{1}{s} \quad (2.9-17)$$

The magnitude ratio is

$$|G(j\omega)| = \frac{1}{\omega} \quad (2.9-18)$$

and the phase angle is -90 degrees, a constant. For logarithmic plots,

$$|G(j\omega)| \text{ in db} = -20\log\omega \quad (2.9-19)$$

which can be represented by a straight line, that intercepts the 0 db axis at $\omega=1$ and has a slope of -20 db/dec. If the integral transfer function is of order q , that is, if

$$G(s) = \frac{1}{s^q} \quad (2.9-20)$$

then the respective magnitude and phase expressions are

$$|G(j\omega)| \text{ in db} = -20q \log \omega, \text{ and } \phi(\omega) = -90q \quad (2.9-21)$$

The magnitude line suggested by Eq. (2.9-21) is of slope -20q/decade, and the phase angle is -90q degrees, a constant.

Second-Order (Quadratic) Factor:
$$\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

Consider the second- order transfer function

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{1}{1 + 2\zeta(s/\omega_n) + (s/\omega_n)^2} \cdot \quad (2.9-22)$$

Letting $s=j\omega$ gives the complex function

$$G(j\omega) = \frac{1}{1 - (\omega/\omega_n)^2 + j2\zeta(\omega/\omega_n)} \cdot \quad (2.9-23)$$

Hence the magnitude ratio

$$|G(j\omega)| = \frac{1}{\sqrt{[1 - (\omega/\omega_n)^2]^2 + [2\zeta(\omega/\omega_n)]^2}} \quad (2.9-24)$$

and the phase angle

$$\phi(\omega) = -\tan^{-1} \left[\frac{2\zeta(\omega/\omega_n)}{1 - (\omega/\omega_n)^2} \right] \quad (2.9-25)$$

For logarithmic plots, the magnitude expression becomes

$$|G(j\omega)| \text{ in db} = -20 \log \sqrt{[1 - (\omega/\omega_n)^2]^2 + [2\zeta(\omega/\omega_n)]^2} \quad (2.9-26)$$

The asymptotic magnitude response can be sketched from the brief set of values given in Table 2.9-2. Exact magnitude and phase curves for various values of ζ are shown in Fig. 2.9-3.

The break frequency of the quadratic factor is the natural frequency ω_n . The shape of the magnitude and phase curves depends on the damping ratio ζ , and not on the break frequency. If the quadratic factor represents zeros, rather than poles, the slope of the high-frequency asymptote becomes +40 db/decade (+12 db/octave), and the phase angle varies from 0 to 180 degrees. For a non-unity gain system, the 0 db line is shifted up or down by the factor $20 \log K$.

The magnitude and phase expressions developed for the quadratic factor can be used to obtain frequency responses for the overdamped, critically-damped, and

underdamped system cases. This differs from the transient response case, where separate expressions had to be developed for each system case. Thus, for a critically damped or overdamped system, the composite frequency responses obtained by considering the two real poles are equivalent to those obtained by considering the quadratic factor.

Table 2.9-2. Frequency data for quadratic factor

ω	$ G(j\omega) $	$f(\omega)$ in degrees	$ G(j\omega) $ in db
$\leq 0.1\omega_n$	1	0	0
ω_n	$1/2\zeta$	-90	$20\log(1/2\zeta)$
$\geq 10\omega_n$	0	-180	-40

Delay Factor: e^{-sT}

The **time delay factor**, also known as **dead time** or **transportation lag**, is represented by the transfer function

$$G(s) = e^{-sT} \quad (2.9-27)$$

where T denotes the amount of time delay. As can be seen from the equation, the magnitude function is unity for all ω , and the negative phase angle, which varies linearly with T , is given by

$$\phi(\omega) = -\omega T \quad (2.9-28)$$

Thus, the time delay factor adds a phase shift of $-\omega T$ to the frequency response, but it does not alter the magnitude characteristics of the system.

The following frequency characteristics are often used to describe quantitatively the performance of a system.

Bandwidth (ω_b): **Bandwidth** is commonly defined as the frequency range from $\omega=0$ to the point at which the magnitude response of a system is 0.707 of (or 3 db below) its zero frequency (steady-state) value. The bandwidth indicates the rise time or speed of response of a system. It is also called the **passband**, because it describes the range of frequencies of the input signal over which a system performs satisfactorily. In the case of feedback control systems, this is significant, because it measures the ability of a system to follow or track accurately an input signal of a particular frequency.

The bandwidth of a first-order system is equal to its break frequency. It can be shown that for the standard-form second-order system, the bandwidth is given by

$$\omega_b = \omega_n \sqrt{1 - 2\zeta^2 + (2 - 4\zeta^2 + 4\zeta^4)^{1/2}} \quad (2.9-29)$$

A plot of the normalized ($\omega_n=1$) bandwidth ω_b versus ζ is shown in Fig. 2.9-4.

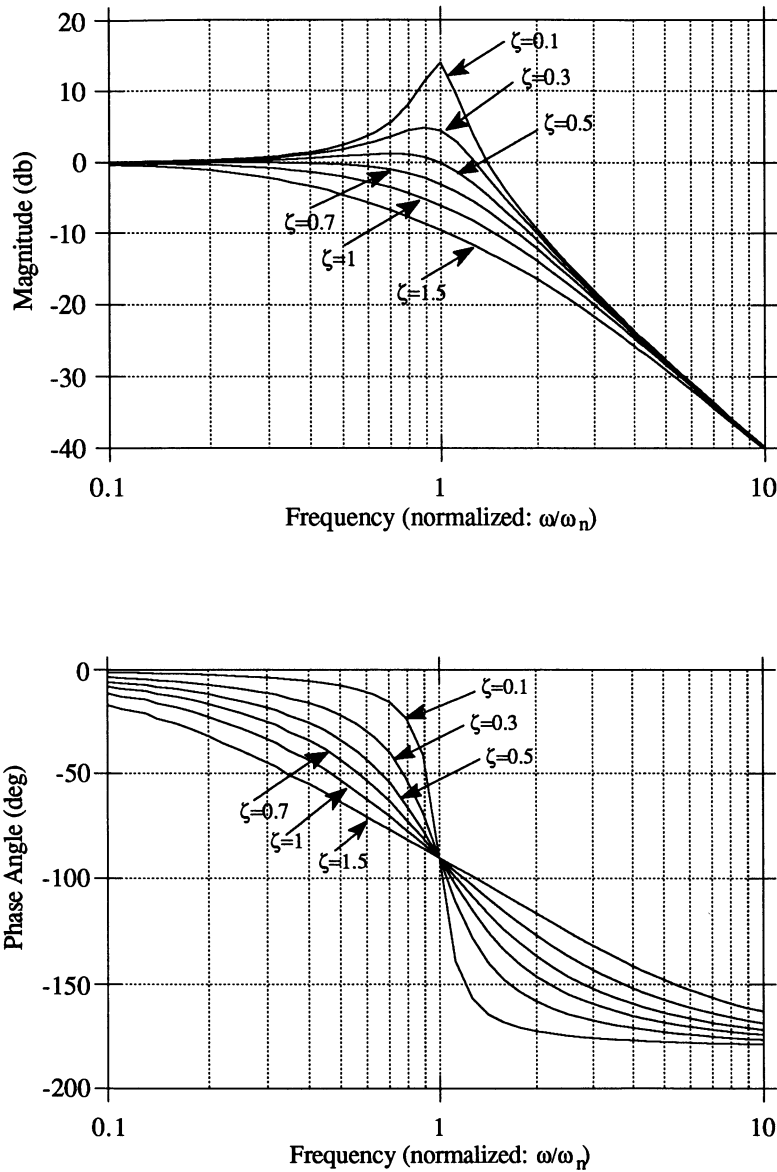


Fig. 2.9-3 Bode (logarithmic) plots for a standard-form, second-order system.

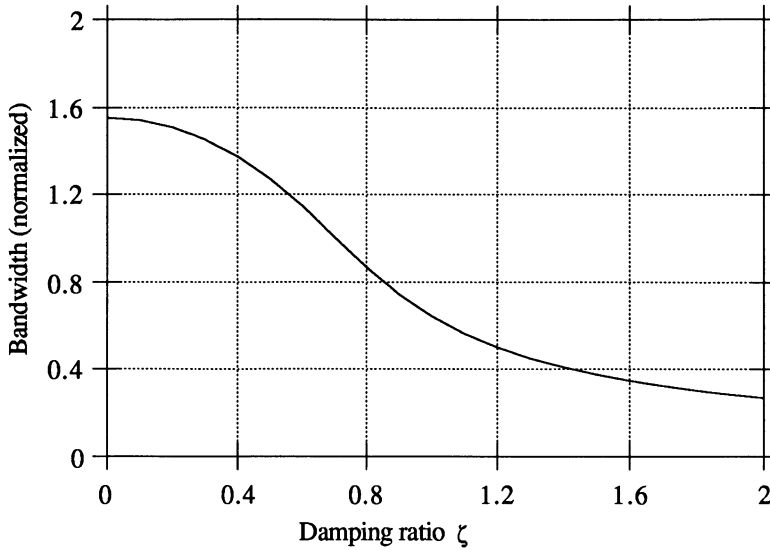


Fig. 2.9-4 Plot of normalized bandwidth vs. ζ for standard-form second-order system.

Cutoff Frequency (ω_c): *Cutoff frequency is defined as the frequency at which the magnitude ratio is 3 db below its zero frequency value. Thus, ω_c specifies the bandwidth of the system, which for the second-order system is given by Eq. (2.9-29).*

Cutoff Rate: *Cutoff rate is the rate at which the magnitude ratio decreases beyond the cutoff frequency. The cutoff rate is normally measured as the slope of the magnitude ratio curve, and indicates the ability of a system to distinguish a signal from noise.*

Resonance Peak (P_r): *Resonance peak is the maximum value of the magnitude response and is defined only for systems with damping ratios of $0 < \zeta \leq 0.707$. For the standard second-order system, P_r is given by*

$$P_r = \frac{1}{2\zeta\sqrt{1-\zeta^2}} \quad (2.9-30)$$

It can be seen from the above equation that the resonant peak increases as the damping ratio decreases. This behavior parallels the behavior of the transient response overshoot $P_{\max}$, and indicates that very small ζ values result in excessive resonance and overshoot peaks that may damage the system.

Resonance Frequency (ω_r): *Resonance frequency is defined as the frequency at which the resonance peak occurs for systems with $0 < \zeta \leq 0.707$. For the standard second-order system, the resonance frequency is given by*

$$\omega_r = \omega_n \sqrt{1-2\zeta^2} \quad (2.9-31)$$

Note that ω_r can be obtained from the magnitude ratio of Eq. (2.9-24) by taking its derivative and setting it equal to zero. Eq. (2.9-30) can then be obtained by evaluating the magnitude expression at $\omega=\omega_r$.

Example 2.9-1

Determine the Bode (logarithmic) plots of the closed-loop transfer functions of the second-order PDF and PID control systems considered earlier. Specifically, the respective transfer functions are

$$(a) \quad \text{PDF:} \quad \frac{C(s)}{R(s)} = \frac{5000}{s^2+100s+5000}$$

$$(b) \quad \text{PID:} \quad \frac{C(s)}{R(s)} = \frac{100s+5000}{s^2+100s+5000}$$

As can be seen from the transfer function, the PDF control system is a standard-form, second-order system. Its natural frequency is $\omega_n=70.71$ rad/sec, and the damping ratio is $\zeta=0.707$.

The PID control system has the same natural frequency and damping ratio. However, the response characteristics of the PID control system are also influenced by its zero, located at $s=-50$.

The Bode plots of the PDF control system can be seen in Fig. 2.9-3. For comparison purposes, the Bode plots of both the PDF and PID control systems are shown in Fig. 2.9-5. Note that the zero has added 20 db/dec to the magnitude, and 90 deg to the phase of the PID control system. Thus, even though the PID is a second-order system, the slope of its magnitude is -20 db/dec, instead of -40 db/dec, and its phase approaches -90 deg, instead of -180 deg.

Example 2.9-2

Determine frequency response characteristics for the third-order PDF control system (see Example 6.4) specified by the closed-loop transfer function

$$\frac{C(s)}{R(s)} = \frac{8000}{s^3+40s^2+800s+8000}$$

Notice that the break frequencies of this system are 20 rad/sec for the first-order factor, and 40 rad/sec for the second-order factor. The computer output for the frequency range of interest is shown in Fig. 2.9-6.

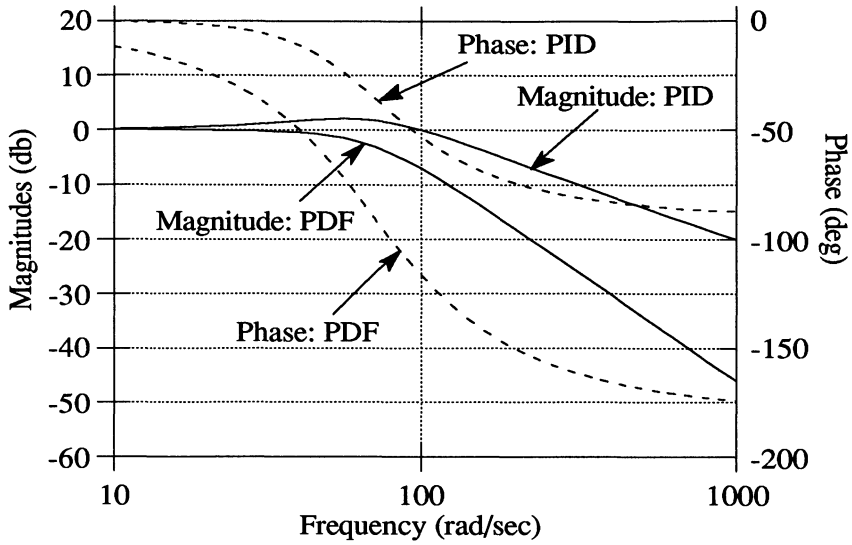


Fig. 2.9-5 Bode plots of PDF and PID control systems used in Example 2.9-5.

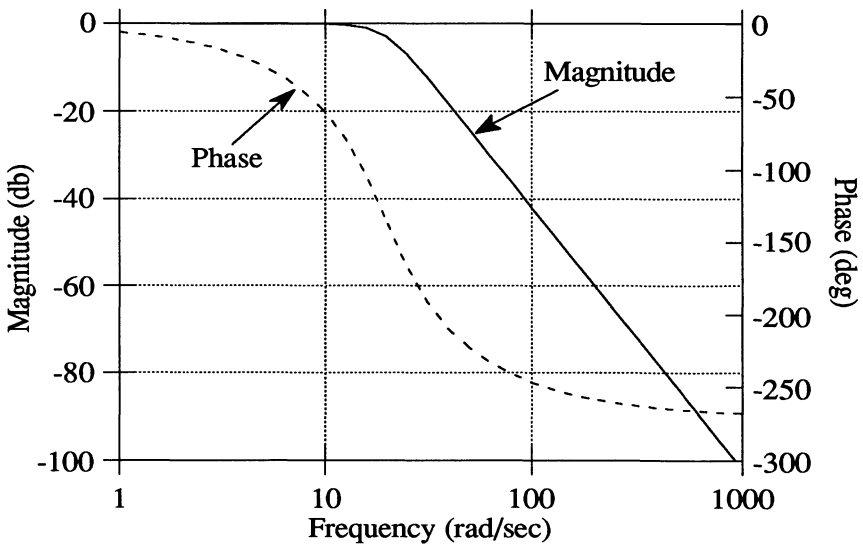


Fig. 2.9-6 Bode plots of 3rd-order PDF control system used in Example 2.9-2.

2.9-2 Frequency Plots of the Open-Loop Transfer Function

In the case of a feedback control system, the Bode plots of the closed-loop transfer function can be used to study its relative and absolute stability. The relative stability of the system can also be determined from the Bode plots of its open-loop transfer function $H(s)G(s)$. The following definitions are relevant to this type of analysis.

Gain Crossover Frequency (ω_g): *Gain crossover frequency is defined as the frequency at which the magnitude of the open-loop transfer function is unity. That is, at $\omega = \omega_g$*

$$|H(j\omega_g)G(j\omega_g)| = 1 \quad (2.9-32)$$

$$\text{or, in db,} \quad 20\log|H(j\omega_g)G(j\omega_g)| = 0 \quad (2.9-33)$$

Phase Crossover Frequency (ω_f): *Phase crossover frequency is the frequency at which the phase angle of the open-loop transfer function is -180 degrees (or π rad). That is, at $\omega = \omega_f$*

$$\phi(\omega_f) = \angle H(j\omega_f)G(j\omega_f) = -180 \text{ degrees} \quad (2.9-34)$$

Gain Margin (G_m): *Gain margin is defined as the magnitude of the reciprocal of the open-loop transfer function evaluated at the phase crossover frequency. That is,*

$$G_m = \frac{1}{|H(j\omega_f)G(j\omega_f)|} \quad (2.9-35)$$

$$\text{or} \quad G_m \text{ in db} = -20\log|H(j\omega_f)G(j\omega_f)| \quad (2.9-36)$$

The gain margin is the negative value in db of the open-loop transfer function evaluated at the phase crossover frequency. The gain margin measures how much the loop gain can be increased before the closed-loop system becomes unstable.

Phase Margin (F_m): *Phase margin is defined as 180 degrees plus the phase angle of the open-loop transfer function evaluated at the gain crossover frequency. Thus,*

$$F_m = [180 + \phi(\omega_g)] \text{ degrees} \quad (2.9-37)$$

The phase margin is the difference between the 180 degree line and the phase angle curve of the open-loop transfer function evaluated at the gain crossover frequency. The phase margin indicates how much the phase of the open-loop transfer function can be increased before the closed-loop system becomes unstable.

For absolute stability of a feedback control system, the phase and gain margins must both be positive. Negative margins indicate instability. For relative stability analysis, both phase and gain margins must be considered together in order to give sufficient indication of how close the system is to becoming unstable.

Example 2.9-3

For a second-order system, it can be shown that the gain crossover frequency is given by

$$\omega_g = \omega_n \sqrt{(1+4\zeta^4)^{1/2} - 2\zeta^2}$$

and that the phase margin is given by

$$F_m = \tan^{-1} \left[\frac{2\zeta}{\sqrt{(1+4\zeta^4)^{1/2} - 2\zeta^2}} \right]$$

It should be pointed out, that obtaining phase and gain margin information analytically involves tedious calculations, even for low-order systems. Computer programs, however, make the task practical and attractive.

Example 2.9-4

Draw Bode plots for the open-loop transfer functions of (a) the PID, and (b) the PDF control systems considered in Section 2.6-4. Then, determine the phase margin, and the gain margin of each system.

$$(a) \quad \text{PID:} \quad H(s)G(s) = \frac{K(K_P s + K_I)}{s^2}$$

$$(b) \quad \text{PDF:} \quad H(s)G(s) = \frac{KK_I}{s(s + KK_P)}$$

Substituting the parameter values $K=20$, $K_P=5$, and $K_I=250$, we obtain the respective open-loop transfer functions

$$(a) \quad \text{PID:} \quad H(s)G(s) = \frac{100s + 5000}{s^2}$$

$$(b) \quad \text{PDF:} \quad H(s)G(s) = \frac{5000}{s(s + 100)}$$

The Bode plots for the open-loop transfer functions are shown in Fig. 2.9-7 and Fig. 2.9-8. The phase margin is about 65 degrees for the PID control system, and about 64 degrees for the PDF control system. The gain margins for both systems are infinite. Such systems are often referred to as **inherently stable** systems.

Example 2.9-5

The open-loop transfer function of the double-integrator PDF control system used in Example 2.6-4 can be expressed in the form

$$H(s)G(s) = \frac{500K_I}{s(s^2 + 40s + 800)}$$

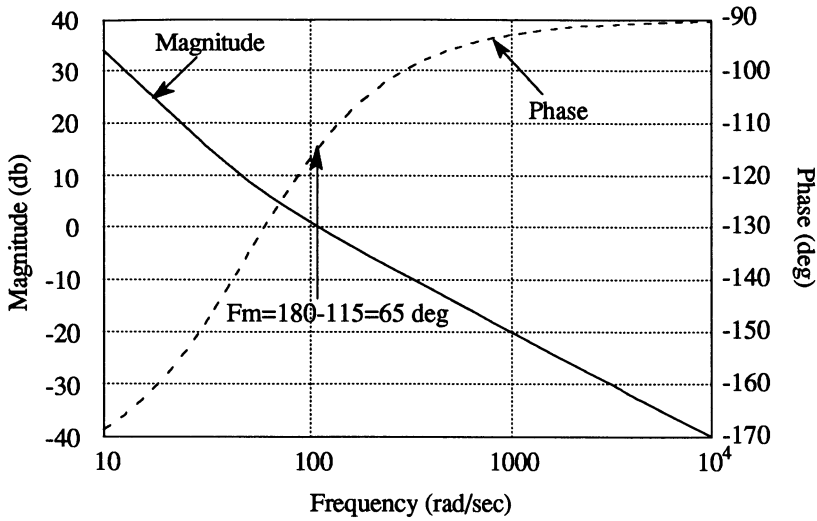


Fig. 2.9-7 Magnitude and phase versus frequency plots for the open-loop transfer function $H(s)G(s)$ of the PID control system used in Example 2.9-3.

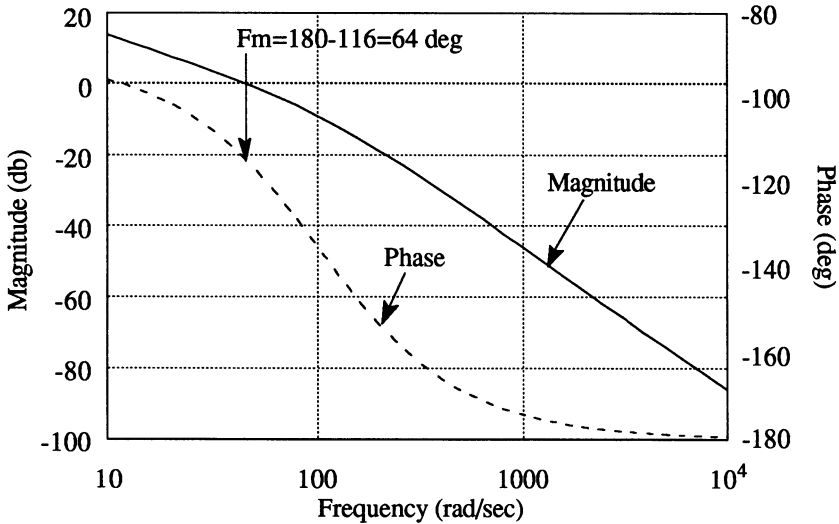


Fig. 2.9-8 Magnitude and phase versus frequency plots for the open-loop transfer function $H(s)G(s)$ of the PDF control system used in Example 2.9-3.

Draw Bode plots and determine the gain margin, phase margin, and stability of the system for the cases (a) $K_I=16$, and (b) $K_I=80$.

The Bode plots for the open-loop transfer functions corresponding to the two integral gain (K_I) values are shown in Fig. 2.9-9 and Fig. 2.9-10, respectively. If $K_I=16$, the phase margin is 60 degrees, and the gain margin is 12 db. If $K_I=80$, the phase margin is -8 degrees, and the gain margin is -1.9 db.

When $K_I=80$, the system is unstable because both margins are negative. It can be shown by Routh's criterion that this system is stable for values of $K_I < 64$, critically stable for $K_I=64$, and unstable for $K_I > 64$.

Log-Magnitude versus Phase Plot. An alternative method of portraying the phase and gain margins is the log-magnitude versus phase angle plot of the open-loop transfer function. This curve represents logarithmic magnitude in decibels versus phase angle in degrees - over a frequency range of interest. The log-magnitude versus phase plots for the system used in Example 2.9-5 are shown in Fig. 2.9-11.

Polar Plot. Another method of representing frequency characteristics for the open-loop transfer function is the **polar** or **Nyquist plot**; this is a plot of the imaginary versus the real components of the open-loop transfer function $H(s)G(s)$ for an appropriate frequency range. The polar plot enables us to determine the phase and gain margins and, thus, establish the stability of the system. Polar plots of the open-loop transfer functions used in Example 2.9-4 are shown in Fig. 2.9-12.

Nyquist criterion. Related to the polar plot is the frequency domain stability criterion, known as the **Nyquist criterion**. According to the Nyquist criterion, *if the open-loop zeros are on the left-half s -plane, a closed-loop system is stable if the polar plot of its open-loop transfer function does not enclose the $(-1,0)$ point*. From Fig. 2.9-12, the system of Example 2.9-4 is stable for $K_I=16$, and unstable for $K_I=80$.

2.9-3 Correlation of Transient Time and Frequency Responses

For a second-order system, the step transient response quantities developed earlier can be easily correlated with the frequency response quantities. For instance, from Eq. (2.3-48), the time at which the maximum overshoot occurs in the step response of an underdamped system can be expressed as a function of the resonance frequency given by Eq. (2.9-31). That is,

$$T_{\max} = \frac{\pi\sqrt{1-2\zeta^2}}{\omega_r\sqrt{1-\zeta^2}} = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} \quad (2.9-38)$$

The inverse relationship between $T_{\max}$ and ω_r implies that, for small values of ζ , large values of ω_r result in small values of $T_{\max}$ and rise time T_r ; this in turn implies a faster system response. The correlation of time and frequency quantities, as the damping ratio changes, is summarized in Table 2.9-3.

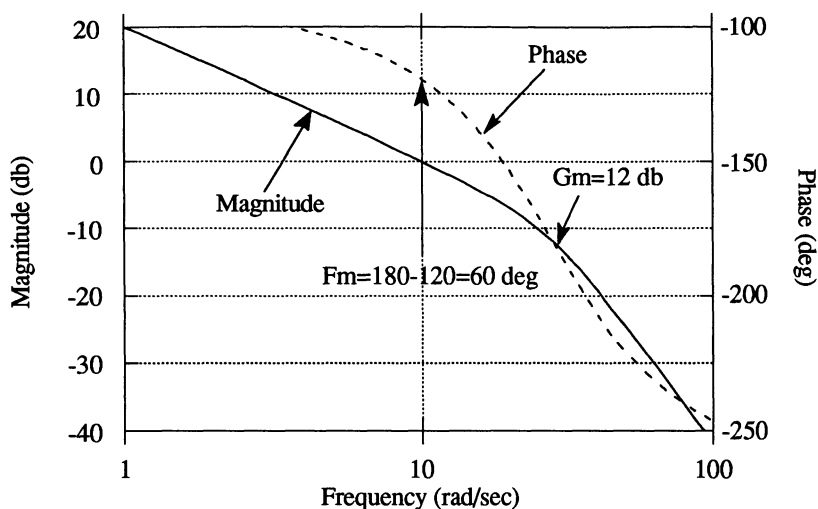


Fig. 2.9-9 Magnitude and phase versus frequency plots for the open-loop transfer function of the PDF control system used in Example 2.9-5 with $K_I = 16$.

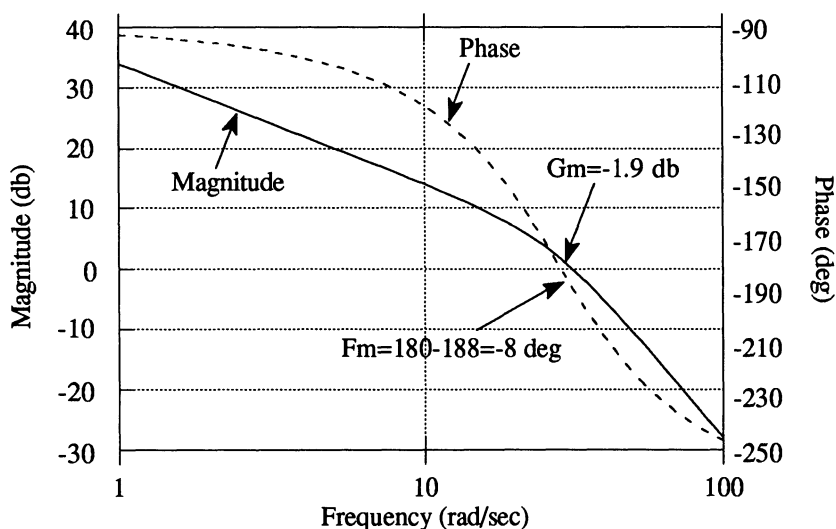


Fig. 2.9-10 Magnitude and phase versus frequency plots for the open-loop transfer function of the PDF control system used in Example 2.9-5 with $K_I = 80$.

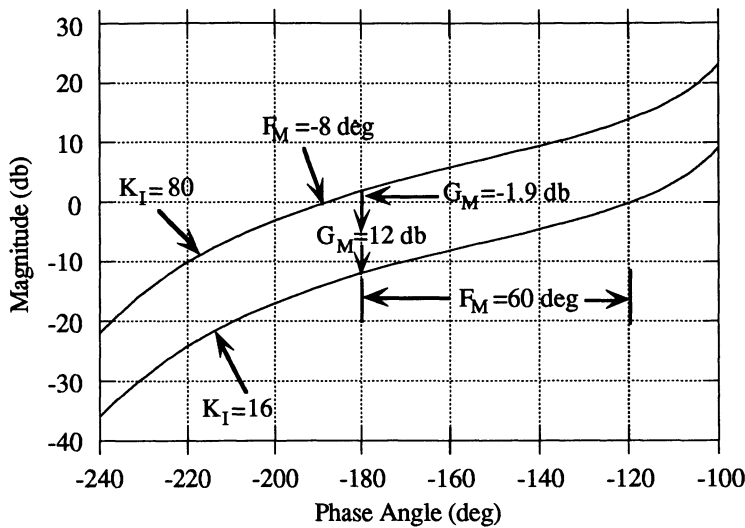


Fig. 2.9-11 Log-magnitude versus phase plots for the open-loop transfer functions of the PDF control system used in Example 2.9-5.

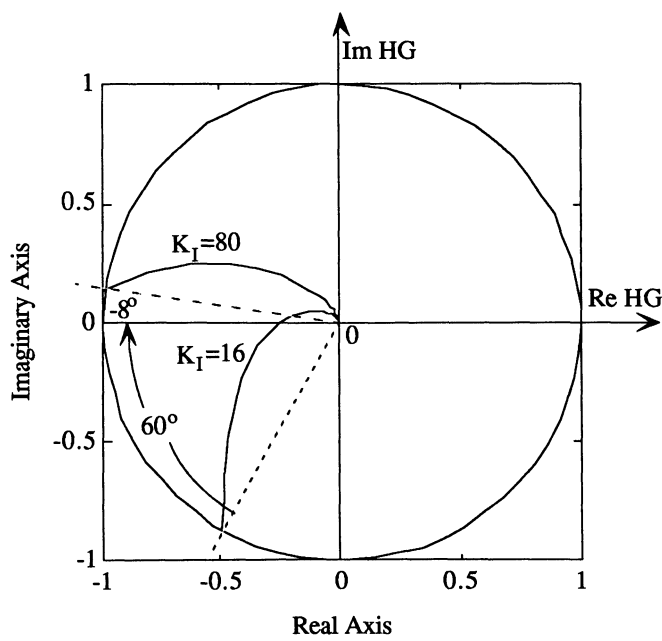


Fig. 2.9-12 Polar (Nyquist) plot for the GH function used in Example 2.9-4.

Table 2.9-3 Time and frequency quantities versus damping ratio ζ

ζ	$P_{\max}$	P_r	$\omega_n T_{\max}$	$\omega_r T_{\max}$	F_m
0.1	1.73	5.025	3.157	3.126	11.31
0.2	1.527	2.551	3.206	3.075	21.83
0.3	1.372	1.747	3.293	2.982	31.14
0.4	1.254	1.364	3.428	2.827	39.19
0.5	1.163	1.155	3.628	2.565	46.28
0.6	1.095	1.042	3.927	2.078	52.59
0.7	1.046	1.000	4.399	0.622	58.42

Example 2.9-6

A second-order system is characterized by the damping ratio $\zeta=0.5$ and natural frequency $\omega_n=10$ rad/sec. Determine its time constant (τ), time of maximum overshoot ($T_{\max}$), maximum overshoot ($P_{\max}$), rise time (T_r), delay time (T_d), resonance frequency (ω_r), resonance peak (P_r), bandwidth (ω_b), and phase margin (F_m).

The bandwidth is calculated from Eq. (2.9-29). The remaining quantities can be calculated from Table 2.3-1 and Table 2.9-3. Their values are summarized in the following table:

t	$T_{\max}$	$P_{\max}$	T_r	T_d	ω_r	P_r	F_m	ω_b
0.2	0.363	1.163	0.164	0.129	7.07	1.155	46.28	12.72

For higher than second-order systems, analytical techniques for the exact correlation between time and frequency responses become impractical. It is often the case, however, that higher-order systems can be approximated by second-order models, if their response is dominated by a pair of complex-conjugate poles. In such cases, the second-order relationships developed in this chapter can be applied to simplify the analysis and design. Otherwise, computer methods become necessary.

2.10 DIGITAL SIMULATION METHODS

Digital computer simulation is of growing importance in many fields, theoretical and applied. It is a numerical technique for conducting experiments on a digital computer using mathematical models that describe the behavior of physical systems.

The computer simulation of a system can be implemented in various ways. A direct approach is to write a computer program from the block diagram or mathematical model of the system. A second alternative is to use an appropriate simulation language. Both of these simulation methods are illustrated next by means of examples.

2.10-1 System Simulation by Programming Block Diagrams

When simulation languages are not readily available, simulation programs can be written directly from a system's block diagram using numerical integration and differentiation rules. In such cases, the integration time parameter must be chosen with care; if the integration time is too large, the results will be unpredictable, and if it is too small, computational time and errors may become excessive.

In general, an integration time which is at least 50 times smaller than the smallest time constant of the system should give acceptable results. If the smallest time constant of the system is not known, however, then we can start with an arbitrary integration time and keep making it smaller until the results stop changing appreciably, and look reasonable. The approach is illustrated by the following example.

Example 2.10-1

Write a program to simulate the analog PI control system shown in Fig. 2.10-1 for the parameter values $K=20$, $K_p=5$, and $K_I=250$ (see Example 2.6-3). Assume rectangular integration, that is, assume the output of an integrator is equal to its previous value plus its current input times the integration time.

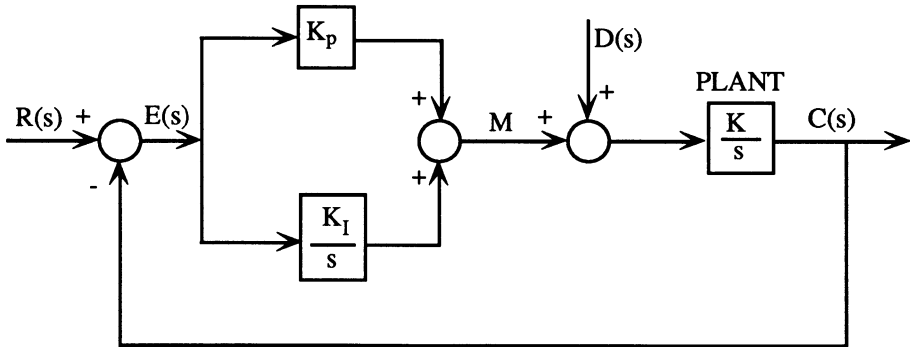


Fig 2.10-1 Block diagram of PI control system used for simulation in Example 2.10-1.

A general simulation program is shown in Fig. 2.10-2. The computer results are shown in Table 2.10-1. A plot of the results is shown in Fig. 2.10-3. Note that the block diagram of the system is described by the program lines 290 and 300, and the integrators are initialized in line 220. The integration time chosen is 0.0001 sec, or 200 times smaller than the time constant of the system.

```

130 PRINT"NOTE: FINTIM=FINISH TIME, DELT=INTEGRATION TIME, "
140 PRINT"OUTDEL=OUTPUT INTERVAL FOR THE SIMULATION."
160 INPUT"ENTER K, THE PLANT GAIN:"; K
170 INPUT"ENTER KP AND KI, THE CONTROLLER GAINS:"; KP,KI
180 INPUT"ENTER FINTIM, DELT, AND OUTDEL:"; T2,T0,DT
190 INPUT"ENTER REF, THE SIZE OF THE STEP INPUT:"; R
210 N1=T2/T0: N0=T2/DT: T1=DT-0.5*T0
220 I1=0: I2=0: E=R-I1: T=0
230 PRINT "TIME", "REF", "OUTPUT", "ERROR", "MANIP"
240 PRINT"=====", "=====", "=====", "=====", "====="
250 PRINT T, R, I1, E, (I2+KP)*E*K
260 N=1
270 FOR J=1 TO N1
280     T=J*T0
290     E=R-I1: I2=I2+E*T0
300     M=(I2*KI+E*KP)*K: I1=I1+M*T0
310     IF T<T1 THEN 350
320     N=N+1
330     PRINT T, R, I1, E, M
340     T1=N*DT-0.5*T0
350 NEXT J
360 END

```

Fig. 2.10-2 Program used to simulate the PID control system of Example 2.10-1.

Table 2.10-1 Input and output of the simulation program of Fig. 2.10-1.

TIME	REF	OUTPUT	ERROR	MANIP
0.000	1	0.0000	1.0000	100.0000
0.005	1	0.4407	0.5669	76.0277
0.010	1	0.7620	0.2433	53.4835
0.015	1	0.9797	0.0238	34.6175
0.020	1	1.1133	-0.1114	19.8205
0.025	1	1.1831	-0.1822	8.9313
0.030	1	1.2075	-0.2073	1.4705
0.035	1	1.2019	-0.2022	-3.1871
0.040	1	1.1788	-0.1794	-5.6935
0.045	1	1.1474	-0.1480	-6.6535
0.050	1	1.1139	-0.1146	-6.5841
0.055	1	1.0826	-0.0832	5.8987
0.060	1	1.0555	-0.0560	-4.9075
0.065	1	1.0338	-0.0341	-3.8283
0.070	1	1.0173	-0.0176	-2.8013
0.075	1	1.0056	-0.0058	-1.9062
0.080	1	0.9980	0.0019	-1.1784
0.085	1	0.9936	0.0063	-0.6229
0.090	1	0.9916	0.0084	-0.2259
0.095	1	0.9912	0.0088	0.0363
0.100	1	0.9918	0.0082	0.1914

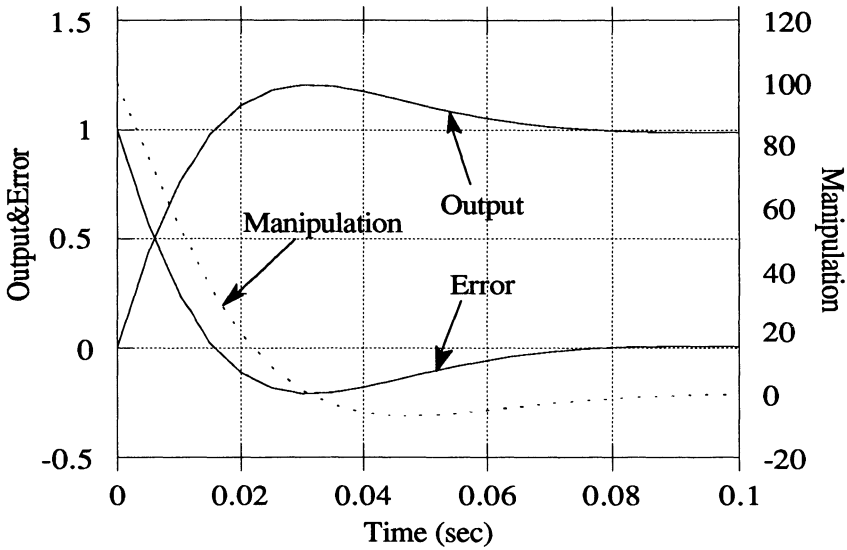


Fig. 2.10-3 Plot of simulation results for PI control system used in Example 2.10-1.

2.10-2 System Simulation Using ACSL

The Advanced Continuous Simulation Language (ACSL) is a problem-oriented language which makes simulating a system from its block diagram practical as well as realistic. The use of ACSL is attractive mainly because of its simple program statements, its flexible program structure, and its powerful set of pre-programmed functional blocks.

ACSL allows the user to specify the system in terms of block diagrams, mathematical equations, or both. It also permits the definition of special functions in terms of FORTRAN statements. Its application to simulate a system from its block diagram model is illustrated by the following example.

Example 2.10-2

Write an ACSL program to simulate the block-diagram model of the PI control system used in Example 2.10-1, above.

The ACSL program listing is shown in Fig. 2.10-4. The simulation results are literally identical to those shown in Fig. 2.10-2.

```

"*****"
"ACSL SIMULATION OF PI CONTROL SYSTEM OF EXAMPLE 2.10-2.
"
"PARAMETER VALUES: K=20, KP = 5, KI = 250, INPUT = UNIT STEP
"
"*****"
CINTERVAL CINT = 0.001      $ "DYNAMIC section interval time (sec.)"
CONSTANT K      = 20.0      $ "System constants"
CONSTANT KP     = 5.0
CONSTANT KI     = 250.0
INITIAL $ "Initialize variables before start of simulation"
OUTPUT         = 0.0
END $ "Initial"
DYNAMIC $ "Start simulation"
DERIVATIVE $ "Continuous analog control"
    REF        = STEP( 0.0 )
    ERROR      = REF - OUTPUT
    A1         = INTEG( ERROR, 0.0 )
    MANIP      = KP * ERROR + KI * A1
    OUTPUT     = K* INTEG( MANIP, 0.0 )
    TERMT( T .GE. 0.1 )      $ "Length of simulation sec."
END $ "Derivative"
END $ "Dynamic"
END $ "End of program"

```

Fig. 2.10-4 ACSL program for simulating the PI control system of Example 2.10-1.

2.10-2 System Simulation Using SIMULAB

SIMULAB is a simulation program that includes a graphical pre-processor. Thus, to simulate a dynamic system, continuous or discrete, we reconstruct the block diagram of the given system on the computer screen using appropriate blocks from SIMULAB's library. As an extension of MATLAB, SIMULAB adds many features specific to dynamic systems while retaining MATLAB's general purpose functionality. To illustrate the application of SIMULAB to simulate a continuous system, consider the following example.

Example 2.10-3

Simulate the analog PI control system of Example 2.10-1 using SIMULAB.

Using available blocks from SIMULAB's library, the simulation diagram shown in Fig. 2.10-5 is readily constructed. The signal generator block allows for sinusoidal or square wave inputs of selected frequencies and amplitudes. A step function generator is also available. Choosing rectangular (Euler) integration, the simulation results (reference input, error, and output) are shown multiplexed for displaying on the "scope" screen. If desired, the simulation data can be downloaded into an array for plotting and/or further analysis. For this example, the results matched those of the previous two examples. Actual computer screen outputs are shown in Fig. 2.10-6.

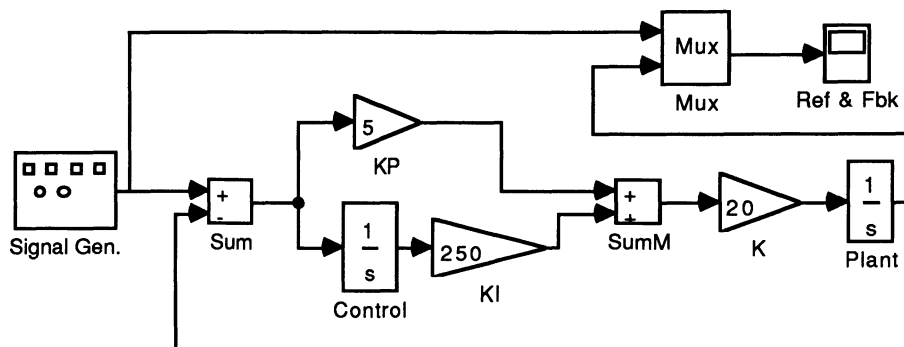


Fig. 2.10-5 SIMULAB simulation diagram for the PI control system used in Example 2.10-1.

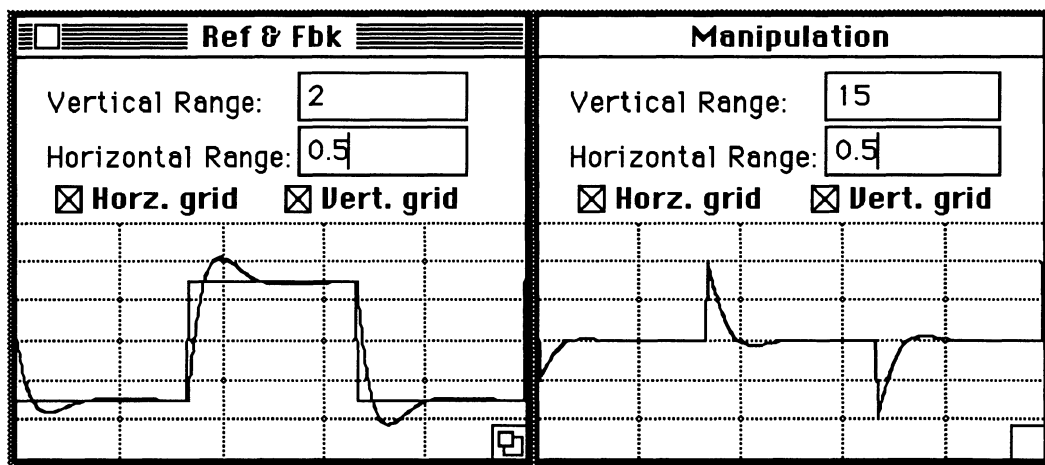


Fig. 2.10-6 SIMULAB simulation results for the PI control system of Fig. 2.10-5.

PROBLEMS

2-1. Systems A & B are represented by their respective differential equation models given by

$$(A) \quad y''(t) + 14y'(t) + 100y(t) = 100x(t)$$

$$(B) \quad y''(t) + 10y'(t) + 100y(t) = 100x(t)$$

where $x(t)$ and $y(t)$ represent the system input and output, respectively. Determine the following parameters for each system:

- (1) The (undamped) natural frequency (ω_n)
- (2) The damped natural frequency (ω_d)
- (3) The damping ratio (ζ)
- (4) The time constant(s) (τ)
- (5) The settling time (T_s)

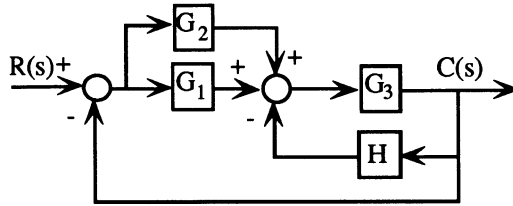
- 2-2.** Determine the unit-step response of each system in Problem 2-1 using the method of undetermined coefficients. Assume zero initial conditions. Plot the step responses on the same page for comparison.
- 2-3.** Determine and plot (on the same page) the unit-step response of the systems in Problem 2-1 using the Laplace transform method.
- 2-4.** Determine and plot the unit-impulse response of the systems in Problem 2-1 using the Laplace transform method.
- 2-5.** Determine and plot the unit-step response of the following system using the Laplace transform method. Assume zero initial conditions.

$$y'''(t) + 20y''(t) + 200y'(t) + 1000y(t) = 1000x(t)$$

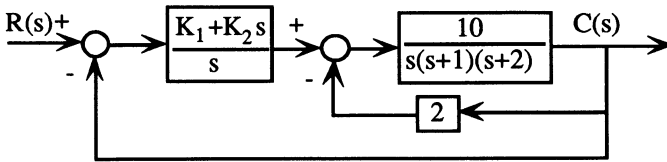
- 2-6.** Determine and plot frequency responses, that is, magnitude and phase vs. frequency of the analog systems in Problem 2-1 and Problem 2-5. Plot both magnitudes on one page and the phase angles on a different page for comparison. Use logarithmic scales for ω , linear scales for magnitude and phase angle.
- 2-7.** A system is described by its transfer function

$$G(s) = \frac{Y(s)}{X(s)} = \frac{(s+10)}{(s+5)(s^2+3s+2)} .$$

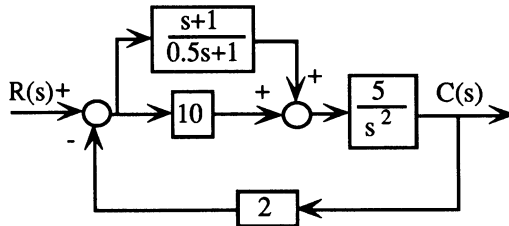
- (a) Determine the time constant(s) of the system
 - (b) Determine the characteristic equation of the system
 - (c) Determine the eigenvalues of the system
 - (d) Determine the differential equation model of the system
 - (e) Determine whether the system is stable or not and why.
- 2-8.** Determine and sketch the unit-step response of the system of Problem 2 assuming zero initial conditions.
- 2-9.** Determine and plot magnitude and phase vs. frequency responses for the system of Problem 2-8.
- 2-10.** Simplify the following block diagram to find the transfer function $C(s)/R(s)$. It is given that $G_1(s) = \frac{10}{s}$, $G_2(s) = 1+2s$, $G_3(s) = 100$ and $H(s) = \frac{1}{s+1}$.



- 2-11. Find $C(s)/R(s)$ for the system shown below. Also, find $E(s)/R(s)$ and then find the steady-state error value if the input is a ramp function.



- 2-12. Find the closed-loop transfer function $C(s)/R(s)$ for the system shown below. Then, determine and plot the unit-step response of the system

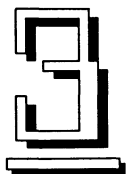


- 2-13. The open-loop transfer function of a system is given by

$$G(s)H(s) = \frac{K(s+6)}{s(s+1)(s+10)(s^2+4s+13)}$$

Sketch/plot the root locus diagram when K is varied from 0 to ∞ . Find the value of K at which the closed-loop system becomes unstable.

- 2-14. Determine the magnitude and phase vs. frequency plots for the open-loop transfer function of Problem 2-13. Find the gain and phase margins of the system.
- 2-15. Write a program to simulate the block diagram of the system in Problem 2-12. Determine and plot the step response of the system



DISCRETE-TIME SYSTEMS

Systems, Signals, and Models

3.1 INTRODUCTION

A signal is defined as continuous within a time range if its amplitude is specified for all t in that range, except possibly at a finite number of discontinuities. If a continuous time variable is quantized, that is, if t is represented by a set of distinct values denoted by t_k , where k takes on positive integer values exclusively, the signal is defined only at discrete-time instants, known as **sampling instants**. *Signals which are defined only at discrete instants of time are called **discrete-time**, or just **discrete signals** and can be represented by a sequence or collection of numbers.*

Even though the spacing of the sampling instants may vary, fixed-time intervals are emphasized in this book. Accordingly, a discrete signal may be viewed as a sequence of numbers representing amplitudes equally-spaced in time. If the spacing is denoted by an arbitrary constant T , the discrete signal can be expressed by $x(kT)$; for notational convenience, $x(kT)$ is often represented by $x(k)$, where k represents discrete time. T is called the **sampling time** or **sampling period**, measured in seconds.

*A discrete signal which can assume any one of only a finite number of values at each sampling instant is often referred to as a **digital signal**. If the discrete signal can assume any one of a continuous range of values at each sampling instant, the signal is called a **sampled-data signal**, or just a **sampled signal**.* In general, sampled signals are assumed to have amplitudes which are constant between sampling instants and proportional to the value of the continuous signal at the sampling instants.

*A **discrete-time system**, or just a **discrete system**, is a system whose inputs and outputs are discrete signals.* A block diagram representation of a discrete system is shown in Fig. 3.1-1. In this diagram, $x(k)$ denotes the input signal and $y(k)$ the output signal. For a system at rest, the input-output transformation relationship is often represented by the functional notation

$$y(k) = P[x(k)] \quad (3.1-1)$$

where P (Processing) is a transformation operator that characterizes the system.

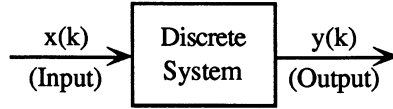


Fig. 3.1-1 Block diagram model of a discrete system.

A discrete system is said to be **homogeneous** if its response to the input

$$x(k) = ax_1(k) \quad (3.1-2)$$

is the response of the system to the input $x_1(k)$ multiplied by 'a', for all values of $x_1(k)$ and any constant 'a'. That is, for a discrete system to be homogeneous

$$y(k) = P[ax_1(k)] = aP[x_1(k)] \quad (3.1-3)$$

A discrete system is said to be **linear** if its response to the input

$$x(k) = ax_1(k) + bx_2(k) \quad (3.1-4)$$

is

$$y(k) = P[ax_1(k) + bx_2(k)] = aP[x_1(k)] + bP[x_2(k)] \quad (3.1-5)$$

for all values of the signals $x_1(k)$, $x_2(k)$, and for any constants a and b . Eq. (3.1-5) expresses the **principle of superposition** for discrete systems.

A discrete system is defined as **deterministic**, if all its parameters are defined completely; otherwise, it is called **nondeterministic**.

A discrete system is said to be **realizable** or **causal** if it does not respond before an input is applied; otherwise, it is called **nonrealizable (noncausal)**.

A discrete system is called **time-invariant** or **constant-parameter** if the response to a time-shifted input signal is a time-shifted output signal. That is, for a time-invariant discrete system

$$y(k-m) = P[x(k-m)] \quad (3.1-6)$$

for any values of discrete time k and delay or shift amount m . Thus, like its continuous counterpart, the output of a time-invariant discrete system depends on the form or shape of the input and not on its time origin.

A system in which continuous and discrete signals are present is known as a **hybrid system**. An example of such a system is shown in Fig. 3.1-2. Corresponding waveforms of interest are shown in Fig. 3.1-3.

In Fig. 3.1-2, the discrete system is assumed to be a digital computer. For a discrete system to communicate with the continuous world, devices such as analog-to-digital (ADC), and digital-to-analog (DAC) converters are used.

The ADC device samples the input signal $x(t)$ every T seconds, and converts each amplitude value to a binary number of sufficient bits according to the specified accuracy. The digital signal $x(k)$ is then processed further to achieve desired results, denoted by $y(k)$. Finally, the digital signal $y(k)$ is converted back to analog form $y(t)$ by the DAC device. In this step, the numerical content of a computer register is converted to an analog voltage which the DAC holds constant until the content of the register is updated.



Fig. 3.1-2 A system showing both continuous (analog) and discrete (digital) signals.

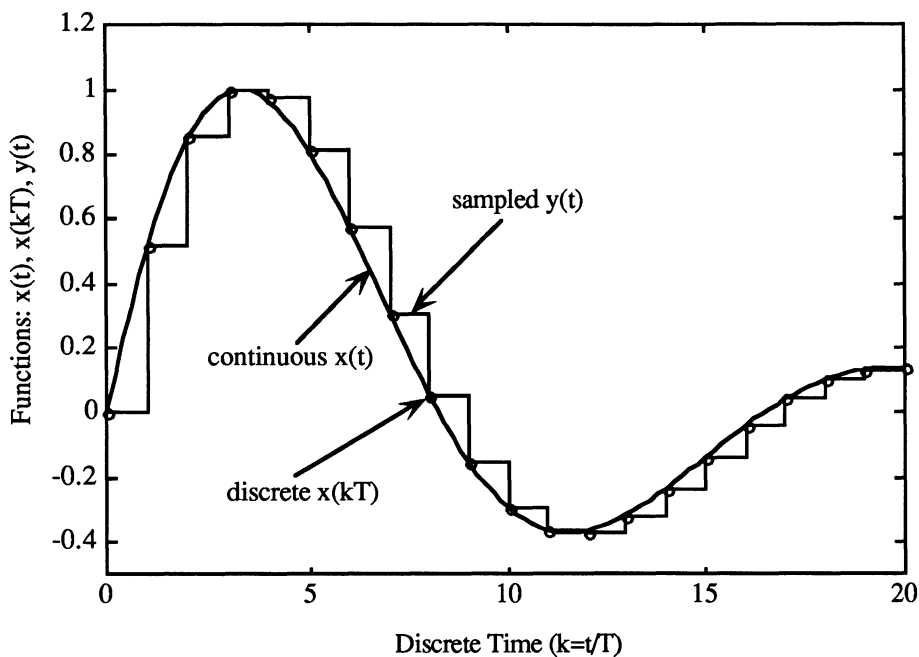


Fig. 3.1-3 Examples of continuous, discrete, and sampled signals.

3.1-1 Discrete Singularity Functions

The analysis of discrete systems can be facilitated by the use of discrete singularity functions or sequences of numbers which are used to describe input or test signals.

Perhaps the most important of the singularity functions is the **discrete unit-step** function defined by

$$u(kT) = u(k) = u_k = \begin{cases} 1 & \text{for } k \geq 0 \\ 0 & \text{for } k < 0 \end{cases} \quad (3.1-7)$$

for any sampling time T . A discrete unit-step function is shown in Fig. 3.1-4.

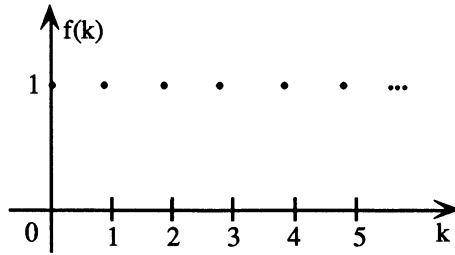


Fig. 3.1-4 The discrete unit-step function $f(k)=u(k)$.

A unit-step function can be delayed (displaced) j sampling times to the right; that is, its origin can be shifted from $k=0$ to $k=j$. Such a function is denoted by

$$u(k-j) = \begin{cases} 1 & \text{for } k \geq j \\ 0 & \text{for } k < j \end{cases} \quad (3.1-8)$$

and is shown plotted in Fig. 3.1-5.

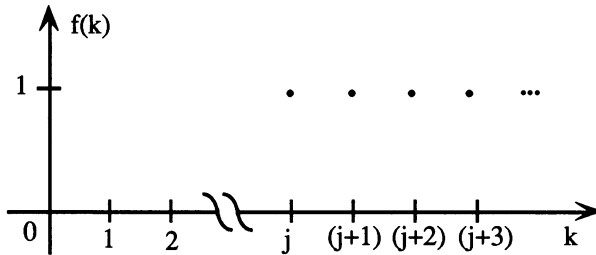


Fig. 3.1-5 The delayed unit-step function $f(k)=u(k-j)$.

Nonunity (arbitrary-size) step functions can be constructed by simply pre-multiplying the unit-step function by a scalar. Step functions can also be combined with other functions to synthesize desired types of discrete waveforms. For instance, the function shown in Fig. 3.1-6a can be formed by subtracting the delayed unit-step function $u(k-1)$, shown in Fig. 3.1-5, from the unit-step function $u(k)$, shown in Fig. 3.1-4.

The function in Fig. 3.1-6a can be expressed analytically as

$$\delta(kT) = \delta(k) = \delta_k = \begin{cases} 1 & \text{for } k=0 \\ 0 & \text{for } k \neq 0 \end{cases} \quad (3.1-9)$$

Eq. (3.1-9) defines the **discrete unit-impulse** function, also known as the **Kronecker delta**, or just **delta**, or **unit** function. This sequence is zero everywhere except at the discrete time $k=0$, where its value is 1.

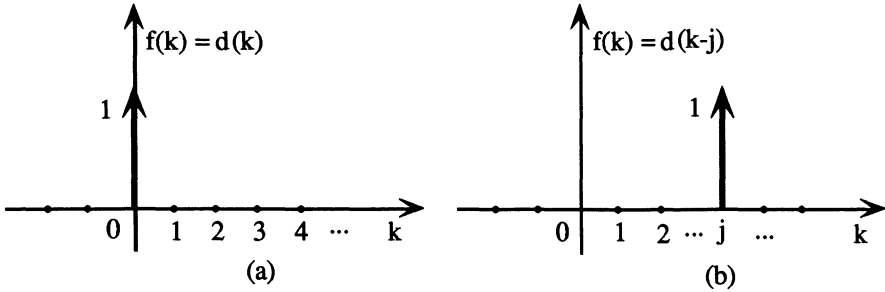


Fig. 3.1-6 The unit-impulse function (a) and the delayed unit-impulse function (b).

A unit-impulse function delayed by j sampling times is shown in Fig. 3.1-6b. The delayed unit-impulse function is defined by

$$\delta(k-j) = \begin{cases} 1 & \text{for } k=j \\ 0 & \text{for } k \neq j \end{cases} \quad (3.1-10)$$

It follows that a discrete function can be expressed as the sum of discrete impulse functions. For instance, the function given by

$$f(k) = \delta(k) + \delta(k-1) + \delta(k-2) + \dots = \sum_{j=0}^{\infty} \delta(k-j) \quad (3.1-11)$$

represents the **unit-impulse train**, shown in Fig. 3.1-7.

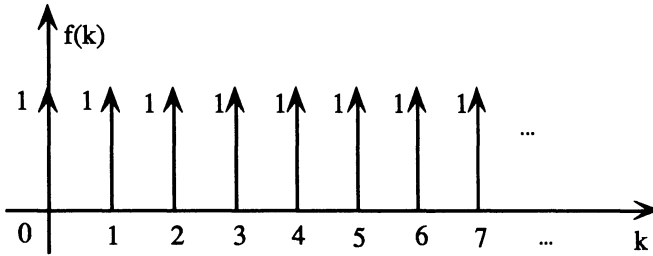


Fig. 3.1-7 The unit-impulse train defined by $f(k) = \sum_{j=0}^{\infty} \delta(k-j)$

Another singularity function is the **discrete unit-ramp** function, defined by

$$f(kT) = f(k) = \begin{cases} k & \text{for } k \geq 0 \\ 0 & \text{for } k < 0 \end{cases} \quad (3.1-12)$$

and shown in Fig. 3.1-8a. The above description can be expressed analytically as

$$f(k) = ku(k) \quad (3.1-13)$$

where $u(k)$ is the unit-step function. A unit-ramp function delayed by j sampling periods is written as

$$f(k) = (k-j)u(k-j) \quad (3.1-14)$$

and is shown in Fig. 3.1-8b, for $j=1$. Parabolic singularity functions as well as other functions can be generated by simply replacing the continuous time variable t by the discrete time variable k (or kT) in the appropriate expression.

There are several methods available for determining time or frequency response characteristics for linear, time-invariant discrete systems. For instance, a single-input-single-output (SISO) discrete system can be represented in the time domain by a difference equation, or by a set of discrete state space equations. On the other hand, in the frequency domain, such a system can be represented by its discrete transfer function. Appropriate techniques and tools, such as the digital computer, can then be applied to analyze discrete systems from their models.

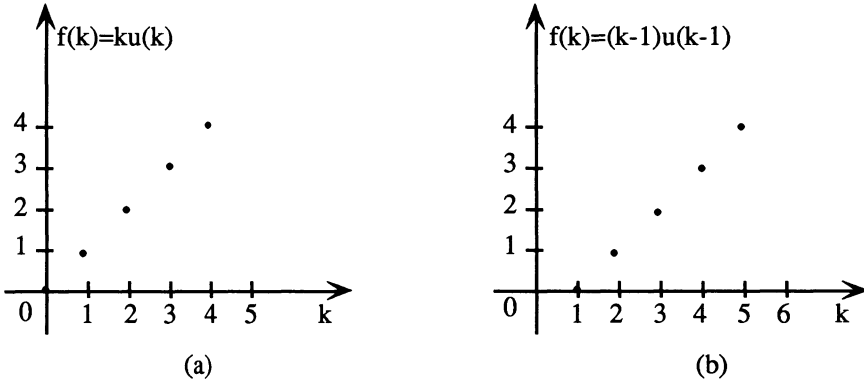


Fig. 3.1-8 Unit-slope ramp function (a) and delayed ramp functions (b).

3.2 DIFFERENCE EQUATIONS

The theory of linear difference equations closely parallels the theory of linear differential equations. The major distinction between these two mathematical models is that a differential equation expresses a relationship between continuous input and continuous output functions, whereas a **difference equation expresses a relationship between discrete input and discrete output functions (sequences)**.

The difference equation model of an n th-order, discrete system with a single input $x(kT)$ and a single output $y(kT)$ may be written as

$$\begin{aligned} y[(k+n)T] + b_1y[(k+n-1)T] + \dots + b_ny(kT) = \\ a_0x[(k+n)T] + a_1x[(k+n-1)T] + \dots + a_nx(kT) \end{aligned} \quad (3.2-1)$$

where the right-hand side represents the forcing function, presumably known. For notational convenience, Eq. (3.2-1) may be expressed in the form

$$y(k+n) + b_1 y(k+n-1) + \dots + b_n y(k) = a_0 x(k+n) + a_1 x(k+n-1) + \dots + a_n x(k) \quad (3.2-2)$$

Using the summation notation,

$$y(k+n) = \sum_{i=0}^n a_i x(k+n-i) - \sum_{i=1}^n b_i y(k+n-i) \quad (3.2-3)$$

An alternative representation of the n th-order difference equation results when k is replaced by $(k-n)$. Eq. (3.2-2) then becomes

$$y(k) + b_1 y(k-1) + \dots + b_n y(k-n) = a_0 x(k) + a_1 x(k-1) + \dots + a_n x(k-n) \quad (3.2-4)$$

The difference equation model of a system may be given or it may have to be derived from another type of model, such as a differential equation. Like differential equations, difference equations can be classified as homogeneous or nonhomogeneous, time-invariant or time-variant, and linear or nonlinear. These concepts are illustrated next by means of examples.

Example 3.2-1 First-Order Difference Equation

Consider the first-order differential equation

$$y'(t) + ay(t) = Kax(t)$$

and assume that we are interested in evaluating its solution at the sampling instants defined by

$$t = kT, \text{ for } k = 0, 1, 2, \dots$$

From elementary calculus, the first derivative of $y(t)$ can be approximated using the forward difference. That is, $y'(t)$ can be evaluated at the sampling time $t=kT$ by

$$y'(t) \Big|_{t=kT} = y'(kT) = y'(k) \approx \frac{y[(k+1)T] - y(kT)}{T} = \frac{y(k+1) - y(k)}{T}$$

The derivative approximation improves as the sampling period T becomes smaller. Substituting this into the differential equation gives

$$\frac{y(k+1) - y(k)}{T} + ay(k) = Kax(k)$$

After rearranging, we get

$$y(k+1) + (aT-1)y(k) = KaTx(k)$$

This is a first-order, time-invariant, linear difference equation. For small enough T , this equation can be used to approximate the solution of the original differential equation at the sampling instants. Once the initial condition $y(0)$ and the input sequence $x(k)$ are known, the difference equation can be used to determine $y(k)$ from the relationship

$$y(k+1) = (1-aT)y(k) + KaTx(k), \text{ for } k \geq 0$$

Example 3.2-2 Second-Order Difference Equations

Consider the second-order differential equation

$$y''(t) + 2\zeta\omega_n y'(t) + \omega_n^2 y(t) = \omega_n^2 x(t)$$

To approximate the second derivative, let

$$y'(k) \approx \frac{y(k+1) - y(k)}{T} = q(k)$$

Using the approximation for the first derivative, $y''(k)$ can be represented by

$$y''(k) \approx q'(k) \approx \frac{q(k+1) - q(k)}{T}$$

Substituting from above gives

$$y''(k) \approx \frac{y(k+2) - 2y(k+1) + y(k)}{T^2}$$

Substituting $y'(k)$ and $y''(k)$ in the differential equation, we obtain the second-order, linear, time-invariant difference equation

$$y(k+2) = (2-2\zeta\omega_n T)y(k+1) + (2\zeta\omega_n T-1-\omega_n^2 T^2)y(k) + \omega_n^2 T^2 x(k)$$

Example 3.2-3 Loan Amortization Algorithm

Consider the problem of formulating a mathematical model used for amortizing a loan, that is, repaying a loan by periodic payments. Under this plan, a certain amount of the payment is for interest for that period, while the remaining portion of the payment is for reducing the outstanding balance.

Assume, for instance, that we wish to amortize a loan amount of A dollars, and that the interest charges are compounded monthly at the rate R percent. Then, for a monthly payment P dollars, the monthly interest charge is given by AR . After the first payment is made, the outstanding balance B is given by

$$B = A + AR - P = A(1+R) - P$$

For the next monthly period, A is replaced by B to calculate a new B , and so forth. To generalize the amortization method, represent the outstanding balance after the k th payment by $y(k)$, and the amount of the k th payment by $x(k)$. The difference equation then is

$$y(k) = y(k-1)(1+R) - x(k)$$

This difference equation is first-order, time-invariant, constant-coefficient, linear, and nonhomogeneous. It is solved later using transform methods.

Example 3.2-3 A Square Root Algorithm

Consider the problem of developing an algorithm for finding the square root of a number using Newton's iterative method. For a function $y(x)$, Newton's method is defined by

$$x(k+1) = x(k) - \frac{y[x(k)]}{y'[x(k)]}, \quad k=0, 1, 2, \dots$$

For the square root of a number A the desired result is $x(k) = \sqrt{A}$. The relevant polynomial equation is

$$y(x) = x^2 - A = 0$$

where A is an arbitrary positive number. Differentiating gives

$$y'(x) = 2x$$

Substituting, we obtain the square root algorithm

$$x(k+1) = x(k) - \frac{[x^2(k) - A]}{2x(k)} = 0.5[x(k) + \frac{A}{x(k)}] \quad ; \text{ for } k=0, 1, 2, \dots$$

Note that it is necessary to supply an initial approximation for the term $x(0)$. The resulting equation is a first-order, constant-coefficient, nonhomogeneous difference equation. It is also nonlinear, because not all of the dependent terms $x(k)$ are of first degree.

3.3 TIME RESPONSE USING CLASSICAL METHODS

To simplify the notation, define a **forward** or **advance operator** represented by the symbol ' z ' such that

$$zy(k) = y(k+1), \quad z^2y(k) = y(k+2), \text{ etc.} \quad (3.3-1)$$

Using this operator, the n th-order difference equation can be written as

$$(z^n + b_1z^{n-1} + \dots + b_n)y(k) = (a_0z^n + a_1z^{n-1} + \dots + a_n)x(k) \quad (3.3-2)$$

The corresponding **backward** or **delay operator** z^{-1} is defined such that

$$z^{-1}y(k+1) = y(k), \quad z^{-2}y(k+2) = y(k), \text{ etc.} \quad (3.3-3)$$

Using the delay operator, the n th-order difference equation becomes

$$(1 + b_1z^{-1} + \dots + b_nz^{-n})y(k) = (a_0 + a_1z^{-1} + \dots + a_nz^{-n})x(k) \quad (3.3-4)$$

Note that the latter form can also be obtained from Eq. (3.3-2) by dividing both sides by z^n , the n th-power of z .

Solving an n th-order difference equation means finding a function $y(k)$ which, when substituted into the original equation, satisfies it. As with differential equations,

one of the standard classical methods for solving difference equations is the **method of undetermined coefficients**. This method is used to solve homogeneous as well as nonhomogeneous difference equations.

3.3-1 Solution of Homogeneous Difference Equations

An n th-order homogeneous difference equation can be written as

$$(z^n + b_1 z^{n-1} + b_2 z^{n-2} + \dots + b_{n-1} z + b_n) y(k) = 0 \quad (3.3-5)$$

Since $y(k)$ is the required nontrivial solution, it cannot be zero. Hence, the above equation implies that

$$z^n + b_1 z^{n-1} + b_2 z^{n-2} + \dots + b_{n-1} z + b_n = 0 \quad (3.3-6)$$

This is the **characteristic equation** of the difference equation and, thus, of the system the difference equation represents. The **characteristic zeros** or **eigenvalues** may be real and distinct, real and repeated, or complex.

If the characteristic equation contains a zero of magnitude larger than 1, then the complementary or homogeneous solution has a term which increases with time, and the system is unstable. Zeros of magnitude less than 1, on the other hand, lead to terms that decay with time and, thus, to a stable system.

If the characteristic equation has a pair of complex zeros with magnitudes of 1, then, the complementary solution is oscillatory and the system is said to be marginally or critically stable. If such zeros are also repeated, the system is unstable.

If the characteristic zeros are plotted in a complex z -plane, all the zeros must be inside the unit circle for a stable system. Any zero outside the unit circle or repeated zeros on the unit circle indicate an unstable system; simple complex zeros on the unit circle imply a critically-stable system. The previous concepts are demonstrated by examples.

Example 3.3-1 First-Order Equation

Consider the first-order, homogeneous difference equation

$$y(k+1) - ay(k) = 0$$

where a is an arbitrary constant.

Using the ' z ' operator notation defined earlier we get

$$(z - a)y(k) = 0$$

The characteristic equation is

$$z - a = 0$$

The characteristic zero is

$$z_1 = a$$

and the solution of the first-order difference equation is given by

$$y(k) = y_h(k) = k_1(z_1)^k = k_1(a)^k$$

where k_1 depends on an initial condition, usually $y(0)$.

To verify that this is the solution of the given difference equation, we first obtain

$$y(k+1) = k_1(a)^{k+1}$$

Substituting $y(k)$ and $y(k+1)$ in the original equation, we get

$$k_1(a)^{k+1} - ak_1(a)^k = ak_1(a)^k - ak_1(a)^k = 0$$

which verifies that the assumed $y(k)$ is the correct solution.

Example 3.3-1a

For a specific case, consider the first-order difference equation

$$y(k+1) - 2y(k) = 0$$

The characteristic equation is

$$z - 2 = 0$$

and the characteristic zero is

$$z_1 = 2$$

a real number. Thus, the corresponding solution is

$$y(k) = k_1(2)^k$$

The constant k_1 depends on an initial condition, usually $y(0)$.

Note that the solution of the difference equation increases with time, which is a characteristic of an unstable system. This is expected, of course, since the characteristic zero (2) of the system is larger than 1, and the zero lies outside the unit circle in the complex z -plane.

Example 3.3-2

Consider the first-order difference equation

$$y(k+1) - 0.60653y(k) = 0$$

along with the initial condition $y(0) = 1$. The characteristic equation is

$$z - 0.60653 = 0$$

and the characteristic zero is

$$z_1 = 0.60653$$

This leads to the homogeneous or complementary solution

$$y_h(k) = k_1(0.60653)^k$$

Applying the initial condition $y(0) = 1$ gives

$$y_h(0) = 1 = k_1$$

Thus, the solution is

$$y(k) = y_h(k) = (0.60653)^k$$

Using the identity

$$A = e^{\ln A}$$

we can express the solution $y(k)$ in the equivalent form

$$y(k) = (e^{\ln 0.60653})^k = e^{-0.5k}$$

Recall that k represents kT which in turn represents the continuous time variable t at the sampling instants. Thus, the above solution can be written in the form

$$y(kT) = y(k) = e^{-0.5t/T} ; \text{ for } t = 0, T, 2T, \dots$$

For instance, if $T = 0.1$ sec, the solution of the difference equation is equivalent to the continuous function

$$y(t) = e^{-5t}, \text{ for } t = 0, 0.1, 0.2, \dots$$

The above exponential function represents the solution of the first-order differential equation considered in Example 2.3-1.

Second-Order Difference Equation. Second-order difference equations lead to second-degree characteristic equations. The two characteristic roots can be real and distinct, real and repeated (equal), or complex conjugates. The respective solutions to these cases are considered next.

Case 1. Real and Distinct Roots

With the real and distinct characteristic roots $z_1, z_2, \dots, z_n$, the complementary solution of an n th-order difference equation is expressed by

$$y(k) = k_1 z_1^k + k_2 z_2^k + \dots + k_n z_n^k \quad (3.3-7)$$

where the coefficients $k_1, k_2, \dots, k_n$ are arbitrary constants, dependent on initial conditions.

Example 3.3-3

Solve the 2nd-order homogeneous difference equation

$$y(k+2) - 0.3y(k+1) - 0.1y(k) = 0$$

The characteristic equation is

$$z^2 - 0.3z - 0.1 = 0$$

with real and distinct characteristic zeros given by

$$z_1 = 0.5, \text{ and } z_2 = -0.2$$

The solution is

$$y(k) = k_1(0.5)^k + k_2(-0.2)^k$$

where k_1 and k_2 are arbitrary constants determined from the system initial conditions, usually $y(0)$ and $y(1)$. The solution expressed can be verified by substituting it into the original difference equation. An equivalent form of the solution is given by

$$y(k) = k_1(0.5)^k + k_2(0.2)^k \cos k\pi$$

This form is valid since

$$(-0.2)^k = (-1)^k(0.2)^k = e^{\pm jk\pi}(0.2)^k = (0.2)^k \cos k\pi, \text{ for } k = 0, 1, 2, \dots$$

Case 2. Real and Equal Roots

If the characteristic roots are real and equal, that is, if

$$z_1 = z_2 = \dots = z_n = z \tag{3.3-8}$$

then the complementary solution is

$$y(k) = k_1 z^k + k_2 k z^k + \dots + k_n k^{n-1} z^k = z^k (k_1 + k_2 k + \dots + k_n k^{n-1}) \tag{3.3-9}$$

where the coefficients $k_1, k_2, \dots, k_n$ depend on initial conditions.

Example 3.3-4

Solve the difference equation

$$y(k+2) - y(k+1) + 0.25y(k) = 0$$

The characteristic equation is

$$z^2 - z + 0.25 = 0$$

with real and equal characteristic zeros given by

$$z_1 = z_2 = 0.5$$

The complementary solution then is

$$y(k) = k_1(0.5)^k + k_2 k(0.5)^k$$

Case 3. Complex-Conjugate Roots

Assume that the complex conjugate roots are given by

$$z_1 = a + jb, \text{ and } z_2 = a - jb \quad (3.3-10)$$

or, using equivalent exponential notation,

$$z_1 = a + jb = Me^{j\theta}, z_2 = a - jb = Me^{-j\theta} \quad (3.3-11)$$

$$\text{where } M = \sqrt{a^2 + b^2}, \text{ and } \theta = \tan^{-1}(b/a) \quad (3.3-12)$$

The solution to the difference equation is

$$\begin{aligned} y(k) &= k_1 z_1^k + k_2 z_2^k = M^k (k_1 e^{j\theta k} + k_2 e^{-j\theta k}) \\ &= M^k [k_1 (\cos\theta k + j\sin\theta k) + k_2 (\cos\theta k - j\sin\theta k)] \\ &= M^k (A \cos\theta k + B \sin\theta k) \end{aligned} \quad (3.3-13)$$

where A and B are constants whose values depend on the initial conditions.

Example 3.3-5

Solve the difference equation

$$y(k+2) - y(k+1) + 0.5y(k) = 0$$

The characteristic equation is

$$z^2 - z + 0.5 = 0$$

and the complex conjugate characteristic zeros are

$$z_1 = 0.5 + j0.5, \text{ and } z_2 = 0.5 - j0.5$$

Since the magnitude of the zeros is given by

$$M = \sqrt{0.25 + 0.25} = 0.70711$$

and the angle is given by

$$\theta = \tan^{-1}(1) = 45^\circ = \frac{\pi}{4}$$

the complementary solution is

$$y(k) = (0.70711)^k [A \cos(k\pi/4) + B \sin(k\pi/4)]$$

The result can be verified by substituting it into the original equation.

Higher-Order Difference Equations. For higher-order difference equations, the characteristic zeros are grouped as real and distinct, real and equal, and complex. The resulting solutions are then combined to determine the complementary solution.

Example 3.3-6

Solve the homogeneous difference equation whose characteristic zeros are

$$z_1 = -0.2, \quad z_2 = 0.2, \quad \text{and} \quad z_3 = 0.2$$

Assume that $y(0) = 0$, $y(1) = 1$, and $y(2) = 2$. The solution is given by

$$y(k) = k_1(-0.2)^k + k_2(0.2)^k + k_3k(0.2)^k$$

Applying the initial conditions results in the following system of equations

$$1.0k_1 + 1.0k_2 + 0.0k_3 = 0$$

$$-0.2k_1 + 0.2k_2 + 0.2k_3 = 1$$

$$0.04k_1 + 0.04k_2 + 0.08k_3 = 2$$

Solving for k_1 , k_2 , and k_3 gives $k_1 = 10$, $k_2 = -10$, and $k_3 = 25$. The solution of the difference equation then is

$$y(k) = 10(-0.2)^k - 10(0.2)^k + 25k(0.2)^k$$

3.3-2 Solution of Nonhomogeneous Difference Equations

The complete solution of a nonhomogeneous difference equation is composed of the complementary component $y_h(k)$ and the particular component $y_p(k)$. Thus, the solution of a nonhomogeneous difference equation can be written as

$$y(k) = y_h(k) + y_p(k) \quad (3.3-14)$$

The procedure for obtaining the particular solution of a difference equation parallels the procedure for obtaining the particular solution of a differential equation. The particular solution $y_p(k)$ is constructed from the forcing function and is substituted into the nonhomogeneous difference equation in order to evaluate the unknown constants. Once $y_p(k)$ is found, the constants associated with $y_h(k)$ can then be determined by applying the appropriate initial conditions to the solution $y(k)$. The method for solving nonhomogeneous difference equations is illustrated by the following examples.

Example 3.3-7

Suppose that a first-order system is modeled by the difference equation

$$y(k+1) - 0.60653y(k) = 0.39347x(k)$$

where $x(k)$ is the forcing function. The problem is to determine the unit-step response of the system assuming zero initial conditions; that is, find $y(k)$ if $y(0)=0$ and $x(k)=u(k)$.

The complementary solution is obtained from the characteristic equation

$$z - 0.60653 = 0$$

The characteristic zero is

$$z_1 = 0.60653$$

and the complementary solution

$$y_h(k) = k_1(0.60653)^k u(k)$$

To find the particular solution, observe that the right-hand side consists of the constant term $0.39347u(k) = 0.39347$ for $k \geq 0$. Therefore, the root that contributes this term has a value of 1. If we call this root z'_1 ,

$$y_p(k) = A(z'_1)^k = A$$

where A is an arbitrary constant. To determine A , substitute $y_p(k)$ into the original equation. This gives

$$A - 0.60653A = 0.39347, \text{ or } A = 1$$

Thus, the total solution becomes

$$y(k) = k_1(0.60653)^k + 1$$

The zero initial condition is applied at this point in order to find k_1 . From $y(0)=0=k_1+1$, we obtain $k_1=-1$. The complete solution of the difference equation then becomes

$$y(k) = [1 - (0.60653)^k]u(k)$$

or equivalently,

$$y(k) = (1 - e^{-0.5k})u(k)$$

If we replace the discrete time k by t/T in the above equation and assume that $T=0.1$ sec, the unit-step response of the discrete system can be expressed as

$$y(t) = (1 - e^{-5t})u(t), \text{ for } t = 0, 0.1, 0.2, \dots$$

Example 3.3-8

Consider the system described by the difference equation

$$y(k+1) - 0.60653y(k) = x(k)$$

where $x(k)$ represents the forcing function. Assuming the system is initially at rest, determine the unit-impulse response, that is, find $y(k)$ when $y(0)=0$ and $x(k) = \delta(k)$. Note that $\delta(k)=0$ for $k \geq 1$, so that the unit-impulse response is the complementary solution

$$y(k) = k_1(0.60653)^k ; \text{ for } k \geq 1$$

To evaluate the arbitrary constant k_1 , let $k=0$ in the original equation. This results in

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

From $y(k)$, we obtain

$$y(1) = 1 = k_1(0.60653) , \text{ or } k_1 = 1.64872$$

Thus, the unit-impulse response of the system is

$$y(k) = (0.60653)^{-1}(0.60653)^k ; \text{ for } k \geq 1$$

Equivalently,

$$y(k) = (0.60653)^{k-1}u(k-1) = e^{-0.5(k-1)}u(k-1) ; \text{ for } k \geq 1$$

Example 3.3-9

Solve the difference equation

$$y(k+1) - y(k) = (1 + 2^k)u(k)$$

with the initial condition $y(0) = 0$. From the characteristic equation

$$z-1=0$$

we obtain the characteristic zero

$$z_1=1$$

This implies the complementary solution

$$y_h(k) = k_1(1)^k = k_1$$

The particular solution due to each term in the forcing function is determined next. That is, the constant term 1, or $u(k)$, is due to the real root

$$z'_1 = 1$$

which is also repeated because we already have the root $z_1=1$ in the characteristic equation. Thus, the component contributed to the particular solution by the term 1 is Ak , where A is an arbitrary constant. The term 2^k is due to the real root

$$z'_2 = 2$$

This root contributes the component $B2^k$ in the particular solution; B is also an arbitrary constant. Thus, the particular solution becomes

$$y_p(k) = Ak + B2^k$$

where A and B are determined by substituting $y_p(k)$ into the original difference equation. Thus,

$$A + Ak + 2B2^k - Ak - B2^k = 1 + 2^k$$

Equating coefficients, we obtain

$$A = 1, \text{ and } B = 1$$

This leads to the particular solution

$$y_p(k) = k + 2^k$$

The solution is

$$y(k) = y_h(k) + y_p(k) = k_1 + k + 2^k$$

Applying the initial condition $y(0) = 0$ results in the value for the coefficient $k_1 = -1$. Finally, the completely defined solution becomes

$$y(k) = (k + 2^k - 1) u(k)$$

The answer can be verified by substituting into the original difference equation.

Example 3.3-10

To illustrate the interrelationship between differential and difference equations, consider a discrete system described by the second-order difference equation

$$y(k+2) - 1.7736y(k+1) + 0.81873y(k) = 0.02332x(k+1) + 0.02181x(k)$$

Determine the response of the system to the input $x(k) = u(k)$; assume that $y(k) = 0$ for $k \leq 0$. The characteristic equation is

$$z^2 - 1.7736z + 0.81873 = 0$$

The complex characteristic zeros are

$$z_{1,2} = 0.8868 \pm j0.17976 = 0.90484e^{\pm j0.2}$$

The complementary solution then can be determined as

$$y_h(k) = (0.90484)^k (k_1 \cos 0.2k + k_2 \sin 0.2k) = e^{-0.1k} (k_1 \cos 0.2k + k_2 \sin 0.2k)$$

Since the right-hand side or forcing function is constant, the particular solution is also a constant. That is,

$$y_p(k) = A$$

Since $x(k) = 1$ for $k \geq 0$, substituting $y_p(k)$ into the original equation gives

$$A - 1.7736A + 0.81873A = (0.02332 + 0.02181) = 0.04513, \text{ or } A = 1$$

Thus, the solution is

$$y(k) = 1 + e^{-0.1k}(k_1 \cos 0.2k + k_2 \sin 0.2k)$$

To find the k's, we must now apply appropriate initial conditions (two of them).

It is given that $y(0)=0$. To obtain a second initial condition, substitute $k=-1$ into the original difference equation. This gives

$$y(1) + 0 + 0 = 0.02332 + 0, \text{ or } y(1) = 0.02332$$

Applying the initial conditions $y(0)=0$ and $y(1)=0.02332$, results in the system of equations

$$y(0) = 0 = 1 + k_1, \text{ or } k_1 = -1$$

and $y(1) = 0.02332 = 1 + 0.90484(k_1 \cos 0.2 + k_2 \sin 0.2)$

Substituting the value of k_1 from the first equation into the second equation and solving for k_2 yields $k_2=-0.5$. Thus, the desired solution of the difference equation is

$$y(k) = 1 - e^{-0.1k}(\cos 0.2k + 0.5 \sin 0.2k)$$

Assuming that $T=0.1$ sec, and since $k = t/T = 10t$, the solution can be written in the form

$$y(t) = [1 - e^{-t}(\cos 2t + 0.5 \sin 2t)]u(t), \text{ for } t = 0, 0.1, 0.2, \dots$$

This $y(t)$ matches (at the sampling instants) the solution of the differential equation

$$y''(t) + 2y'(t) + 5y(t) = 5u(t)$$

already solved in Example 2.3-4.

3.4 SAMPLING THEORY

In most applications, a continuous signal $x(t)$ is sampled periodically, every T seconds, to generate a discrete signal denoted by $x^*(t)$. The discrete signal is subsequently fed into a linear plant or transfer function. This process is called **uniform sampling** and can be represented by a switch, called a **sampler**, that closes every T sec for a duration of h sec, as shown in Fig. 3.4-1.

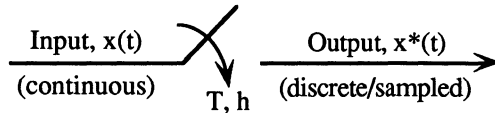


Fig. 3.4-1 Sampler switch which closes every T sec for a duration of h sec.

The output of the sampler can be expressed as the product of the input $x(t)$ and a periodic pulse train $p(t)$, that is,

$$x^*(t) = x(t) p(t) \quad (3.4-1)$$

Corresponding waveforms are shown in Fig. 3.4-2. The amplitudes of the pulses are modulated by $x(t)$. The width h of the pulses is assumed to be relatively small compared

to the sampling time T , as well as the time constants of the transfer function that follows the sampler. In other words, the transfer function must not be able to distinguish between an ideal impulse and a pulse of duration h . This type of sampling is referred to as **nonideal**, because the pulses and thus the samples are of finite width h .

If we assume that, in the limit, the pulse width approaches zero, the sampler output becomes a train of impulse or Dirac delta functions. This type of uniform sampling is referred to as **ideal impulse sampling** and is represented graphically in Fig. 3.4-3. In this case, the sampled signal is represented by a train of impulse functions whose areas correspond to the sampled values of the signal. The impulse train is defined by

$$p(t) = \sum_{k=0}^{\infty} \delta(t-kT) \quad (3.4-2)$$

From Eq. (3.4-1), we obtain

$$x^*(t) = x(t) \sum_{k=0}^{\infty} \delta(t-kT) = \sum_{k=0}^{\infty} x(kT) \delta(t-kT) \quad (3.4-3)$$

since the impulse function $\delta(t-kT)$ is zero everywhere, except at $t=kT$.

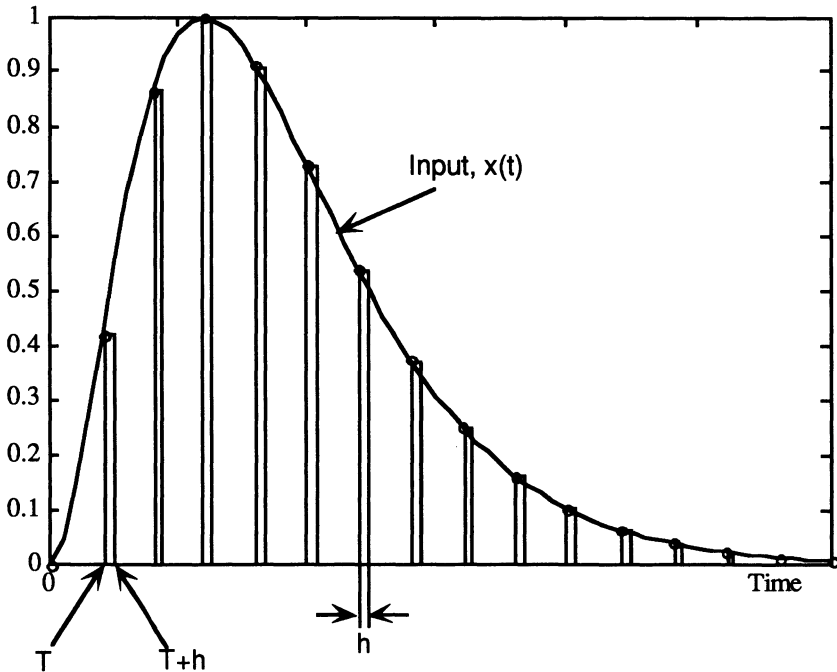


Fig. 3.4-2 Generation of a sampled signal using nonideal sampling ($h \neq 0$). Sampler input and sampler output (pulses) signals.

Even though in practice ideal impulse sampling is not possible, it can be approximated closely and leads to considerably simpler methods of mathematical analysis

and representation of systems. Taking the Laplace transform on both sides of Eq. (3.4-3) results in the input transform

$$X^*(s) = \int_0^{\infty} \sum_{k=0}^{\infty} x(kT) \delta(t-kT) e^{-st} dt \quad (3.4-5)$$

Interchanging integration and summation yields

$$X^*(s) = \sum_{k=0}^{\infty} x(kT) \int_0^{\infty} \delta(t-kT) e^{-st} dt \quad (3.4-6)$$

Using the shifting property of the impulse function, we obtain

$$X^*(s) = \sum_{k=0}^{\infty} x(kT) e^{-ksT} \quad (3.4-7)$$

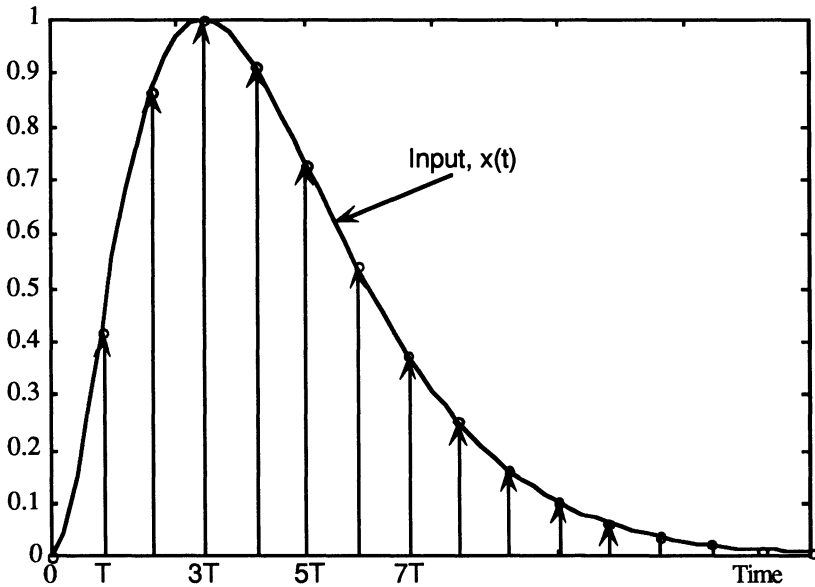


Fig. 3.4-3 Generation of a sampled signal $x^*(t)$ using ideal impulse sampling. Sampler input, and sampler output (impulses) signals.

The above expression can be expanded in the form

$$X^*(s) = x(0) + x(T)e^{-sT} + x(2T)e^{-2sT} + \dots \quad (3.4-8)$$

Taking the inverse transform of $X^*(s)$, we get

$$x^*(t) = x(0)\delta(t) + x(T)\delta(t-T) + x(2T)\delta(t-2T) + \dots \quad (3.4-9)$$

Substituting the continuous frequency variable s by $j\omega$ in Eq. (3.4-7) gives

$$X^*(j\omega) = \sum_{k=0}^{\infty} x(kT)e^{-j\omega kT} \quad (3.4-10)$$

If the sampling time is T , the sampling frequency ω_s is

$$\omega_s = \frac{2\pi}{T} \text{ rad/sec} \quad (3.4-11)$$

If m is an integer, the frequency response is given by

$$X^*(j\omega + jm\omega_s) = \sum_{k=0}^{\infty} x(kT)e^{-jkT(\omega + m\omega_s)} = \sum_{k=0}^{\infty} x(kT)e^{-jk\omega T}e^{-jk\omega_s mT} \quad (3.4-12)$$

where ω_s represents the sampling frequency in rad/sec, and $m\omega_s$ represents the number of sampling frequencies from ω . Since

$$\omega_s T = \frac{2\pi}{T}T = 2\pi \quad (3.4-13)$$

the second exponential term can be written as

$$e^{-jk\omega_s mT} = e^{-jkm2\pi} = 1 \angle 0^\circ \quad (3.4-14)$$

Thus, from Eq. (3.4-12),

$$X^*(j\omega + jm\omega_s) = \sum_{k=0}^{\infty} x(kT)e^{-jk\omega T} \quad (3.4-15)$$

or

$$X^*(j\omega + jm\omega_s) = X^*(j\omega) \quad (3.4-16)$$

Frequency Spectrum. Eq. (3.4-16) shows that the frequency response, also referred to as the **frequency spectrum**, of the sampled function $X^*(s)$ is a periodic function with period ω_s . That is, the sampling process in the time domain has resulted in a periodic function in the frequency domain.

Another important result can be obtained from the definition of the Laplace transform. That is, if $x(t)$ is a continuous signal, then

$$X(s) = \int_0^{\infty} x(t)e^{-st}dt \quad (3.4-17)$$

If the continuous signal is sampled with a sufficiently small sampling time T , the integral operation can be approximated by the summation as

$$X(s) = \sum_{k=0}^{\infty} x(kT)e^{-ksT} = T \sum_{k=0}^{\infty} x(kT)e^{-ksT} = TX^*(s) \quad (3.4-18)$$

From Eq. (3.4-16), we can write

$$X^*(j\omega) = \frac{1}{T} \sum_{m=0}^{\infty} X(j\omega + jm\omega_s) \quad (3.4-19)$$

which implies that the frequency spectrum of the sampled function $X^*(s)$ can be obtained from the frequency spectrum of the continuous function $X(s)$.

Assume that the continuous signal is **band-limited**, that is, assume that

$$|X(j\omega)| = 0 \text{ ; for } \omega > \omega_h \text{ and } \omega < -\omega_h \quad (3.4-20)$$

where ω_h is the highest possible frequency present in $x(t)$, as shown in Fig. 3.4-4. From Eq. (3.4-19), we see that additional components of the amplitude spectrum of the sampler output can be obtained using the expression

$$X^*(j\omega) = \frac{1}{T} X(j\omega + jm\omega_s) \text{ ; for } m=0, 1, 2, \dots \quad (3.4-21)$$

as shown in Fig. 3.4-5. The spectrum component obtained by letting $m=0$ is referred to as the **primary component**, while the other components are known as the **complementary components** or **sidebands**. Thus, unlike the continuous signal, the sampled signal contains high frequency (complementary) components.

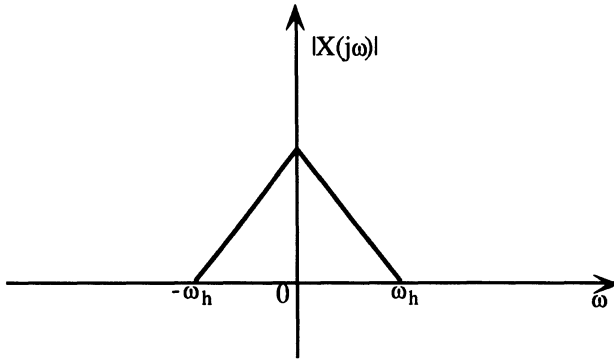


Fig. 3.4-4 Magnitude spectrum of the sampler input signal $x(t)$.

Shannon's Sampling Theorem. If the continuous signal $x(t)$ is to be reconstructed or recovered from its sampled form $x^*(t)$, it is necessary that none of the spectral components overlap with each other. This is possible only if the sampling rate ω_s is at least twice the highest frequency component ω_h . In other words,

$$\omega_s \geq 2\omega_h \quad (3.4-22)$$

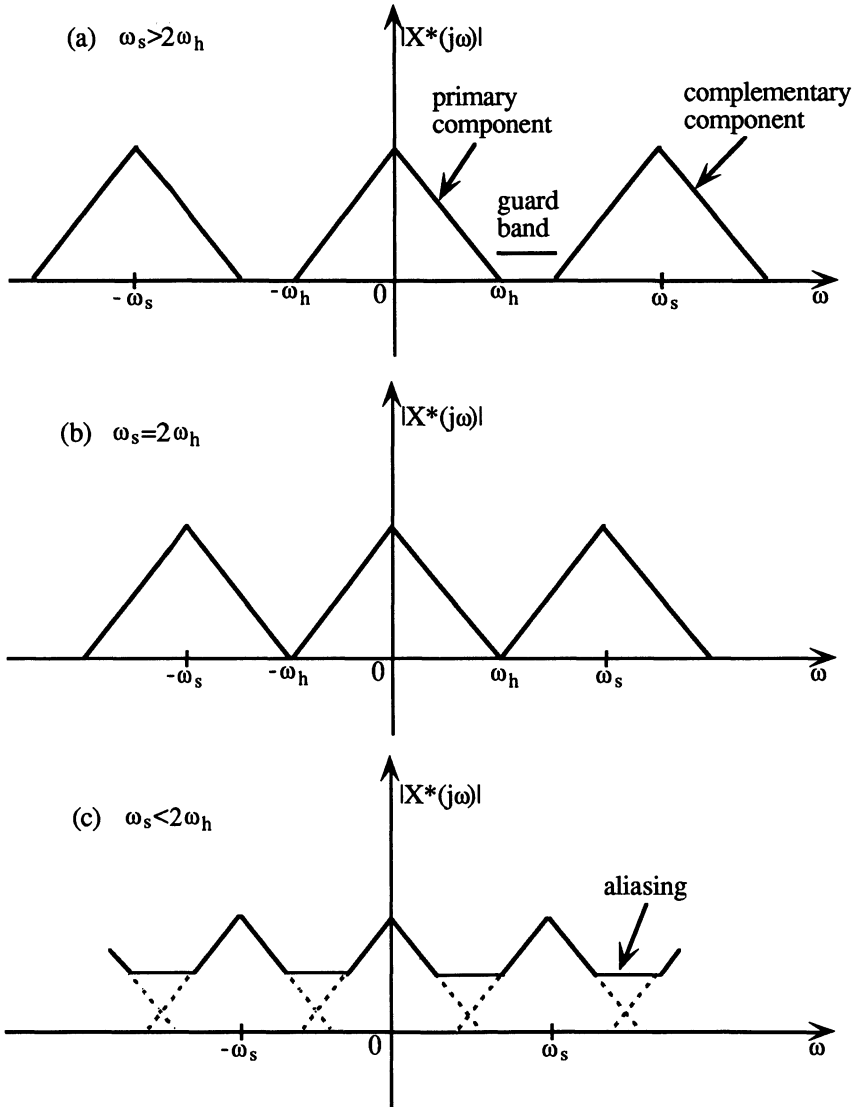


Fig. 3.4-5 Magnitude spectra of the sampler output $x^*(t)$ for the cases: (a) $\omega_s > 2\omega_h$, (b) $\omega_s = 2\omega_h$, and (c) $\omega_s < 2\omega_h$.

or, if the frequency is in Hz,

$$f_s \geq 2f_h \quad (3.4-23)$$

Eq. (3.4-22), or Eq. (3.4-23), is known as **Shannon's Sampling Theorem**. The highest (cutoff) frequency ω_h is known as the **Shannon or Nyquist frequency**. The **Nyquist rate (frequency)** is defined by

$$\omega_s = 2\omega_h \quad (3.4-24)$$

The Nyquist frequency is also referred to as the **folding frequency**, usually denoted by ω_o (rad/sec), or f_o (Hz). That is,

$$\omega_o = \frac{\omega_s}{2} = \pi f_s \text{ rad/sec} \quad (3.4-25)$$

$$\text{or} \quad f_o = \frac{f_s}{2} \text{ Hz} \quad (3.4-26)$$

The folding frequency ω_o represents the highest frequency that can be processed by a discrete system with sampling rate ω_s . Any frequency greater than ω_o will be "folded" and cannot be recovered.

Aliasing. If the sampling rate is too low and does not satisfy the sampling theorem, that is, if the signal is not band-limited, then, spectral overlap occurs and the original signal cannot be recovered from its samples. This spectral overlap phenomenon is commonly known as **aliasing**.

In the time domain, for instance, aliasing implies that two sinusoidal functions of different frequencies can have the same values at the sampling instants. This is illustrated by an example.

Example 3.4-1 Aliasing

Evaluate and plot the following sinusoidal functions $f(t)$ and $g(t)$ using the sampling time $T=0.025$ sec.

$$f(t) = \sin(2\pi 10t), \text{ and } g(t) = \sin(2\pi 50t)$$

According to the sampling theorem, the 50 Hz signal $g(t)$ should be sampled at a rate of at least 100 Hz; equivalently, the sampling time T should be less than 0.01 sec. The sampling time of $T=0.025$ sec (i.e., sampling rate of 40 Hz), violates the sampling theorem. The 50 Hz signal $g(t)$ sampled at 40 Hz results in an "alias" frequency of 10 Hz. Consequently, $g(t)$ cannot be recovered from its samples.

In practice, many signals may be assumed to be band-limited. However, if necessary, *one may filter the continuous signal before sampling to ensure that it meets the band-limited criterion.*

If no aliasing occurs, a continuous signal $x(t)$ can be reconstructed by passing the sampled signal $x^*(t)$ through a low-pass filter, designed to pass the primary component of the signal and to cut off somewhere between ω_h and $\omega_s - \omega_h$. That is, the low-pass filter will filter the high-frequency components introduced by the sampler or, equivalently, will "smooth" the time function. This is often necessary, because the narrow pulses provided by the sampling process contain very little power and, thus, cannot be used directly.

For instance, $x(t)$ can be reconstructed from $x^*(t)$, or $X(j\omega)$ can be recovered from $X^*(j\omega)$, using the ideal low-pass filter whose amplitude spectrum is defined by

$$|G(j\omega)| = 1 ; \text{ for } -\omega_h \leq \omega \leq \omega_h \quad (3.4-27)$$

$$\text{and} \quad |G(j\omega)| = 0 ; \text{ for } \omega < -\omega_h, \text{ and } \omega > \omega_h \quad (3.4-28)$$

Since it is not possible to build an ideal low-pass filter, it is desirable to allow for a so-called **guard band** between ω_h and $\omega_s - \omega_h$. This suggests that, for practical reasons, the sampling rate must be higher than the Shannon rate of $2\omega_h$ rad/sec, or $2f_h$ Hz. So-called "rules of thumb" suggest that the sampling rate be 10-20 times higher than ω_h .

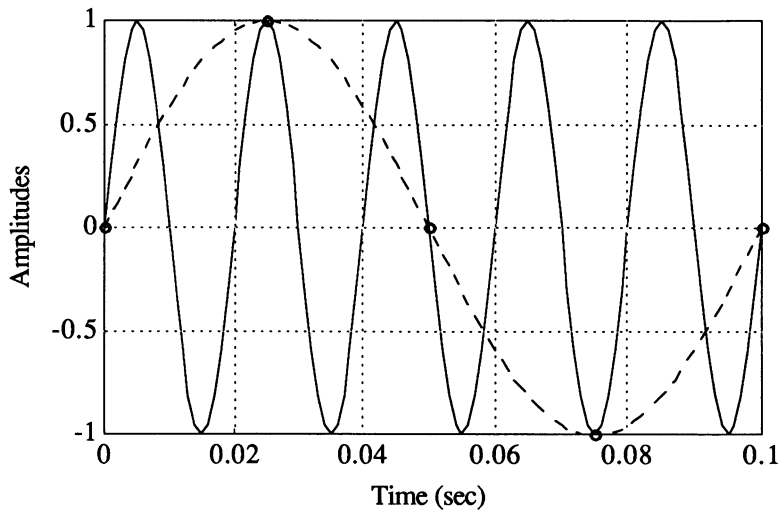


Fig. 3.4-6 Aliasing phenomenon: two sinusoids of different frequencies (10Hz and 50Hz) with identical sample values - denoted by the symbol 'o'.

3.4-1 The Zero-Order-Hold (ZOH) Element

The ideal sampler converts a signal from analog to digital form so that it may be processed by a digital computer. The computer output, however, must be converted back into analog form, if it is to be displayed or communicated to another analog system.

Reconstructing a continuous signal from its samples may be viewed as a problem of polynomial extrapolation. The simplest form of extrapolation involves a zero-order polynomial, which maintains or "holds" the last sample value constant until the next one occurs. This operation is known as **zero-order-hold (ZOH)**, and the device that performs the operation is the ZOH device or element.

Terms such as **boxcar** or **staircase generator** are also used to describe the ZOH element because of the form or shape of its output. The ZOH element is commonly implemented by a digital-to-analog converter (DAC); it can be represented in block diagram form as shown in Fig. 3.4-7.

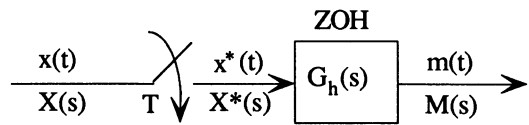


Fig. 3.4-7 Block diagram representation of the Zero-Order-Hold (ZOH) system.

The operation of the ZOH can be described mathematically by the expression

$$m(t) = \sum_{k=0}^{\infty} x(kT) \{u(t-kT) - u[t-(k+1)T]\} \quad (3.4-29)$$

To determine the ZOH transfer function $G_h(s)$, take the Laplace transform on both sides of the equation. The result is

$$\begin{aligned} M(s) &= \int_0^{\infty} \sum_{k=0}^{\infty} x(kT) \{u(t-kT) - u[t-(k+1)T]\} e^{-sT} dt \\ &= \sum_{k=0}^{\infty} x(kT) \left[\frac{e^{-skT} - e^{-(k+1)sT}}{s} \right] = \left[\frac{1 - e^{-sT}}{s} \right] \sum_{k=0}^{\infty} x(kT) e^{-skT} \\ &= \left[\frac{1 - e^{-sT}}{s} \right] X^*(s) \end{aligned} \quad (3.4-30)$$

The ZOH transfer function then becomes

$$G_h(s) = \frac{M(s)}{X^*(s)} = \frac{1 - e^{-sT}}{s} \quad (3.4-31)$$

The ZOH produces a "staircase" output when the input is a sequence of numbers. That is, the output is given by

$$m(t) = x^*(k), \text{ for } kT \leq t < (k+1)T \quad (3.4-32)$$

This performance resembles the operation of the DAC; thus, the ZOH can be thought of as a mathematical model for the DAC.

The frequency response characteristics of the ZOH can be determined by substituting $j\omega$ for s in the transfer function for $G_h(s)$. This gives

$$\begin{aligned} G_h(j\omega) &= \frac{1 - e^{-j\omega T}}{j\omega} = \frac{e^{-j\omega T/2} (e^{j\omega T/2} - e^{-j\omega T/2})}{j\omega} = \frac{2e^{-j\omega T/2} \sin(\omega T/2)}{\omega} \\ &= \frac{T e^{-j\omega T/2} \sin(\omega T/2)}{(\omega T/2)} = \left(\frac{2\pi}{\omega_s} \right) \frac{\sin(\pi\omega/\omega_s)}{(\pi\omega/\omega_s)} e^{-j\omega\pi/\omega_s} \end{aligned} \quad (3.4-33)$$

The magnitude and phase angle functions are given by

$$|G_h(j\omega)| = \left(\frac{2\pi}{\omega_s} \right) \frac{|\sin(\pi\omega/\omega_s)|}{(\pi\omega/\omega_s)} \quad (3.4-34)$$

and

$$f(\omega) = -\frac{\omega T}{2} = -\pi f T = -\frac{\omega\pi}{\omega_s} \quad (3.4-35)$$

Plots of these functions are shown in Fig. 3.4-8a and Fig. 3.4-8b, respectively. As can be seen from the magnitude plot, the ZOH behaves like a low-pass filter for low frequencies. The "lobes" are reduced considerably at high frequencies.

When modeling linear systems, the ZOH transfer function is often approximated by a first-order, low-pass filter with time constant of $T/2$ sec. That is,

$$G_h(s) = \frac{1-e^{-sT}}{s} \approx \frac{K(2/T)}{s+2/T} \quad (3.4-36)$$

The frequency characteristics of the ZOH element are considered again later in the section on the frequency response of discrete systems.

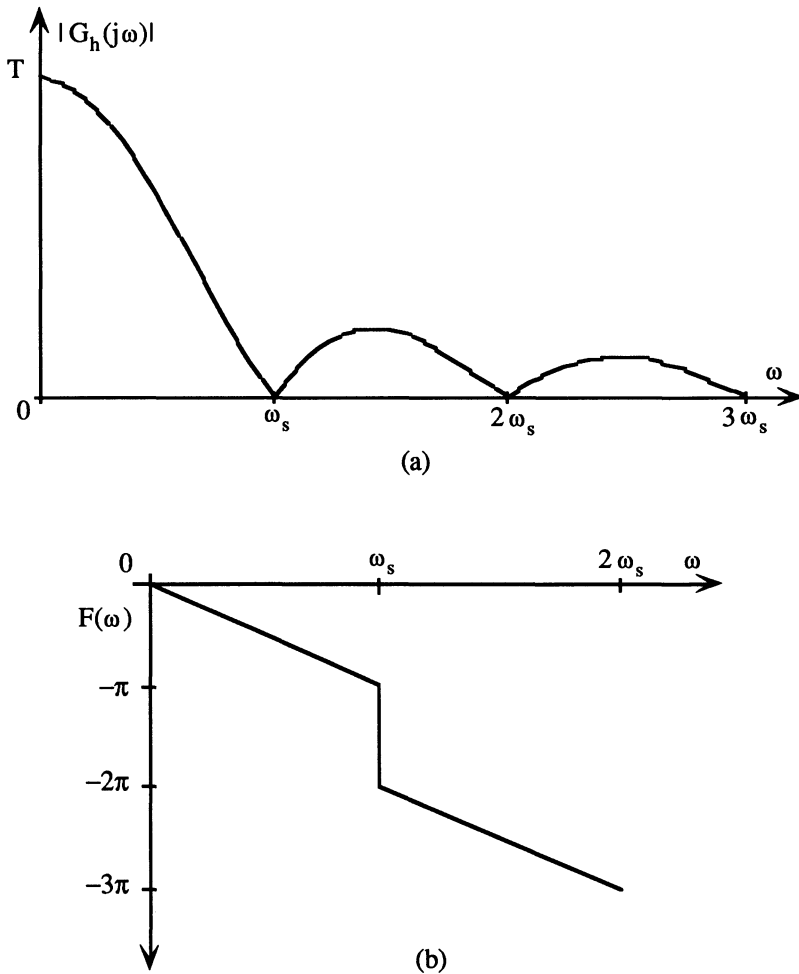


Fig. 3.4-8 Frequency characteristics of the ZOH: (a) magnitude and (b) phase angle.

3.5 THE $\mathcal{Z}$ TRANSFORM

The $\mathcal{Z}$ transform is an important operator used with linear, time-invariant, discrete systems, in the same manner that the Laplace transform is used with continuous systems.

To define the **one-sided or unilateral $\mathcal{Z}$ transform** of a continuous signal $f(t)$, or a discrete signal $f(k)$, we consider the Laplace transform expression for the sampler output given by Eq. (3.4-7). For the function $f(t)$, this is

$$F^*(s) = \sum_{k=0}^{\infty} f(k)e^{-skT} \quad ; \text{ for } k \geq 0 \quad (3.5-1)$$

Next, we introduce a complex variable 'z' defined by

$$z = e^{sT} \quad (3.5-2)$$

Solving for s, we obtain

$$s = \frac{1}{T} \ln z \quad (3.5-3)$$

The $\mathcal{Z}$ transform of $f(t)$ or $f(k)$ is defined by

$$\mathcal{Z}[f(k)] = F(z) = F^*(s) \Big|_{s=(1/T)\ln z} = \sum_{k=0}^{\infty} f(k)z^{-k} \quad (3.5-4)$$

where $\mathcal{Z}$ denotes the 'z' transform operator.

Expanding the summation form of $F(z)$, we can write the $\mathcal{Z}$ transform of $f(k)$ in the expanded form

$$F(z) = f(0) + f(1)z^{-1} + f(2)z^{-2} + \dots \quad (3.5-5)$$

The infinite series converges if $|f(k)|$ is restricted to finite values and if $|z^{-1}| < 1$; this is true inside the unit circle of the z-plane. The series form of $F(z)$ suggests that the inverse $\mathcal{Z}$ transform is given by

$$f^*(t) = \sum_{k=0}^{\infty} f(kT)\delta(t-kT) = f(0)\delta(t) + f(1)\delta(t-T) + f(2)\delta(t-2T) + \dots \quad (3.5-6)$$

The transformation expressed by Eq. (3.5-2) defines the relationship between the s and z variables and, thus, the mapping transformation between the complex s-plane and complex z-plane. Specifically, consider rewriting the transformation in the form

$$z = e^{sT} = e^{(\sigma + j\omega)T} = e^{\sigma T} e^{j\omega T} = e^{\sigma T} (\cos \omega T + j \sin \omega T) \quad (3.5-7)$$

From Eq. (3.5-7), it can be seen that the transformation maps the left-half of the s-plane ($\sigma < 0$) inside the unit circle in the complex z-plane ($|z| < 1$); also, the right-half of the s-plane ($\sigma > 0$) is mapped outside the unit circle in the z-plane ($|z| > 1$), and the $j\omega$ -axis of the s-plane ($\sigma = 0$) maps onto the circumference of the unit circle in the z-plane ($|z| = 1$). Finally, the origin of the s-plane ($\sigma = \omega = 0$) maps onto the point $z = 1$ in the z-plane.

The previous conclusions are drawn when the transformation of Eq. (3.5-2) is considered going from s to z . Care must be exercised, however, when considering the transformation from z to s , because this is a nonunique or multivalued transformation. Specifically, for every given point in the z -plane, there are an infinite number of corresponding values in the s -plane. Consequently, *two functions of time will have the same $\mathcal{Z}$ transform if and only if they are identical for all $t=kT$, where $k=0, 1, 2, \dots$*

The region of practical significance in the left-half of the s -plane is bounded by the $j\omega$ -axis as the radian frequency ω varies over the range

$$-\frac{\omega_s}{2} \leq \omega \leq \frac{\omega_s}{2} \quad (3.5-8)$$

where ω_s represents the **sampling frequency** in rad/sec. Over this frequency range, sometimes referred to as the **primary strip**, the transformation

$$z = e^{j\omega T} = e^{j2\pi f/f_s} = e^{j2\pi\omega/\omega_s} \quad (3.5-9)$$

implies a complete rotation around the unit circle in the z -plane. Note that additional rotations will occur around the same path for each frequency increase of $\omega=\omega_s$. This behavior can also be explained by considering the sampled signal and folding frequency concepts discussed earlier. Equivalent conclusions can be drawn when the frequency is expressed in Hz, rather than rad/sec.

Equally important to finding $F(z)$ from $f(k)$ or $f(t)$ is the problem of finding $f(k)$ or $f(t)$ from $F(z)$, that is, the problem of inverting $F(z)$. The Laplace transform is unique: every $f(t)$ has its $F(s)$ and every $F(s)$ has its $f(t)$. The $\mathcal{Z}$ transform of $f(t)$, on the other hand, is not unique, because of possible oscillations or ripple that may be present between the samples. However, the $\mathcal{Z}$ transform of $f(k)$, $\mathcal{Z}[f(k)]$, and the corresponding inverse transform of $F(z)$, $\mathcal{Z}^{-1}F(z)$, are unique. Thus, the sequence of samples $f(k)$ can be determined from $F(z)$.

The inverse $\mathcal{Z}$ transform of $F(z)$ can be calculated using complex integration, a process which is tedious and impractical. In this book, the methods of long division and partial fraction expansion are emphasized.

3.5-1 $\mathcal{Z}$ Transform Theorems

The Laplace transform theorems considered in Chapter 2 have $\mathcal{Z}$ transform counterparts. The $\mathcal{Z}$ transform theorems are presented next.

Theorem 1. *Linearity*

If $f_1(t)$ and $f_2(t)$ are defined for all $t=kT$, where $k=0,1,2,\dots$, that is, if $F_1(z)$ and $F_2(z)$ exist, then

$$\mathcal{Z}[af_1(t) + bf_2(t)] = aF_1(z) + bF_2(z) \quad (3.5-10)$$

where a and b are arbitrary constants. The proof of this theorem follows when the linear summation definition of the $\mathcal{Z}$ transform is applied.

Theorem 2. Advance (Shift Left)

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, then, for $t=kT$ and $k \geq 0$, the $\mathcal{Z}$ transform of $f(t+nT)$ is given by

$$\mathcal{Z}[f(t+nT)] = z^n[F(z) - \sum_{q=0}^{n-1} f(qT)z^{-q}] \quad (3.5-11)$$

The advance theorem can be used to solve difference equations with initial conditions, the same way the differentiation theorem is used to solve differential equations.

Proof: Applying the $\mathcal{Z}$ transform definition to the first-order difference equation term $f(t+T)$ gives

$$\mathcal{Z}[f(t+T)] = \sum_{k=0}^{\infty} f[(k+1)T]z^{-k} = z \sum_{k=0}^{\infty} f[(k+1)T]z^{-(k+1)} = z \sum_{q=1}^{\infty} f(qT)z^{-q} \quad (3.5-12)$$

where $q=k+1$. Adding and subtracting $f(0)$ under the summation

$$\mathcal{Z}[f(t+T)] = z \left[\sum_{q=0}^{\infty} f(qT)z^{-q} - f(0) \right] \quad (3.5-13)$$

Since the summation term is equal to $F(z)$, the desired result becomes

$$\mathcal{Z}[f(t+T)] = \mathcal{Z}[f(k+1)] = z[F(z) - f(0)] \quad (3.5-14)$$

Extending the procedure to higher-order difference equation terms, leads to the general n th-order result expressed by Eq. (3.5-11).

Theorem 3. Summation

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, then, for an arbitrary integer N ,

$$\mathcal{Z}\left[\sum_{k=0}^N f(kT)\right] = \frac{F(z)}{1-z^{-1}} \quad (3.5-15)$$

Theorem 4. Delay (Shift Right)

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, then, for $t=kT$, where $k \geq 0$,

$$\mathcal{Z}[f(t-nT)u(t-nT)] = z^{-n}F(z) \quad (3.5-16)$$

$u(t)$ is the unit-step function.

Proof: Applying the $\mathcal{Z}$ transform definition

$$\begin{aligned}
\mathcal{Z}[f(t-nT)u(t-nT)] &= \sum_{k=0}^{\infty} \{f[(k-n)T]u[(k-n)T]\} z^{-k} \\
&= z^{-n} \sum_{k=0}^{\infty} \{f[(k-n)T]u[(k-n)T]\} z^{-(k-n)}
\end{aligned} \tag{3.5-17}$$

Substituting $q=k-n$ in Eq. (3.5-17), we obtain the desired result

$$\mathcal{Z}[f(t-nT)u(t-nT)] = z^{-n} \sum_{q=-n}^{\infty} f(qT)u(qT)z^{-q} = z^{-n} \sum_{q=0}^{\infty} f(qT)z^{-q} = z^{-n}F(z) \tag{3.5-18}$$

Note that $u(qT)=1$ for $q \geq 0$, and $u(qT)=0$ for $q < 0$.

Theorem 5. Multiplication by e^{-at} (Damping)

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, then

$$\mathcal{Z}[e^{-at}f(t)] = F(e^{aT}z) \tag{3.5-19}$$

In other words, the $\mathcal{Z}$ transform of the product $e^{-at}f(t)$ can be obtained from $F(z)$ by replacing z by $e^{aT}z$.

Proof: Applying the $\mathcal{Z}$ transform definition gives

$$\mathcal{Z}[e^{-at}f(t)] = \sum_{k=0}^{\infty} f(kT)(e^{aT}z)^{-k} = F(e^{aT}z)$$

as required.

Theorem 6. Complex Differentiation (Multiplication by t)

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, then

$$\mathcal{Z}[tf(t)] = -Tz \frac{d}{dz} F(z) \tag{3.5-20}$$

Thus, multiplication by t (or kT , or k) in the time domain implies differentiation with respect to z in the discrete frequency domain.

Theorem 7. Complex Integration (Division by t)

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, then

$$\mathcal{Z}\left[\frac{f(t)}{t}\right] = -\frac{1}{T} \int \frac{F(z)}{z} dz \quad (3.5-21)$$

This suggests that division by t (or kT , or k) in the time domain implies integration by z in the discrete frequency domain.

Theorem 8. Initial Value Theorem

If $\mathcal{Z}[f(t)] = F(z)$, then

$$f(0) = \lim_{k \rightarrow 0} f(kT) = \lim_{z \rightarrow \infty} F(z) \quad (3.5-22)$$

provided that the limit of $F(z)$ as z approaches infinity exists.

Proof:
$$\lim_{z \rightarrow \infty} F(z) = \lim_{z \rightarrow \infty} \sum_{k=0}^{\infty} f(kT)z^{-k} = \sum_{k=0}^{\infty} f(kT) \lim_{z \rightarrow \infty} z^{-k} = f(0)$$

Note that we can interchange the summation and the limit. *The initial value theorem states that the behavior of the time function near the origin is related to the behavior of its transform, $F(z)$, as z approaches infinity.*

Theorem 9. Final Value Theorem

If the $\mathcal{Z}$ transform of $f(t)$ is $F(z)$, and if the limit of $F(z)(1-z^{-1})$ as z approaches unity exists, then

$$f(\infty) = \lim_{k \rightarrow \infty} f(kT) = \lim_{z \rightarrow 1} (1-z^{-1})F(z) = \lim_{z \rightarrow 1} (z-1)F(z) \quad (3.5-23)$$

The final value theorem states that the behavior of the time function as time approaches infinity is related to the behavior of $(z-1)F(z)$ near $z=1$. Eq. (3.5-23) requires that all of the poles of $F(z)$ lie inside the unit circle, except possibly for a pole at $z=1$. Otherwise, the theorem cannot be applied.

Theorem 10. Convolution Summation

If an ideal sampler operates on a $\mathcal{Z}$ transformable signal $x(t)$ to produce $x(k)$, which is the input to a linear system with transfer function $G(s)$, then, the response of the system $y(k)$ is given by the discrete convolution of the signals $x(k)$ and $g(k)$. That is,

$$y(k) = x(k) * g(k) = \sum_{j=0}^{\infty} g(k-j)x(j) = \sum_{j=0}^{\infty} g(j)x(k-j) \quad (3.5-24)$$

Note, $g(k)$ represents the impulse response of the system, also known as the **weighting sequence**.

The convolution summations of Eq. (3.5-24) are analogous to the convolution integrals for continuous signals, considered in Chapter 2. Thus, if

$$\mathcal{Z}[x(k)] = X(z), \text{ and } \mathcal{Z}[g(k)] = G(z) \quad (3.5-25)$$

then, if $y(k)$ is defined for $k \geq 0$,

$$Y(z) = X(z)G(z) \quad (3.5-26)$$

$$\text{where} \quad G(z) = \frac{Y(z)}{X(z)} \quad (3.5-27)$$

is the transfer function of the discrete system shown in Fig. 3.5-1.

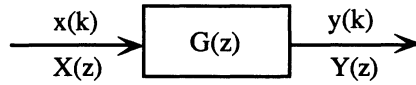


Fig. 3.5-1 Block diagram representation of a discrete system.

Proof: By definition

$$\mathcal{Z}[y(k)] = Y(z) = \sum_{k=0}^{\infty} \left[\sum_{j=0}^{\infty} g(k-j)x(j) \right] z^{-k} \quad (3.5-28)$$

Interchanging the order of summation, we obtain

$$Y(z) = \sum_{j=0}^{\infty} x(j) \sum_{k=0}^{\infty} g(k-j) z^{-k}$$

Substituting $m=(k-j)$, we get the required result

$$Y(z) = \sum_{j=0}^{\infty} x(j) \sum_{m=0}^{\infty} g(m) z^{-m-j} = \sum_{j=0}^{\infty} x(j) z^{-j} \sum_{m=0}^{\infty} g(m) z^{-m} = X(z)G(z)$$

Theorem 11. Periodic Functions

If a function $f_1(t)$ is identically zero except for $0 \leq t \leq NT$, where N is a positive integer, then for the periodic function $f(t)$ defined by

$$f(t) = \sum_{j=0}^{\infty} f_1(t-jNT) \quad (3.5-29)$$

the $\mathcal{Z}$ transform $F(z)$ is given by

$$F(z) = \frac{1}{(1-z^{-N})} F_1(z) \quad (3.5-30)$$

3.5-2 Tables of $\mathcal{Z}$ Transforms

To invert $F(z)$, that is, to determine $f(k)$ or $f(t)$ from $F(z)$, we can form a table of $\mathcal{Z}$ transforms by following a procedure similar to the one used to form the table of Laplace transforms. A table of common $\mathcal{Z}$ transforms is developed next, by applying the definition of the $\mathcal{Z}$ transform.

Example 3.5-1 The unit-step function $u(t)$

The problem is to find $F(z)$ if $f(t)=u(t)$. By definition,

$$F(z) = \mathcal{Z}[u(t)] = \sum_{k=0}^{\infty} u(kT)z^{-k} = \sum_{k=0}^{\infty} z^{-k}$$

Equivalently,

$$F(z) = 1 + z^{-1} + z^{-2} + \dots = \frac{z}{z-1} = \frac{1}{1-z^{-1}} \quad ; \text{ for } |z|>1$$

The result follows from the geometric series

$$\sum_{k=0}^{\infty} a^k = 1 + a + a^2 + \dots = \frac{1}{1-a}$$

Example 3.5-2 The exponential function e^{-at}

If $f(t) = e^{-at}$, where 'a' is an arbitrary constant, find $F(z)$. By definition of the $\mathcal{Z}$ transform,

$$F(z) = \mathcal{Z}(e^{-at}) = \sum_{k=0}^{\infty} e^{-akT}z^{-k} = \sum_{k=0}^{\infty} (e^{-aT}z^{-1})^k = \frac{1}{1-e^{-aT}z^{-1}} = \frac{z}{z-e^{-aT}}$$

Example 3.5-3 The ramp function $tu(t)$

Instead of using the $\mathcal{Z}$ transform definition, let us apply Theorem 6. This suggests that if

$$f(t) = tu(t)$$

then

$$F(z) = \mathcal{Z}[tu(t)] = -Tz \frac{d}{dz} Q(z)$$

where

$$Q(z) = \mathcal{Z}[u(t)] = \frac{z}{(z - 1)}$$

Thus, since

$$\frac{d}{dz} Q(z) = -\frac{1}{(z-1)^2}$$

the result is

$$F(z) = \frac{Tz}{(z-1)^2} = \frac{Tz^{-1}}{(1-z^{-1})^2}$$

Example 3.5-4 $\cos bt$

If $f(t) = \cos bt$, the definition of the $\mathcal{Z}$ transform gives

$$F(z) = \mathcal{Z}(\cos bt) = \sum_{k=0}^{\infty} (\cos b k T) z^{-k}$$

Using Euler's identity, we obtain

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} \left(\frac{e^{jb k T} + e^{-jb k T}}{2} \right) z^{-k} = \frac{1}{2} \left[\sum_{k=0}^{\infty} (e^{jb T} z^{-1})^k + \sum_{k=0}^{\infty} (e^{-jb T} z^{-1})^k \right] = \\ &= \frac{1}{2} \left[\frac{1}{1 - e^{jb T} z^{-1}} + \frac{1}{1 - e^{-jb T} z^{-1}} \right] = \frac{z^2 - z \cos b T}{z^2 - 2z \cos b T + 1} \end{aligned}$$

Example 3.5-5 $e^{-at} \cos bt$

From Example 3.5-4 and Theorem 5, we have that

$$\mathcal{Z}(e^{-at} \cos bt) = \frac{(e^{aT} z)^2 - e^{aT} z \cos b T}{(e^{aT} z)^2 - 2e^{aT} z \cos b T + 1} = \frac{z^2 - z e^{-aT} \cos b T}{z^2 - 2e^{-aT} z \cos b T + e^{-2aT}}$$

Additional $\mathcal{Z}$ transform pairs can be developed in a similar manner. Some of the most common pairs are included in Table 3.5-1.

Table 3.5-1 Table of Laplace and $\mathcal{Z}$ Transform Pairs

Item	$f(t)$	$f(kT)$	$F(s)$	$F(z)$
1	$\delta(t)$	$\delta(kT)$	1	1
2	$u(t)$	$u(kT)$	$\frac{1}{s}$	$\frac{z}{(z-1)}$
3	t	kT	$\frac{1}{s^2}$	$\frac{Tz}{(z-1)^2}$
4	t^2	$(kT)^2$	$\frac{2}{s^3}$	$\frac{T^2 z(z+1)}{(z-1)^3}$
5	e^{-at}	e^{-akT}	$\frac{1}{(s+a)}$	$\frac{z}{(z-e^{-aT})}$
6	te^{-at}	$(kT)e^{-akT}$	$\frac{1}{(s+a)^2}$	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$
7	$\sin bt$	$\sin bkT$	$\frac{b}{(s^2+b^2)}$	$\frac{z \sin bT}{z^2 - 2z \cos bT + 1}$
8	$\cos bt$	$\cos bkT$	$\frac{s}{(s^2+b^2)}$	$\frac{z^2 - z \cos bT}{z^2 - 2z \cos bT + 1}$
9	$e^{-at} \sin bt$	$e^{-akT} \sin bkT$	$\frac{b}{[(s+a)^2+b^2]}$	$\frac{ze^{-aT} \sin bT}{z^2 - 2ze^{-aT} \cos bT + e^{-2aT}}$
10	$e^{-at} \cos bt$	$e^{-akT} \cos bkT$	$\frac{(s+a)}{[(s+a)^2+b^2]}$	$\frac{z^2 - ze^{-aT} \cos bT}{z^2 - 2ze^{-aT} \cos bT + e^{-2aT}}$
11	$y(t-nT)u(t-nT)$	$y(k-n)u(k-n)$	$e^{-snT}Y(s)$	$z^{-n}Y(z)$
12	$y^{(n)}(t)$		$s^n Y(s) - \sum_{k=1}^n s^{n-k} y^{(k-1)}(0)$	
13	$y(t+nT)$	$y(k+n)$		$z^n [Y(z) - \sum_{q=0}^{n-1} y(q)z^{-q}]$
14		$a^k = (e^{\ln a})^k$		$\frac{z}{(z-a)}$
15		$(-a)^k = a^k \cos k\pi$		$\frac{z}{(z+a)}$

3.5-3 Method of Long Division

Assume that $F(z)$ is written in rational polynomial form, using negative powers of z . That is, let

$$F(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_m z^{-m}}{1 + b_1 z^{-1} + b_2 z^{-2} + \dots + b_n z^{-n}} \quad (3.5-31)$$

Using the power series definition of $F(z)$, namely,

$$F(z) = f(0) + f(1)z^{-1} + f(2)z^{-2} + \dots \quad (3.5-32)$$

the functional values $f(0)$, $f(1)$, etc., can be determined by expanding $F(z)$ of Eq. (3.5-32) by long division. This approach of inverting $F(z)$ is practical and easy to program on a digital computer. It does not, however, lead to a "closed-form" expression for $f(k)$. If a closed-form expression for $f(k)$ or $f(t)$ is desired, then, the transform tables and, if needed, partial fraction expansion can be used.

In the case of partial fraction expansion, the function $F(z)/z$, rather than $F(z)$, is expanded, because the functions $F(z)$ appearing in the $\mathcal{Z}$ transform table have the variable ' z ' in their numerator. Once $F(z)/z$ is expanded into partial fractions, $F(z)$ should contain terms that exist in the transform table. The methods for inverting $F(z)$, in order to obtain $f(k)$ or $f(t)$, are illustrated by means of several examples.

Example 3.5-6

Determine the first 5 values of the sequence $f(k)$ for the function whose transform is given by

$$F(z) = \frac{z}{z - 0.5} = \frac{1}{1 - 0.5z^{-1}}$$

(a) Long Division Method

Using the second form of $F(z)$, and long-dividing, we obtain the series

$$F(z) = 1 + 0.5z^{-1} + 0.25z^{-2} + 0.125z^{-3} + 0.0625z^{-4} + \dots$$

Thus, the first 5 values are given by $f(k) = \{1, 0.5, 0.25, 0.125, 0.0625\}$.

(b) $\mathcal{Z}$ Transform Tables Method

Using item 14 of Table 3.5-1, we obtain the closed-form solution

$$f(k) = (0.5)^k = (e^{\ln 0.5})^k = e^{-0.69315k}$$

For $k=0, 1, 2, 3, 4$, this $f(k)$ gives the same functional values as part (a).

Example 3.5-7

Find $f(k)$, if $F(z)$ is given by

$$F(z) = \frac{5}{z+5}$$

(a) Express $F(z)$ in the form

$$F(z) = 5z^{-1} \left(\frac{z}{z+5} \right)$$

From item 15 of Table 3.5-1, we get

$$\begin{aligned} f(k) &= 5z^{-1}(5^k \cos k\pi)u(k) = 5[5^{k-1} \cos(k-1)\pi]u(k-1) \\ &= [5^k \cos(k-1)\pi]u(k-1) \end{aligned}$$

That is, $f(0)=0$ and $f(k) = 5^k \cos(k-1)\pi$, for $k \geq 1$. Note that since

$$F(z) = \frac{5}{z+5} = \frac{5}{z-(-5)}$$

item 14 in Table 3.5-1 can be applied to obtain the equivalent answer

$$f(k) = 5^k(-1)^{k-1}u(k-1)$$

(b) The given $F(z)$ can also be expressed in the form

$$F(z) = A+B \frac{z}{z+5} = \frac{Az+5A+Bz}{z+5}$$

Equating this with the given $F(z)$, we obtain

$$\frac{5}{z+5} = \frac{(A+B)z+5A}{z+5}$$

Thus, $A=1$ and $B=-1$, and the expansion is

$$F(z) = \frac{5}{z+5} = 1 - \frac{z}{z+5}$$

The inverse transform then becomes

$$f(k) = \delta(k) - 5^k \cos k\pi; \text{ for } k \geq 0$$

Since $\delta(k)=1$, it can be seen that this answer is also correct.

Example 3.5-8

Determine $f(k)$ if $F(z)$ is given by

$$F(z) = \frac{z^2+0.7601z}{z^2-1.36468z+0.54881}$$

Determine, also, $f(t)$ and $F(s)$ if the sampling time is $T=0.1$ sec.

Since the denominator is of second degree, we must factor it to determine the type of roots. This leads to the complex conjugate pair

$$z_1, z_2 = 0.68234 \pm j0.28848$$

The relevant items in Table 3.5-1 are 9 and/or 10. Equating denominators between the $F(z)$ and item 9 (or 10) in the $\mathcal{Z}$ transform table, results in the equations

$$e^{-2aT} = 0.54881$$

$$2e^{-aT}\cos bT = 1.36468$$

Solving for aT , yields

$$aT = 0.3$$

Also,

$$\cos bT = 0.92106$$

Solving for bT , gives

$$bT = 0.4$$

From the numerator of $F(z)$, and taking into consideration the numerator forms of items 9 and 10 in the transform table, we can write

$$z^2 + 0.7601z = z^2 - ze^{-aT}\cos bT + zAe^{-aT}\sin bT$$

where A is an arbitrary constant. Using the previously determined values for aT and bT , the last equation becomes

$$z^2 + 0.7601z = z^2 - 0.68223z + A0.28849z$$

By equating the coefficients, we get

$$A = 5$$

Thus, the transform function $F(z)$ can be expressed in the form

$$F(z) = \frac{z^2 - 0.68223z}{z^2 - 1.36468z + 0.54881} + 5 \frac{0.28849z}{z^2 - 1.36468z + 0.54881}$$

Using entries 9 and 10 in Table 3.5-1, we obtain

$$f(k) = e^{-0.3k}\cos 0.4k + 5e^{-0.3k}\sin 0.4k$$

Finally, since $t=kT$ and $T=0.1$ sec,

$$f(t) = e^{-3t}\cos 4t + 5e^{-3t}\sin 4t$$

and

$$F(s) = \frac{s+3}{(s+3)^2+4^2} + 5 \frac{4}{(s+3)^2+4^2}$$

3.5-4 Partial Fraction Expansion

The general form of $F(z)$ for which the inverse $\mathcal{Z}$ transform is desired can be expressed in rational polynomial form as

$$F(z) = \frac{A(z)}{B(z)} = \frac{a_0 z^m + a_1 z^{m-1} + \dots + a_{m-1} z + a_m}{z^n + b_1 z^{n-1} + \dots + b_{n-1} z + b_n} \quad (3.5-33)$$

where m and n are real positive integers and the coefficients a and b are real constants. If $F(z)$ represents the discrete transfer function of a realizable system, the condition of realizability dictates that

$$m \leq n \quad (3.5-34)$$

otherwise, the system output will depend on future values of the input. If the condition of realizability is not satisfied, we can divide $A(z)$ by the polynomial $B(z)$ as many times as necessary to produce a polynomial in z and a rational polynomial remainder which satisfies the realizability condition.

The first step in expanding $F(z)$ into partial fractions is to factor its denominator polynomial into first-order and second-order terms, like those found in Table 3.5-1. Then, $F(z)/z$, is expanded into a series of partial fractions using the procedures developed in Chapter 2 for expanding $F(s)$. Finally, $F(z)$ is inverted using the $\mathcal{Z}$ transform tables. The method of expanding $F(z)$ into partial fractions is illustrated by the following examples.

Example 3.5-9

Expand the following function $F(z)$ into partial fractions. Also, find its inverse transform $f(k)$ using the transform table.

$$F(z) = \frac{2z^2 - 3z}{(z-1)^2(z-2)}$$

Note that this $F(z)$ has one real and distinct denominator root located at $z=2$, and two real and equal (repeated) roots at $z=1$. Thus, the partial fraction expansion is of the form

$$\frac{F(z)}{z} = \frac{2z-3}{(z-1)^2(z-2)} = \frac{A}{z-2} + \frac{B}{z-1} + \frac{C}{(z-1)^2}$$

where A , B , and C are the residues. The values of A and C are evaluated from

$$A = \left. \frac{2z-3}{(z-1)^2} \right|_{z=2} = 1$$

and

$$C = \left. \frac{2z-3}{z-2} \right|_{z=1} = 1$$

To find B , we put both sides of the expansion under the same denominator and equate the numerators. This gives

$$2z-3 = A(z-1)^2 + B(z-1)(z-2) + C(z-2)$$

Substituting $A=C=1$, we get

$$2z-3 = z^2(1+B) + z(-3B-1) + (2B-1)$$

There are three equations that result by equating coefficients, that is,

$$1+B = 0, \text{ or } -3B-1 = 2, \text{ or } 2B-1 = -3$$

Solving any of them leads to the value $B=-1$. The partial fraction expansion of $F(z)/z$ is

$$\frac{F(z)}{z} = \frac{1}{z-2} - \frac{1}{z-1} + \frac{1}{(z-1)^2}$$

Multiplying both sides by z , we obtain

$$F(z) = \frac{z}{z-2} - \frac{z}{z-1} + \frac{z}{(z-1)^2}$$

Using Table 3.5-1, the inverse $\mathcal{Z}$ transform becomes

$$f(k) = (2)^k - 1 + k, \text{ for } k \geq 0$$

This $f(k)$ resembles the solution $y(k)$ of the first-order difference equation

$$y(k+1) - y(k) = (1+2^k)u(k)$$

considered earlier in Example 3.3-9.

Example 3.5-10

Determine $f(k)$ if $F(z)$ is given by

$$F(z) = \frac{z+2}{(z-1)(z-2)}$$

Since the numerator terms do not have the common factor z , let us expand $F(z)$ as is. Let

$$F(z) = \frac{z+2}{(z-1)(z-2)} = \frac{A}{z-1} + \frac{B}{z-2}$$

from which we obtain $A=-3$ and $B=4$. Thus,

$$F(z) = -3 \frac{1}{z-1} + 4 \frac{1}{z-2}$$

The right-hand side can also be written in the form

$$F(z) = z^{-1}(-3 \frac{z}{z-1} + 4 \frac{z}{z-2})$$

Using Table 3.5-1, we get

$$f(k) = z^{-1}[-3 + 4(2)^k]u(k) = [-3 + 4(2)^{k-1}]u(k-1)$$

Equivalently,

$$f(0)=0, \text{ and } f(k) = -3 + 4(2)^{k-1} = -3 + (2)^{k+1} \text{ for } k \geq 1$$

Example 3.5-11

Determine $f(k)$ if the $F(z)$ is given by

$$F(z) = \frac{z^2+2}{(z-1)(z-2)}$$

Note that the numerator does not have the common factor z ; in addition, the numerator is of the same order as the denominator. Dividing the numerator by the denominator gives

$$F(z) = 1 + \frac{3z}{(z-1)(z-2)}$$

Next, expanding this new function into partial fractions, we get

$$F(z) = 1 + \frac{A}{z-1} + \frac{B}{z-2} = 1 - \frac{3}{z-1} + \frac{6}{z-2} = 1 + z^{-1}(-3 \frac{z}{z-1} + 6 \frac{z}{z-2})$$

Using the $\mathcal{Z}$ transform table, we obtain

$$f(k) = \delta(k) + [-3 + 6(2)^{k-1}]u(k-1)$$

Thus, for $k=0$, $f(0)=1$, and for $k \geq 1$,

$$f(k) = -3 + 6(2)^{k-1} = -3 + 3(2)^k$$

As another approach, consider expanding $F(z)$ in the form

$$F(z) = \frac{z^2+2}{(z-1)(z-2)} = A + B \frac{z}{z-1} + C \frac{z}{z-2}$$

To find A , let $z = 0$ on both sides of the equation. This gives $A=1$. The value of B can be determined by multiplying by $(z-1)/z$ and letting $z=1$. Therefore

$$B = \frac{z^2+2}{z(z-2)} \Big|_{z=1} = -3$$

and

$$C = \frac{z^2+2}{z(z-1)} \Big|_{z=2} = 3$$

Thus, the expanded transform function $F(z)$ is

$$F(z) = 1 - 3 \frac{z}{z-1} + 3 \frac{z}{z-2}$$

from which the inverse $\mathcal{Z}$ transform becomes

$$f(k) = \delta(k) - 3 + 3(2)^k; \text{ for } k \geq 0$$

Since $\delta(0)=1$, this answer is equivalent to the one derived previously.

3.6 TIME RESPONSE USING $\mathcal{Z}$ TRANSFORM METHODS

The $\mathcal{Z}$ transform method for solving difference equations parallels the Laplace transform method for solving differential equations. Like the Laplace transform, the $\mathcal{Z}$ transform method is systematic and gives the total, that is, complementary plus particular solution in one operation. The initial conditions are taken care of automatically.

To solve difference equations by $\mathcal{Z}$ transforms and to determine the response of discrete systems, the advance (shift left) theorem can be applied. The advance theorem is listed as item 12 in Table 3.5-1. This method is developed by means of examples.

Example 3.6-1

Consider the amortization algorithm developed in Example 3.2-3. The difference equation is

$$y(k+1) = (1+R)y(k) - x(k)$$

where R represents the periodic interest rate, usually monthly. The initial condition $y(0)$ represents the amount A of the loan to be amortized, and the forcing function $x(k)$ denotes the payment amount P , a constant. The objective is to solve the difference equation.

Applying the advance theorem to both sides of the difference equation, we obtain

$$z[Y(z)-y(0)] = (1+R)Y(z)-X(z)$$

Since $x(k)$ is a step of size P , the input transform is given by

$$X(z) = \mathcal{Z}[Pu(k)] = \frac{Pz}{z-1}$$

Using the initial condition $y(0)=A$, we get the output transform

$$Y(z) = \frac{Az}{z-(1+R)} - \frac{Pz}{(z-1)[z-(1+R)]}$$

After dividing by z and expanding the last term into partial fractions, we get

$$Y(z) = A \frac{z}{z-(1+R)} + \frac{P}{R} \frac{z}{z-1} - \frac{P}{R} \frac{z}{z-(1+R)}$$

From the transform table, the inverse $\mathcal{Z}$ transform is

$$y(k) = A(1+R)^k + \frac{P}{R} [1 - (1+R)^k] ; \text{ for } k \geq 0$$

which is the solution of the original difference equation.

To determine the payment amount P so that the loan amount A is repaid (reduced to zero) in N payments, substitute $k=N$ in the above equation and set $y(N)=0$. This gives

$$y(N) = 0 = A(1+R)^N + \frac{P}{R} [1 - (1+R)^N]$$

Thus, the payment amount is

$$P = \frac{AR(1+R)^N}{(1+R)^N - 1} = \frac{AR}{1 - \frac{1}{(1+R)^N}}$$

Example 3.6-2

Solve the difference equation

$$y(k+1) - 0.60653y(k) = 0$$

subject to the initial condition $y(0)=1$.

Application of the advance theorem gives

$$z[Y(z) - y(0)] - 0.60653Y(z) = 0$$

Rearranging, we obtain

$$Y(z) = \frac{y(0)z}{z - 0.60653} = \frac{z}{z - 0.60653}$$

From Table 3.5-1, the solution is

$$y(k) = (0.60653)^k = e^{-0.5k} ; \text{ for } k \geq 0$$

This result agrees with the result obtained in Example 3.3-2.

Example 3.6-3

Solve the following difference equation using $\mathcal{Z}$ transforms.

$$y(k+2) - 0.3y(k+1) - 0.1y(k) = 0$$

From the advance theorem,

$$z^2[Y(z) - y(0) - y(1)z^{-1}] - 0.3z[Y(z) - y(0)] - 0.1Y(z) = 0$$

The output transform $Y(z)$ is

$$Y(z) = \frac{y(0)z^2 + [y(1) - 0.3y(0)]z}{z^2 - 0.3z - 0.1} = \frac{y(0)z^2 + [y(1) - 0.3y(0)]z}{(z - 0.5)(z + 0.2)}$$

The real and distinct denominator roots imply the solution

$$y(k) = A(0.5)^k + B(-0.2)^k = A(0.5)^k + B(\cos k\pi)(0.2)^k$$

where the coefficients A and B depend on the initial conditions $y(0)$ and $y(1)$.

Example 3.6-4

A system is modeled by the difference equation

$$y(k+1) - 0.60653y(k) = 0.39347x(k)$$

where $x(k)$ is the input function. Determine the unit-step response of the system, when $y(0)=0$.

From the advance theorem

$$zY(z) - 0.60653Y(z) = 0.39347X(z)$$

Since $X(z)=z/(z-1)$, the output transform is

$$Y(z) = \frac{0.39347z}{(z-1)(z-0.60653)}$$

Expanding $Y(z)/z$ into partial fractions gives

$$\frac{Y(z)}{z} = \frac{1}{z-1} - \frac{1}{z-0.60653}$$

from which the output transform $Y(z)$ becomes

$$Y(z) = \frac{z}{z-1} - \frac{z}{z-0.60653}$$

Using the transform table, the desired unit-step response is

$$y(k) = (1 - 0.60653^k)u(k)$$

This result agrees with the result obtained in Example 3.3-7 using classical methods.

Example 3.6-5

Determine the unit-impulse response of the difference equation used in Example 3.6-4. That is, determine $y(k)$ when $x(k)=\delta(k)$.

Since $x(k)=\delta(k)$, its transform is $X(z)=1$. The output transform is

$$Y(z) = \frac{0.39347}{z-0.60653} = z^{-1} [0.39347 \frac{z}{z-0.60653}]$$

which leads to the solution

$$y(k) = 0.39347(0.60653)^{k-1}u(k-1)$$

Equivalently, the response sequence is described by

$$y(0)=0, \text{ and } y(k) = 0.39347(0.60653)^{k-1}, \text{ for } k \geq 1$$

Example 3.6-6

A discrete system is represented by the difference equation

$$y(k+2) - 1.7736y(k+1) + 0.81873y(k) = 0.02332x(k+1) + 0.02181x(k)$$

Determine the unit-step response of the system assuming zero initial conditions.

Applying the advance theorem results in the output transform

$$Y(z) = \frac{(0.02332z+0.02181)X(z)}{z^2-1.7736z+0.81873} = \frac{(0.02332z+0.02181)z}{(z-1)[(z-0.8868)^2+0.17977^2]}$$

Expanding $Y(z)/z$ into partial fractions gives

$$\frac{Y(z)}{z} = \frac{A}{z-1} + \frac{Bz+D}{z^2-1.7736z+0.81873} = \frac{1}{z-1} + \frac{-z+0.79692}{z^2-1.7736z+0.81873}$$

Therefore, the output transform is

$$Y(z) = \frac{z}{z-1} - \left[\frac{z^2-0.79692z}{z^2-1.7736z+0.81873} \right]$$

To invert the term in the square brackets, consider first the denominator and compare it to the denominator of items 9 and 10 in Table 3.5-1. From the constant terms, we obtain

$$e^{-2aT} = 0.81873, \text{ or } aT = 0.1$$

From the coefficients of the denominator

$$2e^{-aT}\cos bT = 1.7736, \text{ or } bT = 0.2$$

From the respective numerators,

$$e^{-aT}\cos bT = 0.8868 \text{ and } e^{-aT}\sin bT = 0.17976$$

Thus, the output transform can be written in the form

$$Y(z) = \frac{z}{z-1} - \left[\frac{z^2 - 0.8868z}{z^2 - 1.7736z + 0.81873} + 0.5 \frac{0.17976z}{z^2 - 1.7736z + 0.81873} \right]$$

Finally, using the $\mathcal{Z}$ transform table, the solution becomes

$$y(k) = 1 - e^{-0.1k}(\cos 0.2k + 0.5 \sin 0.2k) = 1 - (0.90484)^k(\cos 0.2k + 0.5 \sin 0.2k)$$

The result agrees with that obtained in Example 3.3-10 using classical methods.

3.7 TRANSFER FUNCTIONS AND BLOCK DIAGRAMS

The transfer function of a linear, time-invariant, discrete system with input $x(k)$ and output $y(k)$ is defined as the ratio of the output transform $Y(z)$ to the input transform $X(z)$. Thus, if the $\mathcal{Z}$ transfer function, or just discrete transfer function, is denoted by $G(z)$, then

$$G(z) = \frac{Y(z)}{X(z)} \quad (3.7-1)$$

Eq. (3.7-1) suggests representing a single-input-single-output (SISO) discrete system in block diagram form, as shown in Fig. 3.7-1.

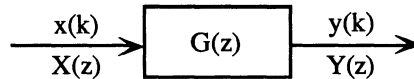


Fig. 3.7-1 Block diagram model of a discrete system.

A discrete transfer function may be provided or it may have to be derived from system models such as difference equations, block diagrams, or signal flow graphs. Once determined, the discrete transfer function can be used for analysis and design, in both time as well as frequency domains.

If an n th-order discrete system is represented by a difference equation, the corresponding discrete transfer function can be easily obtained by $\mathcal{Z}$ transforming both sides of the equation, assuming zero initial conditions. Assuming same order numerator and denominator polynomials, we obtain the transfer function

$$G(z) = \frac{Y(z)}{X(z)} = \frac{a_0 z^n + a_1 z^{n-1} + \dots + a_{n-1} z + a_n}{z^n + b_1 z^{n-1} + \dots + b_{n-1} z + b_n} \quad (3.7-2)$$

or, equivalently,

$$G(z) = \frac{Y(z)}{X(z)} = \frac{a_0 + a_1 z^{-1} + \dots + a_{n-1} z^{-n+1} + a_n z^{-n}}{1 + b_1 z^{-1} + \dots + b_{n-1} z^{-n+1} + b_n z^{-n}} \quad (3.7-3)$$

If the transfer function of a discrete system is known, Eq. (3.7-3) can be used to obtain the corresponding difference equation model. Cross-multiplying terms in Eq. (3.7-3), we obtain

$$(1 + b_1 z^{-1} + \dots + b_{n-1} z^{-n+1} + b_n z^{-n}) Y(z) = (a_0 + a_1 z^{-1} + \dots + a_{m-1} z^{-m+1} + a_m z^{-m}) X(z) \quad (3.7-4)$$

Inverting the transform functions $Y(z)$ and $X(z)$ to their discrete forms $y(k)$ and $x(k)$, respectively, and recognizing that the negative powers of z represent the delay operator, we obtain the corresponding n th-order difference equation model

$$\begin{aligned} y(k) + b_1 y(k-1) + \dots + b_{n-1} y(k-n+1) + b_n y(k-n) \\ = a_0 x(k) + a_1 x(k-1) + \dots + a_{n-1} x(k-n+1) + a_n x(k-n) \end{aligned} \quad (3.7-5)$$

It is also possible to obtain the alternative form of the difference equation by replacing k with $k+n$.

3.7-1 The Pulse Transfer Function

In many practical systems, a sampled signal $x^*(t)$, coming from an analog-to-digital converter (ADC) is applied to some analog process or plant, represented by the transfer function $G_p(s)$, via a digital-to-analog converter (DAC). The DAC is modeled by the zero-order-hold (ZOH) element as shown in Fig. 3.7-2.

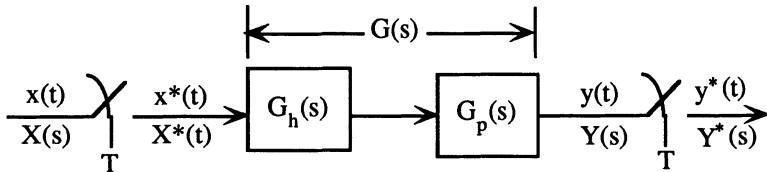


Fig. 3.7-2 Block diagram of a sampled-data system.

A so-called fictitious sampler has been added at the output of the analog process to illustrate that the $\mathcal{Z}$ transform relates a sampled output to a sampled input. Equivalently, *the discrete transfer function is defined from the output of a sampler to the input of the next sampler*. It is assumed that the samplers are synchronized and of the same duration.

From the discrete convolution theorem, we can express the output sequence in the form

$$y(k) = x(k) * g(k) = \sum_{j=0}^{\infty} g(jT) x(kT - jT) \quad (3.7-6)$$

where $g(t)$ is the impulse response of $G(s) = G_h(s)G_p(s)$. Then, from Eq. (3.5-26),

$$Y(z) = G(z)X(z)$$

or
$$G(z) = \frac{Y(z)}{X(z)} \quad (3.7-7)$$

The discrete transfer function $G(z)$ is called the **Pulse Transfer Function (PTF)** and is defined by

$$G_h G_p(z) = \mathcal{Z}[G_h(s)G_p(s)] \quad (3.7-8)$$

where
$$G_h(s) = \frac{(1-e^{-sT})}{s} \quad (3.7-9)$$

When working with hybrid systems, it is often convenient to adopt a linear operator such as a '*' (star) and follow a frequency domain approach. For instance, for the system of Fig. 3.7-2, we can write

$$Y(s) = [G_h(s)G_p(s)]X^*(s) \quad (3.7-10)$$

Applying the '*' operator on both sides of the equation gives

$$Y^*(s) = [G_h(s)G_p(s)]^* X^*(s) \quad (3.7-11)$$

$X^*(s)$ is already sampled and generally,

$$[G_h(s)G_p(s)]^* \neq G_h^*(s)G_p^*(s) \quad (3.7-12)$$

The starred expression in the brackets is usually written as

$$[G_h(s)G_p(s)]^* = G_h G_p^*(s) \quad (3.7-13)$$

This leads to the sampled output transform expression

$$Y^*(s) = G_h G_p^*(s) X^*(s) \quad (3.7-14)$$

Substituting $s=(\ln z)/T$ leads to the $\mathcal{Z}$ transform result

$$Y(z) = G_h G_p(z) X(z) \quad (3.7-15)$$

The pulse transfer function then is

$$G(z) = G_h G_p(z) = \frac{Y(z)}{X(z)} \quad (3.7-16)$$

Note, again, that the pulse transfer function is defined by

$$G_h G_p(z) = \mathcal{Z}[G_h(s)G_p(s)] \quad (3.7-17)$$

This last relationship is generally different from the product of the individual transforms

$$G_h(z)G_p(z) \quad (3.7-18)$$

regardless of what the two continuous transfer functions $G_h(s)$ and $G_p(s)$ may represent. Looking at this in another way, the two transfer functions $G_h(s)$ and $G_p(s)$ are not separated by samplers. The above concepts are illustrated next by means of examples.

Example 3.7-1

Determine the overall transfer function of the system shown in Fig. 3.7-3. That is, find $Y(z)/X(z)$ when the two continuous transfer functions $G_1(s)$ and $G_2(s)$ are separated by synchronous samplers.

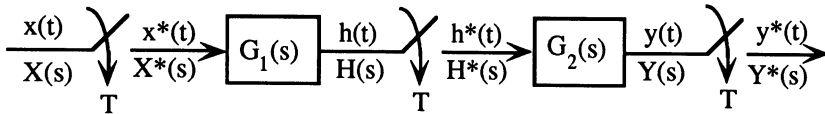


Fig. 3.7-3 Continuous transfer functions separated by samplers.

From the block diagram,

$$H^*(s) = G_1(s)X^*(s)$$

Applying the '*' (star) operator gives

$$H^*(s) = G_1^*(s)X^*(s)$$

Thus,

$$H(z) = G_1(z)X(z)$$

Similarly, for the second transfer function, we can write

$$Y(z) = G_2(z)H(z)$$

Substituting the previous expression for $H(z)$, we obtain

$$Y(z) = G_2(z)G_1(z)X(z)$$

The overall transfer function then is

$$\frac{Y(z)}{X(z)} = G_1(z)G_2(z)$$

Example 3.7-2

Determine the closed-loop transfer function $C(z)/R(z)$ for the sampled-data feedback control system shown in Fig. 3.7-4.

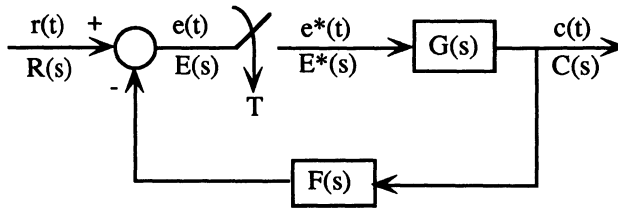


Fig. 3.7-4 A sampled-data feedback control system.

From the block diagram, we can write the algebraic equations

$$E(s) = R(s) - F(s)C(s)$$

and

$$C(s) = E^*(s)G(s)$$

Since the output of the sampler is a linear device in the sense that superposition is valid, we can substitute $C(s)$ from the second equation into the first and then apply the $*$ operator. This gives

$$E(s) = R(s) - F(s)G(s)E^*(s)$$

from which

$$E^*(s) = R^*(s) - FG^*(s)E^*(s)$$

where

$$FG^*(s) = [F(s)G(s)]^*$$

Solving for $E^*(s)$, we obtain

$$E^*(s) = \frac{R^*(s)}{1+FG^*(s)}$$

Substituting $s = (\ln z)/T$ in the last expression, leads to the form

$$E(z) = \frac{R(z)}{1+FG(z)}$$

From the second equation, we get

$$C^*(s) = E^*(s)G^*(s), \text{ or } C(z) = E(z)G(z)$$

The desired closed-loop transfer function then becomes

$$\frac{C(z)}{R(z)} = \frac{G(z)}{1+FG(z)}$$

Example 3.7-3

Determine the pulse transfer function of the system shown in Fig. 3.7-5.

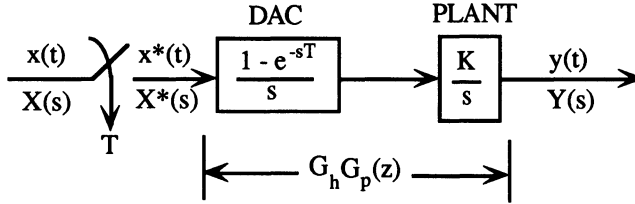


Fig. 3.7-5 Block diagram of a sampled-data system.

The pulse transfer function is obtained from

$$\begin{aligned} G_h G_p(z) &= \frac{Y(z)}{X(z)} = \mathcal{Z}[G_h(s)G_p(s)] = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{K}{s}\right] \\ &= K(1-z^{-1})\mathcal{Z}\left(\frac{1}{s^2}\right) = K \frac{z-1}{z} \frac{Tz}{(z-1)^2} = \frac{KT}{z-1} \end{aligned}$$

This result leads to the block-diagram representation shown in Fig. 3.7-6.

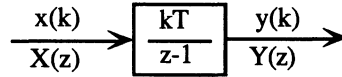


Fig. 3.7-6 Discrete system model for system shown in Fig. 3.7-5.

Example 3.7-4

Determine the unit-step response of the system shown in Fig. 3.7-7.

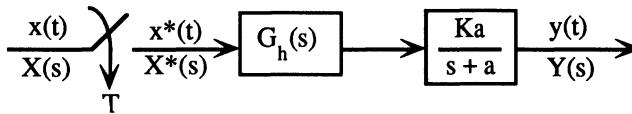


Fig. 3.7-7 Sampled-data system for Example 3.7-4.

The pulse transfer function of the system is given by

$$G_h G_p(z) = \frac{Y(z)}{X(z)} = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{Ka}{s+a}\right] = K\left(\frac{z-1}{z}\right)\mathcal{Z}\left[\frac{a}{s(s+a)}\right] = K\left(\frac{z-1}{z}\right)\mathcal{Z}\left(\frac{1}{s} - \frac{1}{s+a}\right)$$

$$= K \left(\frac{z-1}{z} \right) \left(\frac{z}{z-1} - \frac{z}{z-e^{-aT}} \right) = K \frac{1-e^{-aT}}{z-e^{-aT}} = K \frac{(1-e^{-aT})z^{-1}}{1-e^{-aT}z^{-1}}$$

Thus, the output transform can be expressed by

$$Y(z) = K \left(\frac{1-e^{-aT}}{z-e^{-aT}} \right) X(z)$$

Since the input is a unit-step function, its transform is given by $X(z) = z/(z-1)$. The output transform is then

$$Y(z) = K \frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}$$

The response $y(k)$ can be determine by partial fractions. The result is

$$\frac{Y(z)}{z} = K(1-e^{-aT}) \left[\frac{A}{z-1} + \frac{B}{z-e^{-aT}} \right] = K \left[\frac{1}{z-1} - \frac{1}{z-e^{-aT}} \right]$$

or

$$Y(z) = K \left[\frac{z}{z-1} - \frac{z}{z-e^{-aT}} \right]$$

Thus, the unit-step response of the system is

$$y(k) = K[1 - e^{-akT}]u(k)$$

Example 3.7-5

An analog process (K/s) is controlled by a digital computer as shown in Fig. 3.7-8. It is required to analyze the computer control system by performing the following:

- Determine the closed-loop transfer function $C(z)/R(z)$.
- Determine the output-to-disturbance transfer function $C(z)/D(z)$, and the open-loop transfer function $C(z)/E(z)$.
- Apply the final-value theorem to determine the steady-state characteristics of the system.
- Determine the unit-step response of the system for the parameter values: $K=20$, $K_p=5$, $K_i=1.25$, and $T=0.005$ sec.

The pulse transfer function of the plain integrator was calculated in Example 3.7-3 as

$$G_h G_p(z) = \frac{C(z)}{M(z)} = \frac{KT}{z-1}$$

Assuming that the disturbance enters the control loop at the sampling instants, the control system can be redrawn as shown in Fig. 3.7-9. The analysis of the digital control system can now proceed using conventional block-diagram manipulation techniques.

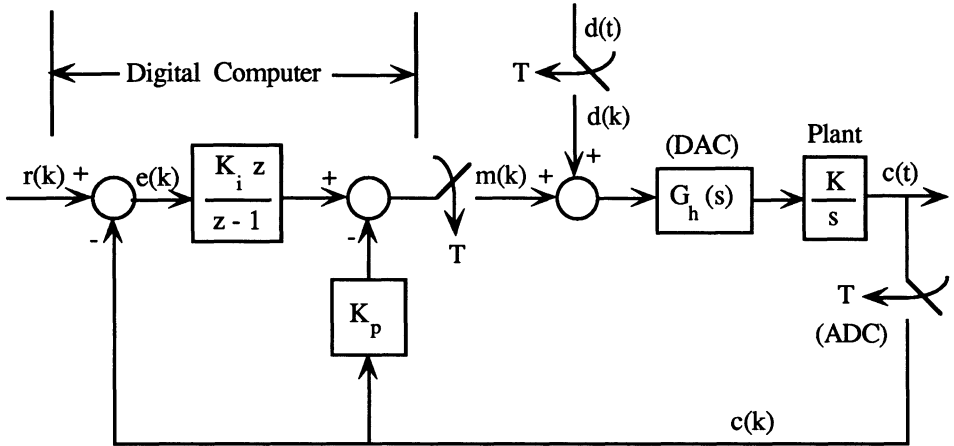


Fig. 3.7-8 Digital PDF control system for K/s plant.

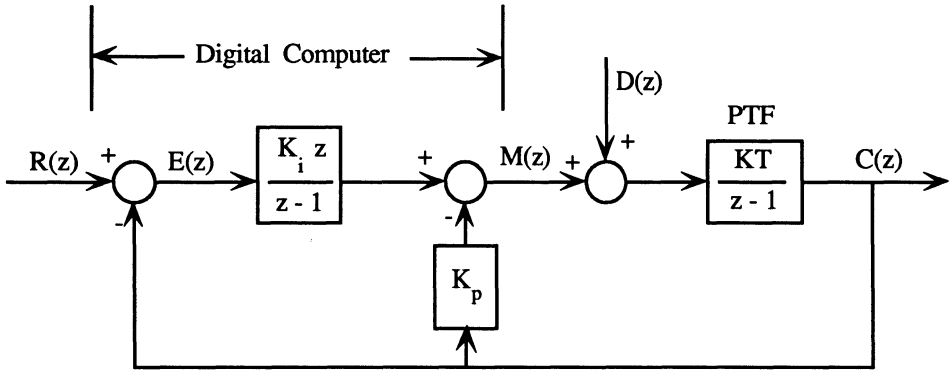


Fig. 3.7-9 Equivalent block diagram of the system in Fig. 3.7-8.

(a) To find the closed-loop transfer function, block diagram reduction methods can be used. Assuming first that $D(z)=0$, we obtain the closed-loop transfer function

$$\frac{C(z)}{R(z)} = \frac{\frac{K_i z}{z-1} \frac{KT}{z-1+KK_p T}}{1 + \frac{K_i z KT}{(z-1)(z-1+KK_p T)}} = \frac{KK_i T z}{z^2 + (KK_p T + KK_i T - 2)z + (1 - KK_p T)}$$

(b) Assuming next that $R(z)=0$, we get the output-to-disturbance transfer function

$$\frac{C(z)}{D(z)} = \frac{\frac{KT}{z-1}}{1 + \frac{KT}{z-1} \left(K_p + \frac{K_i z}{z-1} \right)} = \frac{KT(z-1)}{z^2 + (KK_p T + KK_i T - 2)z + (1 - KK_p T)}$$

The open-loop transfer function is given by

$$\frac{C(z)}{E(z)} = H(z)G(z) = \left(\frac{K_i z}{z-1} \right) \left(\frac{KT}{z-1 + KK_p T} \right) = \frac{KK_i T z}{(z-1)(z-1 + KK_p T)}$$

(c) The discrete final value theorem is defined as

$$f(\infty) = \lim_{z \rightarrow 1} (z-1)F(z)$$

Applying this theorem to the output transform $C(z)$ when the input transform is $R(z) = z/(z-1)$, we obtain the steady-state value

$$c(\infty) = \lim_{z \rightarrow 1} (z-1) \left[\frac{KK_i T z}{z^2 + (KK_p T + KK_i T - 2)z + (1 - KK_p T)} \right] \left(\frac{z}{z-1} \right) = \frac{KK_i T}{KK_i T} = 1$$

This suggests that the system follows step inputs with zero steady-state error.

Applying the final value theorem to the output transform when the disturbance input is a unit-step function, we get

$$c(\infty) = \lim_{z \rightarrow 1} (z-1) \left[\frac{KT(z-1)}{z^2 + (KK_p T + KK_i T - 2)z + (1 - KK_p T)} \right] \left(\frac{z}{z-1} \right) = 0$$

This result implies that the control system rejects steps in disturbance, usually a desirable characteristic.

The open-loop transfer function implies a type 1 system, because of the single integrator term $z/(z-1)$. From the analysis of a comparable type 1 analog control system, it follows that the discrete system can track step inputs with zero steady-state error, and ramp inputs with finite steady-state error. However, it cannot track parabolic inputs.

To verify this, recall that the error transform of a standard-form control system is given by

$$E(z) = \frac{R(z)}{1 + H(z)G(z)}$$

From the final value theorem, the steady-state error of the system is given by

$$e(\infty) = E_{ss} = \lim_{z \rightarrow 1} (1-z^{-1})E(z) = \lim_{z \rightarrow 1} \frac{(1-z^{-1})R(z)}{1 + H(z)G(z)}$$

If the input is a step function of size R_o , that is, if

$$R(z) = \frac{R_o z}{z-1} = \frac{R_o}{1-z^{-1}}$$

then

$$E_{ss}(\text{step}) = \lim_{z \rightarrow 1} \frac{R_o}{1+H(z)G(z)} = \frac{R_o}{1+C_p}$$

where the positional or displacement error constant C_p is given by

$$C_p = \lim_{z \rightarrow 1} H(z)G(z)$$

In the present example

$$C_p = \lim_{z \rightarrow 1} \frac{KK_i Tz}{(z-1)(z-1+KK_p T)} = \infty$$

and, as expected,

$$E_{ss}(\text{step}) = 0$$

If the input is a ramp function of slope R_o , that is, if

$$R(z) = \frac{R_o Tz}{(z-1)^2}$$

then, the steady-state error is

$$E_{ss}(\text{ramp}) = \lim_{z \rightarrow 1} \frac{R_o T}{(z-1)[1+H(z)G(z)]} = \frac{R_o T}{C_v}$$

where the velocity error constant C_v is given by

$$C_v = \lim_{z \rightarrow 1} (z-1)H(z)G(z)$$

Thus, for our example,

$$C_v = \lim_{z \rightarrow 1} \frac{KK_i Tz}{z-1+KK_p T} = \frac{K_i}{K_p}$$

This results in the steady-state-error expression

$$E_{ss}(\text{ramp}) = \frac{R_o T}{(K_i/K_p)} = \frac{R_o K_p T}{K_i}$$

If the analog integral gain K_I is replaced by the digital integral gain K_i divided by the sampling time T , the above result agrees with the result obtained in Chapter 2 for the analog PDF control system. In other words, if $K_i = TK_I$, the results for steady-state error are the same for both analog and digital control systems. In a similar manner, if the input is parabolic, that is, if

$$R(z) = \frac{(R_o/2)T^2 z(z+1)}{(z-1)^3}$$

then

$$E_{ss}(\text{parabolic}) = \lim_{z \rightarrow 1} \frac{(R_o/2)T^2 z(z+1)}{(z-1)^2[1+H(z)G(z)]} = \frac{R_o T^2}{C_a}$$

where the acceleration error constant, C_a , is given by

$$C_a = \lim_{z \rightarrow 1} (z - 1)^2 H(z) G(z)$$

For our example, $C_a=0$ and

$$E_{ss}(\text{parabolic}) = \infty$$

which agrees with the result obtained for the corresponding analog PDF control system considered in Chapter 2.

(d) To find the unit-step response of the control system, we can evaluate its transfer function for the given parameter values. This gives

$$\frac{C(z)}{R(z)} = \frac{0.125z}{z^2 - 1.375z + 0.5}$$

Since the input is a unit-step function, the output transform becomes

$$C(z) = \frac{0.125z^2}{(z-1)(z^2 - 1.375z + 0.5)}$$

The second-order denominator term has complex poles. This suggests expanding $C(z)/z$ into partial fractions of the form

$$\frac{C(z)}{z} = \frac{A}{z-1} + \frac{Bz+D}{z^2 - 1.375z + 0.5} = \frac{A}{z-1} - \left[\frac{(z-0.5)}{z^2 - 1.375z + 0.5} \right]$$

The output transform can be written in a form so that items 9 and 10 of Table 3.5-1 can be applied to the term in square brackets. Thus,

$$\begin{aligned} C(z) &= \frac{z}{z-1} - \left[\frac{z^2 - 0.5z}{z^2 - 1.375z + 0.5} \right] \\ &= \frac{z}{z-1} - \left[\frac{z^2 - 0.6875z}{z^2 - 1.375z + 0.5} + 1.1338 \frac{0.16537z}{z^2 - 1.375z + 0.5} \right] \end{aligned}$$

Setting $e^{-2aT}=0.5$ and $2e^{-aT}\cos bT=1.375$, we obtain aT and bT from which the inverse transform is written as

$$c(k) = 1 - e^{-0.34657k}(\cos 0.23606k + 1.1338 \sin 0.23606k), \text{ for } k=0,1,2,\dots$$

Remember that the spacing of the samples is $T=0.005$ sec.

Example 3.7-6

To illustrate the concept of signal flow graphs for discrete systems, consider the digital control system shown in Fig. 3.7-10a. Recall that this system is referred to as PDFF, because the second F denotes the feedforward (F) control gain K_f . Draw an equivalent

signal flow graph and use Mason's gain formula to find the transfer functions $C(z)/R(z)$ and $C(z)/D(z)$. In addition, determine the unit-step response of the system when $K=20$, $K_p=5$, $K_i=1.25$, $K_f=K_p=5$, and $T=0.005$ sec. Finally, compare the steady-state error performance of this system to the performance of the digital PDF control system (no feedforward) considered in Example 3.7-5.

The discrete signal flow graph is shown in Fig. 3.7-10b. Notice that the delay (z^{-1}) term, used in discrete signal flow graphs, plays a role which parallels the role of the integrator (s^{-1}) term, used in continuous signal flow graphs. The forward paths are:

$$P_1(z) = (1)(1)(1)(1)(K_i)(1)(1)(z^{-1})(KT) = KK_iTz^{-1}$$

$$P_2(z) = (1)(K_f)(1)(1)(z^{-1})(KT) = KK_fTz^{-1}$$

The loops are:

$$L_1(z) = (1)(1)(K_i)(1)(1)(z^{-1})(KT)(-1) = -KK_iTz^{-1}$$

$$L_2(z) = (1)(1)(z^{-1})(KT)(-K_p) = -KK_pTz^{-1}$$

$$L_3(z) = (1)(z^{-1}) = z^{-1}, \text{ and } L_4(z) = (1)(z^{-1}) = z^{-1}$$

Therefore, the system determinant and cofactors are

$$\begin{aligned}\Delta(z) &= 1 - [L_1(z) + L_2(z) + L_3(z) + L_4(z)] + [L_3(z)L_4(z) + L_2(z)L_3(z)] \\ &= 1 + (KK_pT + KK_iT - 2)z^{-1} + (1 - KK_pT)z^{-2}\end{aligned}$$

$$\Delta_1(z) = 1, \text{ and } \Delta_2(z) = 1 - L_3(z) = 1 - z^{-1}$$

The closed-loop transfer function then is

$$\frac{C(z)}{R(z)} = \frac{P_1(z)\Delta_1(z) + P_2(z)\Delta_2(z)}{\Delta(z)} = \frac{KK_iTz^{-1} + KK_fTz^{-1}(1 - z^{-1})}{1 + (KK_pT + KK_iT - 2)z^{-1} + (1 - KK_pT)z^{-2}}$$

If $K_f=0$, the result is identical to that obtained for the PDF control system of the previous example, as expected.

For the $C(z)/D(z)$ transfer function, the path gain from the input $D(z)$ to the output $C(z)$ is

$$P'_1(z) = (1)(z^{-1})KT = KTz^{-1}$$

This implies that the relevant cofactor is

$$\Delta'_1(z) = 1 - L_3(z) = (1 - z^{-1})$$

Since the system determinant does not change, the output-to-disturbance transfer function is given by

$$\frac{C(z)}{D(z)} = \frac{P'_1(z)\Delta'_1(z)}{\Delta(z)} = \frac{KTz^{-1}(1 - z^{-1})}{1 + (KK_pT + KK_iT - 2)z^{-1} + (1 - KK_pT)z^{-2}}$$

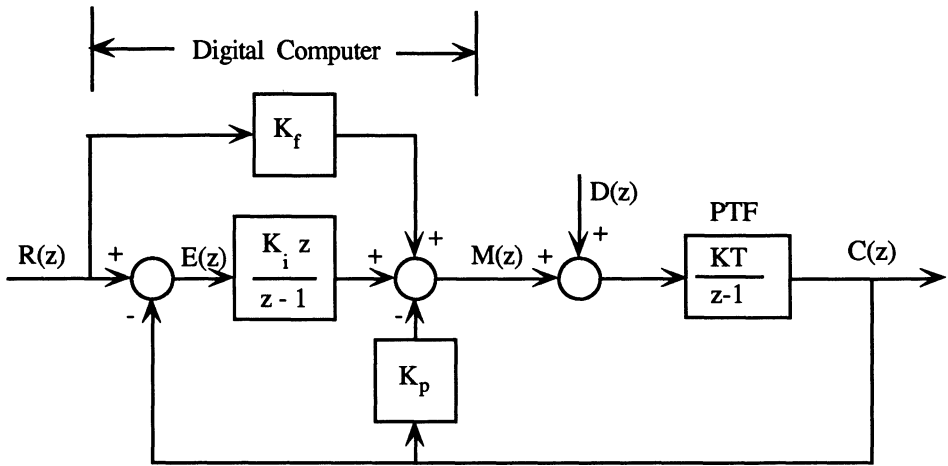


Fig. 3.7-10a Block diagram of digital PDF control system.

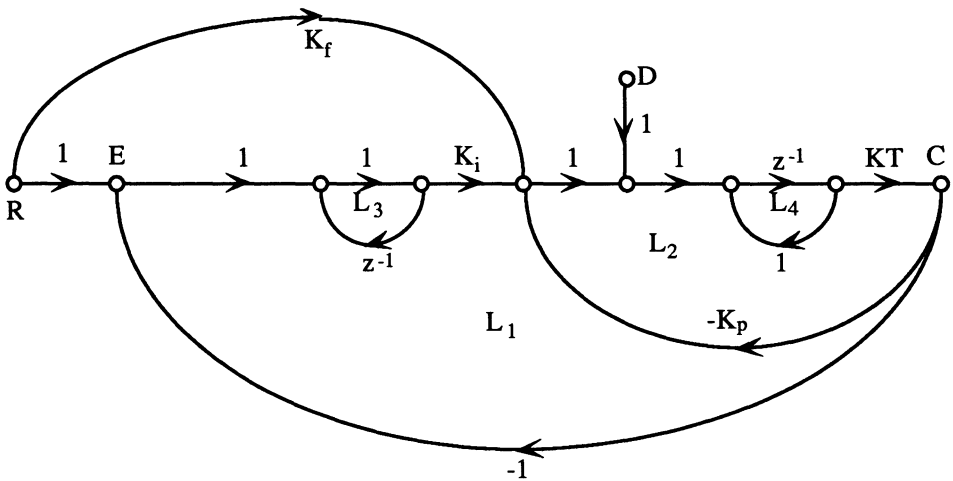


Fig. 3.7-10b A signal flow graph for system of Fig. 3.7-10a.

This transfer function is identical to the $C(z)/D(z)$ obtained for the digital PDF control system. For the given parameter values, the system transfer function becomes

$$\frac{C(z)}{R(z)} = \frac{0.625z - 0.5}{z^2 - 1.375z + 0.5}$$

For a unit-step input, the output transform is

$$C(z) = \frac{(0.625z - 0.5)z}{(z-1)(z^2 - 1.375z + 0.5)}$$

The inverse transform $c(k)$ can be obtained by long division and is shown plotted in Fig. 3.7-11.

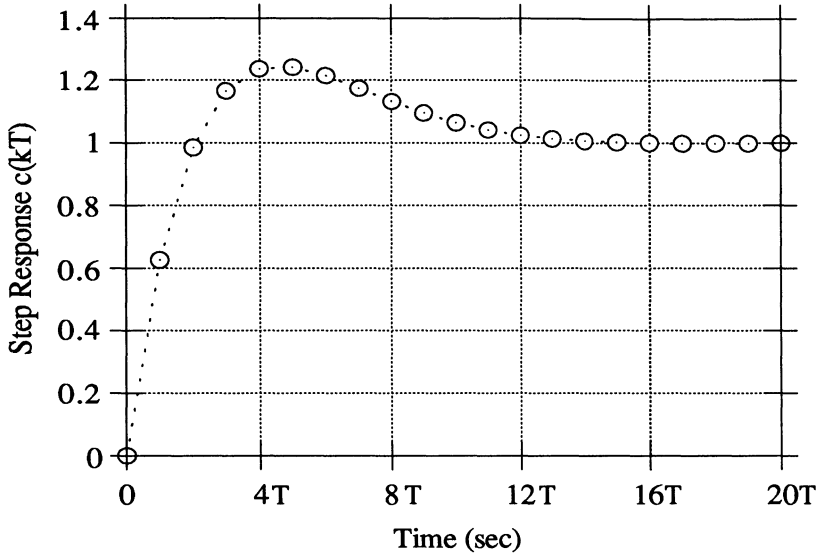


Fig. 3.7-11 The unit-step response of the PDF control system used in Example 3.7-6.

It can be shown that for the given control system the open-loop transfer function is given by

$$H(z)G(z) = \frac{KT[(K_f + K_i)z - K_f]}{(z-1)(z-1 - KK_fT + KK_pT)}$$

For the case in which $K_f = K_p$, this becomes

$$H(z)G(z) = \frac{KT[(K_p + K_i)z - K_p]}{(z-1)^2}$$

Since the error transform for a standard-form, single-loop control system is given by

$$E(z) = \frac{R(z)}{1 + H(z)G(z)}$$

the steady-state error characteristics of the system can be determined from the final value theorem. That is, the theorem given by

$$e(\infty) = E_{ss} = \lim_{z \rightarrow 1} (1-z^{-1})E(z)$$

can be applied for various inputs. Thus, for a step input of size R_o , the error transform is

$$E(z) = \frac{R_o}{(z-1)[1+H(z)G(z)]}$$

and the final value theorem gives

$$E_{ss}(\text{step}) = \lim_{z \rightarrow 1} \frac{R_o}{1+H(z)G(z)} = \frac{R_o}{1+C_p}$$

where the positional error constant, C_p , is given by

$$C_p = \lim_{z \rightarrow 1} H(z)G(z) = \lim_{z \rightarrow 1} \frac{KT[(K_p+K_i)z-K_p]}{(z-1)^2} = \infty$$

This leads to a steady-state error value of zero for step inputs, or

$$E_{ss}(\text{step}) = 0$$

Thus, the digital PDFF control system, like its analog counterpart, follows step inputs with zero steady-state error.

For ramp inputs of slope R_o , the steady-state error is

$$E_{ss}(\text{ramp}) = \lim_{z \rightarrow 1} \frac{R_o T}{C_v}$$

where the velocity error constant, C_v , is given by

$$C_v = \lim_{z \rightarrow 1} \frac{KT[(K_p+K_i)z-K_p]}{(z-1)} = \infty$$

Therefore,

$$E_{ss}(\text{ramp})=0$$

This implies that, in addition to step inputs, the digital PDFF control system can track ramp inputs with zero steady-state error. If the input is parabolic, then

$$R(z) = \frac{(R_o/2)T^2(z+1)}{(z-1)^3}$$

and

$$E_{ss}(\text{parabolic}) = \frac{R_o T^2}{C_a}$$

The acceleration error constant C_a is given by

$$C_a = \lim_{z \rightarrow 1} KT[(K_p+K_i)z-K_p] = KK_i T$$

Therefore, for parabolic inputs the steady-state error is

$$E_{ss}(\text{parabolic}) = \frac{R_o T^2}{K K_i T} = \frac{R_o T}{K K_i}$$

Again, if the analog integral gain K_I is related to the digital integral gain K_i by

$$K_I = \frac{K_i}{T}, \quad \text{or } K_i = T K_I$$

the above steady-state error value matches that obtained for the corresponding analog PDFF control system considered in Chapter 2.

In summary, the digital PDFF control system, like its analog counterpart, follows step and ramp inputs with zero steady-state error, and parabolic inputs with finite steady-state error.

3.8 DISCRETE SYSTEMS: POLES, ZEROS, AND STABILITY

For an n th-order discrete system, the transfer function $G(z)$ defined in Section 3.7 can be expressed in factored form as

$$G(z) = \frac{Y(z)}{X(z)} = \frac{a_o(z-z_1)(z-z_2) \dots (z-z_m)}{(z-p_1)(z-p_2) \dots (z-p_n)} \quad (3.8-1)$$

where the numerator $Y(z)$ and the denominator $X(z)$ represent the transforms of the output $y(k)$ and input $x(k)$, respectively. Also, $z_1, z_2, \dots, z_m$ denote the **zeros**, and $p_1, p_2, \dots, p_n$ denote the **poles** (characteristic roots) of the discrete transfer function. Both zeros and poles may be real or complex, and the multiplying constant a_o is arbitrary. As in the continuous system case, the characteristic equation of the discrete system can be obtained by setting the denominator of $G(z)$ equal to zero.

Bounded-Input-Bounded-Output (BIBO) Stability. *A linear, time-invariant discrete system is said to be bounded-input-bounded-output (BIBO) stable, or just **stable**, if every bounded input sequence always produces a bounded output sequence. A system that is not BIBO stable is said to be **unstable**.*

The stability as well as other performance characteristics of a discrete system can be determined from the location of its poles. The BIBO definition of stability corresponds to the condition that all the poles or characteristic roots of a discrete system lie within the unit circle in the complex z -plane. That is, *if all the zeros of the characteristic equation*

$$(z - p_1)(z - p_2) \dots (z - p_n) = 0 \quad (3.8-2)$$

are such that

$$|p_i| < 1; \quad \text{for } i = 1, 2, \dots, n \quad (3.8-3)$$

*then the system is **stable**. Alternatively, a discrete system is **unstable** if its transfer function $G(z)$ has any poles outside the unit circle.*

A discrete system is said to be **marginally** or **critically stable** if it has simple poles on the unit circle; such systems are BIBO unstable. Repeated poles on the unit circle represent an unstable system.

Generally, system zeros can be anywhere in the z -plane, even though for a **minimum-phase** discrete system both poles and zeros must lie inside the unit circle.

Analogous to the continuous system case, the stability of a discrete system can also be determined from its impulse response, that is, from the output sequence $y(k)$ when the input sequence $x(k)$ is

$$x(k) = \delta(k) = \begin{cases} 1 & \text{for } k=0 \\ 0 & \text{for } k \neq 0 \end{cases} \quad (3.8-4)$$

From Eq. (3.8-1), with $X(z)=1$, the output transform is $Y(z)=G(z)$, and the impulse response is

$$y(k) = g(k) = \mathcal{Z}^{-1}[G(z)] \quad (3.8-5)$$

If the impulse response is considered, an equivalent stability criterion for a discrete system is the following:

For a discrete system to be BIBO stable, the impulse response sequence $g(k)$ goes to zero as k approaches infinity. That is,

$$\lim_{k \rightarrow \infty} g(k) = 0 \quad (3.8-6)$$

Conversely, if

$$\lim_{k \rightarrow \infty} g(k) = \infty \quad (3.8-7)$$

then the system is unstable. Finally, if

$$\lim_{k \rightarrow \infty} g(k) = P \quad (3.8-8)$$

where P is a finite, nonzero value or a bounded oscillation, then the system is marginally stable.

Example 3.8-1

Determine the stability of the first-order transfer function

$$G(z) = \frac{z}{z-a}$$

The corresponding unit-impulse response is

$$g(k) = a^k, \text{ for } k \geq 0$$

The system is stable if $|a| < 1$, because its impulse response goes to zero as k approaches infinity. If $|a| > 1$, then the system is unstable, because $\lim_{k \rightarrow \infty} g(k) = \infty$. Equivalently, the pole is inside the unit circle for the first case (stable), and outside for the second case (unstable).

Example 3.8-2

Determine the stability of the system whose transfer function is

$$G(z) = \frac{z}{z+1}$$

Its unit-impulse response is given by

$$g(k) = (-1)^k, \text{ for } k \geq 0$$

The system is marginally stable, or BIBO unstable, because its impulse response oscillates between ± 1 .

Example 3.8-3

Consider a transfer function with repeated real poles at $z=1$. That is,

$$G(z) = \frac{z}{(z-1)^2}$$

Determine the stability of the system. The unit-impulse response is given by

$$g(k) = k, \text{ for } k \geq 0$$

Thus,

$$\lim_{k \rightarrow \infty} g(k) = \infty$$

and the system is unstable.

If all inputs are zero, the response of a discrete system due exclusively to initial conditions is referred to as **zero-input response**. According to the zero-input stability criterion,

*a linear, time-invariant discrete system is said to be **zero-input stable**, if its zero-input response goes to zero as k approaches infinity; otherwise, the system is unstable.*

Zero-input stability, also known as **asymptotic stability**, implies that the roots of the characteristic equation are located inside the unit circle of the z -plane. Since this is also a requirement for BIBO stability, it follows that a BIBO stable discrete system is also zero-input or asymptotically stable.

The previous definitions of stability address **absolute stability**, which answers the question of whether the system is stable or unstable. For a stable system, it is often necessary to determine its **relative stability**, that is, to indicate how close the system is to becoming unstable. To determine the relative stability of a system, one alternative is to consider the location of the system poles and their associated time sequences. A summary of this information is shown in Fig. 3.8-1.

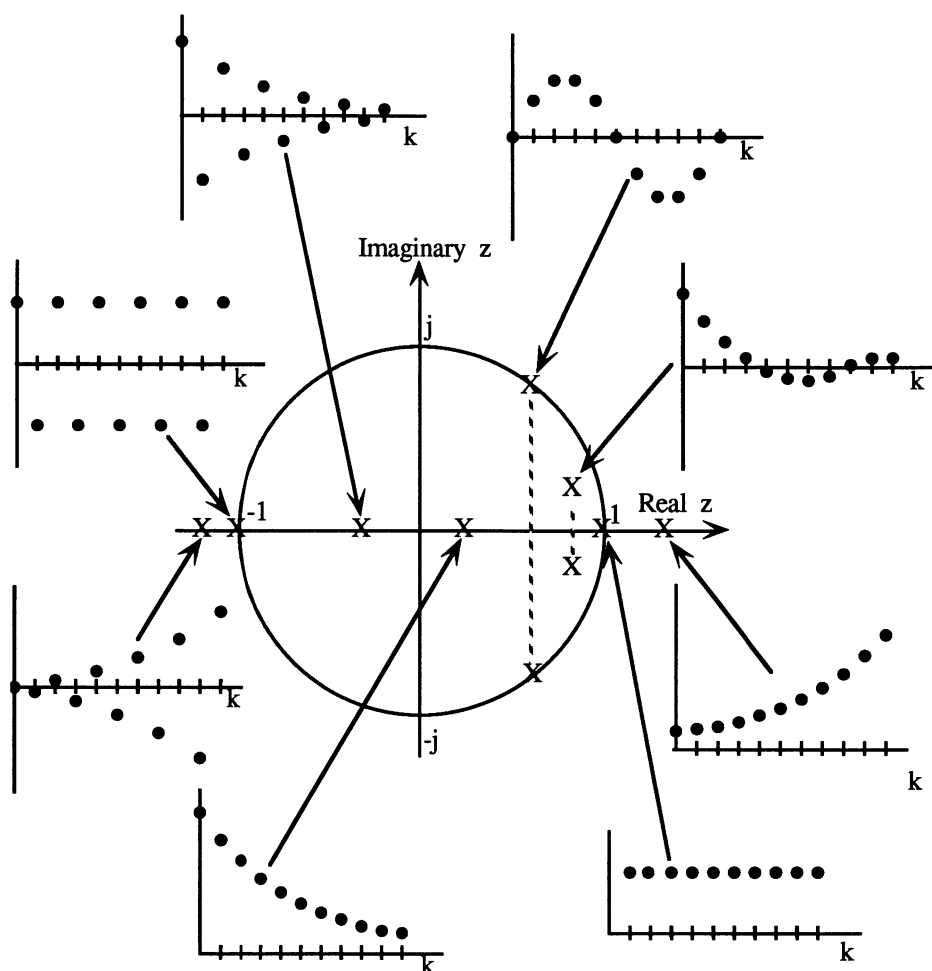


Fig. 3.8-1 Discrete system pole locations (X) and associated time sequences.

It is also helpful to be able to relate the z -plane poles, real or complex, to their s -plane counterparts. For instance, a positive real pole in the z -plane can be readily mapped into its s -plane equivalent using the transformation

$$s_1 = \frac{1}{T} \ln z_1 \quad (3.8-9)$$

Thus, if

$$z_1 = a \quad (3.8-10)$$

then, for positive a

$$s_1 = \frac{1}{T} \ln a \quad (3.8-11)$$

For negative a , however,

$$s_{1,2} = \frac{1}{T} \ln(-a) = \frac{1}{T} \ln a e^{\pm j\pi} = \frac{1}{T} (\ln a \pm j\pi) \quad (3.8-12)$$

Thus, a negative real pole in the z -plane maps into a pair of complex poles in the s -plane. The time response contributed by this negative z -plane pole dies out or decays if $a < 1$, and oscillates or rings if $a = 1$. The pole at $z = -1$ is often referred to as the **ringing pole**.

For complex poles, we can express a second-order polynomial in z as a function of the natural frequency ω_n and damping ratio ζ . Recall that the poles of a second-order continuous system can be written as

$$s_{1,2} = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} \quad (3.8-13)$$

Using the simpler notation

$$s_{1,2} = -a \pm jb \quad (3.8-14)$$

we obtain the real and imaginary components as

$$a = \zeta\omega_n, \text{ and } b = \omega_n\sqrt{1-\zeta^2} \quad (3.8-15)$$

From the denominator of items 9 and 10 in Table 3.5-1, a second-order polynomial in z can be written in the form

$$z^2 - 2ze^{-aT}\cos bT + e^{-2aT} = z^2 - 2ze^{-\zeta\omega_n T}\cos\omega_n\sqrt{1-\zeta^2}T + e^{-2\zeta\omega_n T} \quad (3.8-16)$$

Once the coefficients of the polynomial in z are known, the natural frequency and damping ratio can be calculated. Inversely, if the natural frequency and damping ratio are known, we can calculate the z -plane poles for a sampling time T . This means that the location of the poles and, thus, the stability of a discrete system may depend also on the sampling time T .

3.8-1 Bilinear Transformation

A simple transformation that maps the complex z -plane into the complex s -plane is the one-to-one (unique) bilinear transformation. The bilinear transformation is defined by

$$s = \frac{z-1}{z+1} \quad (3.8-17)$$

Solving the equation for z , we obtain

$$z = \frac{1+s}{1-s} \quad (3.8-18)$$

Substituting $s = \sigma + j\omega$ in the above equation gives

$$z = \frac{(1+\sigma) + j\omega}{(1-\sigma) - j\omega} \quad (3.8-19)$$

and

$$|z|^2 = \frac{(1+\sigma)^2 + \omega^2}{(1-\sigma)^2 + \omega^2} \quad (3.8-20)$$

From the last equation, we make the following observations:

- If $\sigma < 0$, the numerator becomes smaller than the denominator and, thus, $|z| < 1$; this implies that the "z-to-s" transformation maps the left-half s-plane inside the unit circle in the z-plane.
- If $\sigma > 0$, the numerator becomes larger than the denominator and the right-half s-plane maps into the outside of the unit circle in the z-plane.
- If $\sigma = 0$, $|z| = 1$; then the $j\omega$ -axis of the s-plane maps onto the unit circle in the z-plane.
- If, in addition to $\sigma = 0$, $\omega = 0$, then the origin of the s-plane maps into the point $z = 1$ in the z-plane.

The bilinear transformation is similar to the transformation

$$z = e^{sT}, \text{ or } s = \frac{1}{T} \ln z \quad (3.8-21)$$

in that both transformations generate a similar mapping. However, only the bilinear transformation is "unique". Specifically, the bilinear transformation is a one-to-one transformation, while the $s = (\ln z)/T$ transformation is not. The latter transformation is periodic, with multiple-value characteristics. If $z = e^{\alpha T}$, for instance, and $\beta = \alpha \pm jk2\pi/T$, for $k = 0, 1, 2, 3, \dots$, then also $z = e^{\beta T}$. Note that $(2\pi/T)$ is the sampling frequency in rad/sec.

The s-plane to z-plane mapping is illustrated in Fig. 3.8-2. Observe that the s-plane is divided into an infinite number of periodic strips, determined by the sampling frequency ω_s . The range of frequencies between $\pm\omega_s/2$, that is, $-\omega_s/2 \leq \omega \leq \omega_s/2$, defines the **primary strip**. Any point inside the primary strip is mapped uniquely into a point in the z-plane.

It is possible to determine the stability of a system by tests that do not require finding its poles. This can be accomplished by Routh's and Jury's methods.

3.8-2 Routh's Stability Criterion

The bilinear transformation can be applied to determine whether the poles of a discrete system lie inside the unit circle or not. Let the characteristic equation of a discrete system be given by

$$F(z) = a_n z^n + a_{n-1} z^{n-1} + \dots + a_1 z + a_0 = 0 \quad (3.8-22)$$

Then, using the bilinear transformation, we can write the characteristic equation in the form

$$F(s) = a_n \left(\frac{1+s}{1-s} \right)^n + a_{n-1} \left(\frac{1+s}{1-s} \right)^{n-1} + \dots + a_1 \left(\frac{1+s}{1-s} \right) + a_0 = 0 \quad (3.8-23)$$

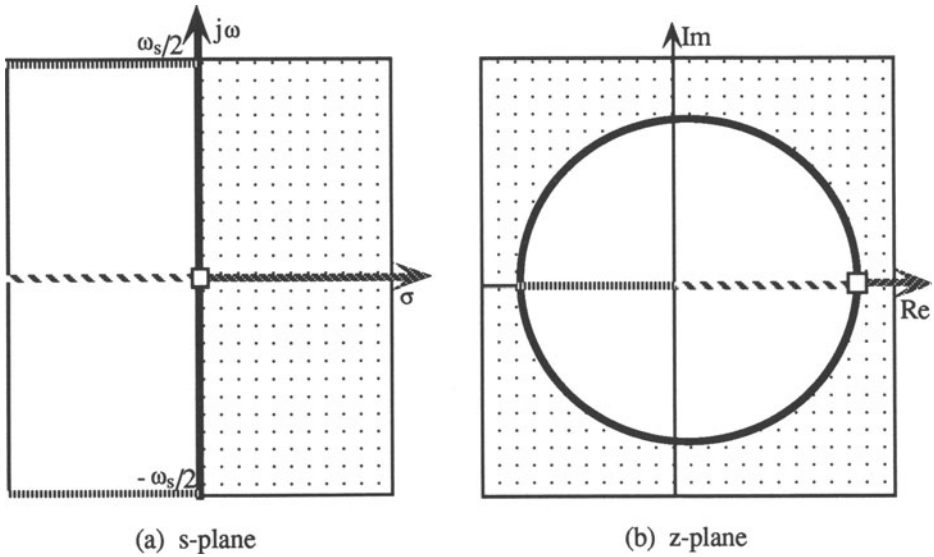


Fig. 3.8-2 Mapping from (a) the s-plane into (b) the z-plane.

Eq. (3.8-23) can be written in the form

$$b_n s^n + b_{n-1} s^{n-1} + \dots + b_1 s + b_0 = 0 \quad (3.8-24)$$

where b is arbitrary. This polynomial equation can then be tested for stability.

The following example illustrates how Routh's criterion, discussed in Chapter 2, can be used to determine the stability of a discrete system.

Example 3.8-4

Use the bilinear transformation to transform the characteristic equation

$$F(z) = 17z^3 + 29z^2 + 27z + 7 = 0$$

into the corresponding $F(s)$. Then, apply Routh's criterion to determine the stability of the system.

The s-plane characteristic equation is given by

$$F(s) = 17 \frac{(1+s)^3}{(1-s)^3} + 29 \frac{(1+s)^2}{(1-s)^2} + 27 \frac{(1+s)}{(1-s)} + 7 = 0$$

After simplification, this gives

$$F(s) = s^3 + 2s^2 + 4s + 10 = 0$$

This system was considered in Example 2.8-1. It has two roots on the right-half s-plane, making the discrete system unstable.

When Routh's stability criterion is applied to higher-order discrete systems, the algebra involved makes the work tedious and impractical. Another criterion which can be applied directly to the characteristic polynomial $F(z)$ is the Jury stability criterion, which is presented next.

3.8-3 The Jury Stability Criterion

Jury's criterion can be used to determine the stability of higher-order discrete systems directly. The criterion is based on the construction of a table using the coefficients of the characteristic equation

$$F(z) = a_n z^n + a_{n-1} z^{n-1} + \dots + a_1 z + a_0 = 0 \quad (3.8-25)$$

where $a_n > 0$. If $a_n < 0$, it can be made positive by multiplying $F(z)$ by -1 .

Jury's Table is shown in Table 3.8-1. The stability constraints are given by the relationships

$$F(1) > 0, \quad (-1)^n F(-1) > 0 \quad (3.8-26)$$

$$\left\{ \begin{array}{l} |a_0| < a_n \\ |b_0| > |b_{n-1}| \\ |c_0| > |c_{n-2}| \\ |d_0| > |d_{n-3}| \\ \vdots \\ |q_0| > |q_2| \end{array} \right\} \quad (n-1) \text{ constraints} \quad (3.8-27)$$

where

$$b_k = \begin{vmatrix} a_0 & a_{n-k} \\ a_n & a_k \end{vmatrix},$$

$$c_k = \begin{vmatrix} b_0 & b_{n-1-k} \\ b_{n-1} & b_k \end{vmatrix},$$

$$d_k = \begin{vmatrix} c_0 & c_{n-2-k} \\ c_{n-2} & c_k \end{vmatrix} \dots$$

$$q_0 = \begin{vmatrix} p_0 & p_3 \\ p_3 & p_0 \end{vmatrix} \dots q_2 = \begin{vmatrix} p_0 & p_1 \\ p_3 & p_2 \end{vmatrix} \quad (3.8-28)$$

Table 3.8-1 Jury's Table

Row	z^0	z^1	z^2	...	z^{n-k}	...	z^{n-1}	z^n
1	a_0	a_1	a_2	...	a_{n-k}	...	a_{n-1}	a_n
2	a_n	a_{n-1}	a_{n-2}	...	a_k	...	a_1	a_0
3	b_0	b_1	b_2	...	b_{n-k}	...	b_{n-1}	
4	b_{n-1}	b_{n-2}	b_{n-3}	...	b_k	...	b_0	
5	c_0	c_1	c_2	...	...	c_{n-2}		
6	c_{n-2}	c_{n-3}	c_{n-4}	...	...	c_0		
..	..	..	..	...	...	..		
2n-5		p_0	p_1	p_2	p_3			
2n-4		p_3	p_2	p_1	p_0			
2n-3		q_0	q_1	q_2				

Jury's criterion states that a necessary and sufficient condition for stability is that the stability constraints listed below must be satisfied. This guarantees that the z-plane poles lie inside the unit circle as required for absolute stability.

The stability constraints obtained from Jury's Table for some commonly used low-order systems are listed below. The constraints are expressed in terms of the coefficients of the respective characteristic equations.

1. First Order ($n=1$): $F(z) = a_1z + a_0 = 0$, $a_1 > 0$

$$\text{Stability constraints: } \left| \frac{a_0}{a_1} \right| < 1 \quad (3.8-29)$$

2. Second Order ($n=2$): $F(z) = a_2z^2 + a_1z + a_0 = 0$, $a_2 > 0$

$$\begin{aligned} \text{Stability constraints: } & a_2 + a_1 + a_0 > 0 \\ & a_2 - a_1 + a_0 > 0 \\ & |a_0| < a_2 \end{aligned} \quad (3.8-30)$$

3. Third Order ($n=3$): $F(z) = a_3z^3 + a_2z^2 + a_1z + a_0 = 0$, $a_3 > 0$

$$\begin{aligned} \text{Stability constraints: } & a_3 + a_2 + a_1 + a_0 > 0 \\ & a_3 - a_2 + a_1 - a_0 > 0 \\ & |a_0| < a_3 \\ & |a_0^2 - a_3^2| > |a_0a_2 - a_1a_3| \end{aligned} \quad (3.8-31)$$

Jury's stability criterion is illustrated next by examples.

Example 3.8-5

Determine the stability of a system whose characteristic equation is

$$F(z) = 17z^3 + 29z^2 + 27z + 7 = 0$$

Applying the stability constraints for $n=3$, we obtain

$$F(1) = 80 > 0$$

Since n is odd,

$$(-1)^3 F(-1) = -F(-1) > 0, \text{ or } F(-1) < 0$$

from which

$$F(-1) = -8 < 0$$

The constraint $|a_0| < a_3$ is also satisfied, because $7 < 17$. The last constraint implies that $|-240| < |-256|$, or $240 > 256$, which fails the test. Thus, the system is unstable.

Example 3.8-6

Find the stability constraints on K_p and K_i of the digital PDF control system considered in Example 3.7-5. The characteristic equation is

$$F(z) = z^2 + (KK_p T + KK_i T - 2)z + (1 - KK_p T) = 0$$

Comparing coefficients with a second-order characteristic equation we have

$$a_0 = 1 - KK_p T, \quad a_1 = (KK_p T + KK_i T - 2), \quad \text{and} \quad a_2 = 1$$

For stability, we must have

$$\begin{aligned} (1) \quad & 1 + KK_p T + KK_i T - 2 + 1 - KK_p T = KK_i T > 0 \\ (2) \quad & 1 - KK_p T - KK_i T + 2 + 1 - KK_p T = 4 - 2KK_p T - KK_i T > 0 \\ \text{and } (3) \quad & |1 - KK_p T| < 1, \text{ or } 0 < KK_p T < 2 \end{aligned}$$

Conditions (1) and (3) suggest that the parameters K , K_i , K_p , and T must be positive. Condition (2) can be rearranged in the form

$$K_p < \frac{(4 - KK_i T)}{2KT}$$

For instance, suppose that the plant gain is set at $K=20$, and that the sampling time is $T=0.005$ sec. Then,

$$K_p < \frac{(40 - K_i)}{2}$$

must be satisfied for stability of the control system. The stability region is shown shaded in Fig. 3.8-2.

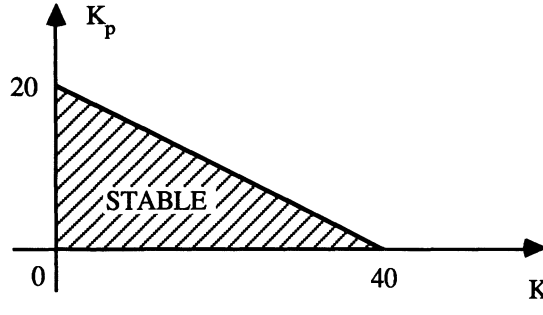


Fig. 3.8-2 K_p vs. K_i plot for the system used in Example 3.8-6.

3.8-4 Root Locus Diagram for Discrete systems

The poles of a discrete system can be determined by factoring the characteristic polynomial in z on a digital computer. A computer can also be used to calculate and plot a root locus diagram that shows the trajectory of the z -plane poles as the value of some adjustable parameter K is varied. If the stability of a discrete system depends on K , the system is said to be **conditionally stable**.

Conditional stability can be established by the root locus method, as illustrated by the following example.

Example 3.8-7

Plot a root locus diagram for a digital PDF control system (see Example 3.7-5) whose open-loop transfer function is given by

$$H(z)G(z) = (K_i) \left(\frac{z}{z-1} \right) \left(\frac{KT}{z-1+KTK_p} \right)$$

Assume the values $K_p=5$, $K=20$, and $T=0.005$ sec. Determine the z -plane poles and plot the corresponding root locus diagram as the integral controller gain K_i varies between 0 and 30.

Substituting the given values in the open-loop transfer function expression, we obtain

$$H(z)G(z) = (K_i) \left(\frac{z}{z-1} \right) \left(\frac{0.1}{z-0.5} \right) = (K_i) \frac{0.1z}{z^2-1.5z+0.5}$$

The gain K_i is varied from 0 to 30 in increments of 1. The z -plane roots are tabulated in Table 3.8-2, and plotted as a root locus diagram in Fig. 3.8-3.

The characteristic equation is given by the relationship

$$1 + H(z)G(z) = 0$$

If $K_i=0$, the z -plane poles are: $z_1 = 0.5$, and $z_2=1$. As K_i increases, the poles move from the right-half z -plane to the left-half z -plane, decreasing the stability of the system. For $K_i>30$, the poles move on the negative real axis and outside the unit circle, resulting in an unstable system.

Consider, for instance, the case $K_i=1$. From Table 3.8-2, the z -plane poles are $z_{1,2}=0.7\pm j0.1$. If these poles are mapped into the s -plane using the transformation $s=(1/T)\ln z$, with $T=0.005$ sec, the corresponding s -plane poles are given by $s_{1,2}=-69.3166\pm j28.38$. This corresponds to a second-order system characterized by a damping ratio $\zeta=0.92$, slightly underdamped, with a natural frequency $\omega_n=74.9$ rad/sec. Therefore, this system compares favorably with the PDF control system used in Example 3.7-5, with $K_i=1.25$.

Table 3.8-2 Data for the root locus diagram shown in Fig. 3.8-3.

K_i	z-plane poles	K_i	z-plane poles
0	1.0000, 0.5000	16	-0.0500 $\pm j$ 0.7053
1	0.7000 $\pm j$ 0.1000	17	-0.1000 $\pm j$ 0.7000
2	0.6500 $\pm j$ 0.2784	17	-0.1000 $\pm j$ 0.7000
3	0.6000 $\pm j$ 0.3742	18	-0.1500 $\pm j$ 0.6910
4	0.5500 $\pm j$ 0.4444	19	-0.2000 $\pm j$ 0.6782
5	0.5000 $\pm j$ 0.5000	20	-0.2500 $\pm j$ 0.6614
6	0.4500 $\pm j$ 0.5454	21	-0.3000 $\pm j$ 0.6403
7	0.4000 $\pm j$ 0.5831	22	-0.3500 $\pm j$ 0.6144
8	0.3500 $\pm j$ 0.6144	23	-0.4000 $\pm j$ 0.5831
9	0.3000 $\pm j$ 0.6403	24	-0.4500 $\pm j$ 0.5454
10	0.2500 $\pm j$ 0.6614	25	-0.5000 $\pm j$ 0.5000
11	0.2000 $\pm j$ 0.6782	26	-0.5500 $\pm j$ 0.4444
12	0.1500 $\pm j$ 0.6910	27	-0.6000 $\pm j$ 0.3742
13	0.1000 $\pm j$ 0.7000	28	-0.6500 $\pm j$ 0.2784
14	0.0500 $\pm j$ 0.7053	29	-0.7000 $\pm j$ 0.1000
15	0.0000 $\pm j$ 0.7071	30	-1.0000, -0.5000

3.9 FREQUENCY RESPONSE OF DISCRETE SYSTEMS

Even though time-domain methods are generally preferred for the analysis and evaluation of discrete systems, the dynamic performance of such systems can also be measured from their frequency response. The frequency-domain approach employs graphical means which become particularly attractive when designing or modifying the plant and/or the algorithm of a control system. In addition to being advantageous in system design, frequency response analysis permits the determination of appropriate passband characteristics. From knowledge of these characteristics, the noise rejection ability of a system can be ascertained.

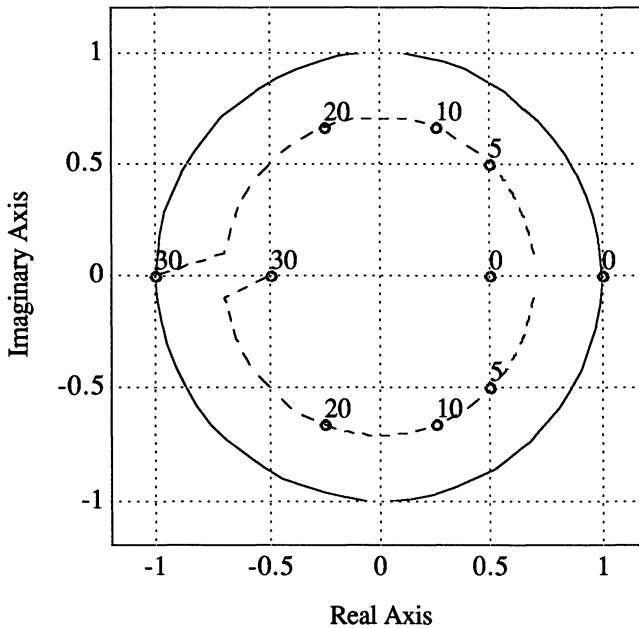


Fig. 3.8-3 The root locus diagram of a digital PDF control system, for $0 \leq K_i \leq 30$. Plotting symbol '-' indicates trajectory of z-plane poles; the symbol 'o' shows the location of z-plane poles and corresponding K_i values.

The first step in the procedure for obtaining frequency response data for a discrete system is to replace z by $e^{j\omega T}$ in the corresponding discrete transfer function. The general concepts relevant to the development of frequency response characteristics for discrete systems parallel those of their continuous counterparts. We illustrate some of these concepts by means of examples.

Determining the frequency response characteristics of second or higher-order discrete systems analytically involves tedious calculations. To make frequency analysis of discrete systems practical and attractive, computer programs, such as those used in this book, should be used. Most of these programs require the discrete transfer function numerator and denominator polynomials, in factored or unfactored form, and the sampling time T . The author's own FREQD program is illustrated by examples.

Example 3.9-1

A continuous system is described by the transfer function

$$G(s) = \frac{a}{s+a}$$

and is preceded by an ideal sampler, as shown in Fig. 3.9-1. Analyze the performance of the overall system.

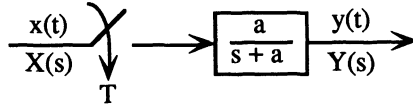


Fig. 3.9-1 The system of Example 3.9-1.

The discrete transfer function of the overall system is

$$G(z) = \frac{Y(z)}{X(z)} = a\mathcal{Z}\left(\frac{1}{s+a}\right) = \frac{az}{z-e^{-aT}}$$

and the corresponding difference equation is

$$y(k) = ax(k) + e^{-aT}y(k-1)$$

Replacing z by $e^{j\omega T}$ in $G(z)$ gives

$$G(j\omega) = \frac{a}{1-e^{-j\omega T}e^{-aT}} = \frac{a}{1-e^{-aT}(\cos\omega T - j\sin\omega T)} = \frac{a}{(1-e^{-aT}\cos\omega T) + je^{-aT}\sin\omega T}$$

Expressing $G(j\omega)$ in terms of magnitude and phase angle components, we obtain

$$G(j\omega) = \frac{a\angle\phi(\omega)}{\sqrt{(1-e^{-aT}\cos\omega T)^2 + e^{-2aT}\sin^2\omega T}} = \frac{a\angle\phi(\omega)}{\sqrt{1+e^{-2aT}-2e^{-aT}\cos\omega T}}$$

where the phase angle function is given by

$$\phi(\omega) = -\tan^{-1}\left[\frac{e^{-aT}\sin\omega T}{1-e^{-aT}\cos\omega T}\right]$$

The magnitude component is given by

$$|G(j\omega)| = \frac{a}{\sqrt{1+e^{-2aT}-2e^{-aT}\cos\omega T}}$$

Observe that at steady state, that is, as the discrete time k approaches infinity (or $\omega=0$), the steady-state value of the transfer function, known also as the **dc gain**, is

$$|G(0)| = \frac{a}{\sqrt{1+e^{-2aT}-2e^{-aT}}} = \frac{a}{1-e^{-aT}}$$

Thus, the dc gain of the discrete transfer function $G(z)$ is not unity, even though the dc gain of the corresponding continuous transfer function $G(s)$ is unity. In addition, the dc gain of the discrete system is a function of the sampling time T .

The discrete system represented in Fig. 3.9-1 is also nonrealizable, because, from its difference equation model, an output $y(k)$ is produced at the same instant that an input $x(k)$ is applied.

Example 3.9-2

Consider the system represented in block diagram form as shown in Fig. 3.9-2. Analyze its performance and compare it to that of the system in Fig. 3.9-1.

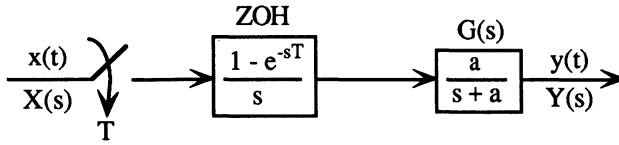


Fig. 3.9-2 The system of Example 3.9-2.

Compared to the last example, the continuous system $G(s)$ is now preceded by a ZOH element. The overall system transfer function is the pulse transfer function

$$\frac{Y(z)}{X(z)} = G_h G(z) = \mathcal{Z}\left[\frac{1 - e^{-sT}}{s} \frac{a}{s + a} \right] = \frac{1 - e^{-aT}}{z - e^{-aT}}$$

The corresponding difference equation is

$$y(k) = (1 - e^{-aT})x(k-1) + e^{-aT}y(k-1)$$

From the difference equation, it can be seen that the system output $y(k)$ depends on previous (past not present) values of the input $x(k)$, making the system realizable. Letting $z = e^{j\omega T}$, we obtain

$$G_h G(j\omega) = \frac{1 - e^{-aT}}{e^{j\omega T} - e^{-aT}} = \frac{1 - e^{-aT}}{\cos\omega T - e^{-aT} + j\sin\omega T}$$

The corresponding magnitude and phase angle functions are

$$|G_h G(j\omega)| = \frac{1 - e^{-aT}}{\sqrt{1 - 2e^{-aT}\cos\omega T + e^{-2aT}}}$$

$$\phi(\omega) = -\tan^{-1}\left[\frac{\sin\omega T}{\cos\omega T - e^{-aT}} \right]$$

At steady state, the dc gain is of this system is

$$|G_h G(0)| = \frac{1 - e^{-aT}}{\sqrt{(1 - e^{-aT})^2}} = 1$$

The unity dc gain of the discrete system is the same as the dc gain of the corresponding continuous system; in addition, this dc gain is not a function of the sampling time T . Essentially, the low-pass filtering characteristics of the ZOH device have helped eliminate the gain problems, caused by the high-frequency components introduced in the system by the ideal sampler.

Example 3.9-3

Let $a=2\pi$ in Example 3.9-2. Find and plot the magnitude and phase versus frequency responses for the system. The transfer function is given by

$$G(s) = \frac{2\pi}{s+2\pi}$$

The pulse transfer function is

$$G_h G(z) = \frac{1-e^{-2\pi T}}{z-e^{-2\pi T}}$$

The magnitude function is

$$|G_h G(j\omega)| = \frac{1-e^{-2\pi T}}{\sqrt{1-2e^{-2\pi T}\cos\omega T + e^{-4\pi T}}}$$

and the phase angle function is

$$\phi(\omega) = -\tan^{-1}\left[\frac{\sin\omega T}{\cos\omega T - e^{-2\pi T}}\right]$$

To determine and plot the magnitude and phase responses, we fix the sampling time T , and vary ω in the range of interest. Assume that the corner frequency is also the highest frequency, that is, let

$$\omega_h = 2\pi$$

From the sampling theorem,

$$\omega_s = \frac{2\pi}{T} \geq 2\omega_h$$

Thus, the minimum value of T is

$$T < \frac{\pi}{\omega_h} = \frac{\pi}{2\pi} = \frac{1}{2} \text{ sec}$$

Since the magnitude $|G(j\omega)|$ versus ω plot of a first-order system does not "cutoff" sharply at the break frequency, the frequency $\omega_h=2\pi$ is not really the highest frequency component present in our example system. Therefore, in order to account for frequencies higher than ω_h and thus obtain acceptable results, a sampling frequency higher than twice the highest frequency ω_h must be chosen.

"Rules of thumb" specify the usage of sampling frequencies which are 10-20 times the highest frequency ω_h . Magnitude plots versus ω for $T=0.5$ sec and $T=0.05$ sec are shown in Fig. 3.9-3. The magnitude of the continuous transfer function $G(s)$ has also

been plotted for comparison. Notice that if $T=0.05$ sec, the discrete system approximates the continuous system reasonably well, while if $T=0.5$ sec, it does not.

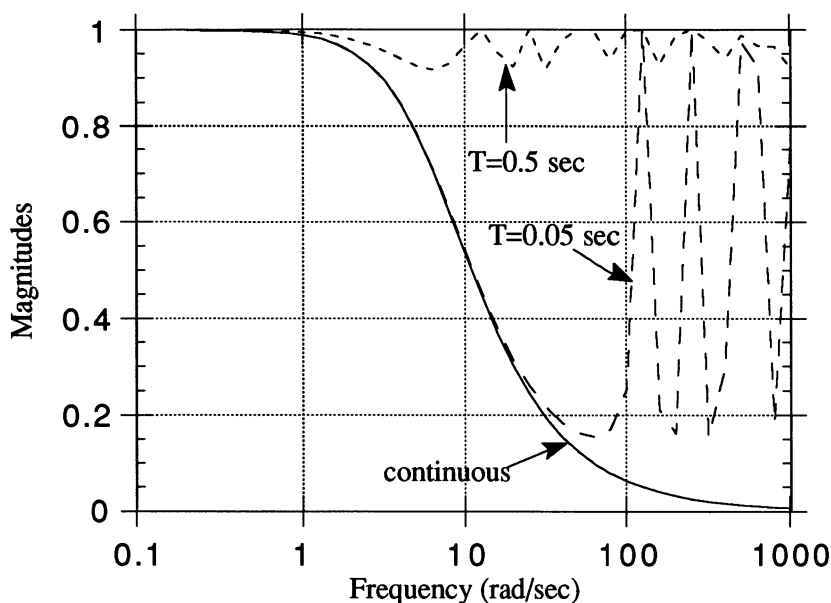


Fig. 3.9-3 Magnitude vs. frequency plots for the $G(s)$ and $G_h G(z)$ functions used in Example 3.9-2.

Example 3.9-4

Determine the magnitude and phase versus frequency responses for the closed-loop transfer function $C(z)/R(z)$ of the digital PDF control system used in Example 3.7-5. Assume the values: $K=20$, $K_i=1.25$, $K_p=5$, and $T=0.005$ sec. The $C(z)/R(z)$ transfer function is

$$\frac{C(z)}{R(z)} = \frac{KK_i Tz}{z^2 + (KK_p T + KK_i T - 2)z + (1 - KK_p T)}$$

Substituting the given parameter values, we obtain the discrete transfer function

$$\frac{C(z)}{R(z)} = \frac{0.125z}{z^2 - 1.375z + 0.5}$$

Results obtained using the program FREQD, for the frequency range 10-1000 rad/sec, are shown in Table 3.9-1. As can be seen from the table, the 3-db point for this

system occurs at about 70 rad/sec. Magnitude and phase versus frequency plots are shown in Fig. 3.9-4.

Since $T=0.005$ sec, the sampling frequency is 200 Hz (1257 rad/sec). Thus, the folding frequency is given by

$$\omega_o = \frac{\omega_s}{2} = 628 \text{ rad/sec}$$

At this frequency, the magnitude function attains its lowest value before it "folds"; that is, after ω_o , the magnitude function starts increasing and repeats itself at $\omega = \omega_s, 2\omega_s$, etc.

Example 3.9-5

Determine the magnitude and phase versus frequency responses for the open-loop transfer function $H(z)G(z)$ of the digital PDF control system used in Example 3.7-5. Assume the values: $K=20$, $K_i=1.25$, $K_p=5$, and $T=0.005$ sec. The $H(z)G(z)$ transfer function is

$$\frac{C(z)}{E(z)} = H(z)G(z) = \frac{KK_iTz}{(z-1)(z-1+KK_pT)}$$

Substituting the given values, we get

$$H(z)G(z) = \frac{0.125z}{z^2 - 1.5z + 0.5}$$

The computer results for the frequency response of the open-loop transfer function $H(z)G(z)$ are shown in Table 3.9-2. The phase margin is about 70° (i.e., $180-110$), and the gain margin, measured at zero db gain, is about 27 db. A plot of the magnitude and phase versus frequency data is shown in Fig. 3.9-5.

3.10 DIGITAL SIMULATION AND $\mathcal{Z}$ TRANSFORMS

We have seen that the $\mathcal{Z}$ transform method can facilitate the solution of difference equations. A difference equation is a recursive relation that can be easily programmed on a digital computer. This makes computer simulation of discrete systems practical and attractive.

An important objective of this section is to develop a systematic procedure for approximating the solution of differential equations by difference equations; this would enable us to carry out the solution of continuous systems on digital computers. A related objective is to develop numerical integration and differentiation rules which are essential for system simulation and for implementing computer control algorithms.

3.10-1 Numerical Differentiation and Integration

It was shown in Example 3.2-1 that the derivative of $y(t)$ can be approximated at the sampling instants $t=kT$ by the **forward difference** formula

$$y'(t) \Big|_{t=kT} = y'(kT) = y'(k) \approx \frac{y[(k+1)T] - y(kT)}{T} = \frac{y(k+1) - y(k)}{T} \quad (3.10-1)$$

Table 3.9-1 Frequency response characteristics for the closed-loop transfer function $C(z)/R(z)$ used in Example 3.9-4.

OMEGA	REAL	IMAGINARY	MAGNITUDE	DECIBELS	DEGREES
10.00	0.9751	-0.1979	0.9949	-0.0441	-11.4730
12.59	0.9605	-0.2476	0.9919	-0.0704	-14.4531
15.85	0.9376	-0.3085	0.9871	-0.1130	-18.2132
19.95	0.9016	-0.3820	0.9792	-0.1826	-22.9609
25.12	0.8455	-0.4679	0.9663	-0.2977	-28.9585
31.62	0.7594	-0.5625	0.9450	-0.4911	-36.5273
39.81	0.6317	-0.6548	0.9099	-0.8202	-46.0292
50.12	0.4548	-0.7215	0.8529	-1.3822	-57.7777
63.10	0.2389	-0.7272	0.7655	-2.3213	-71.8114
79.43	0.0275	-0.6451	0.6457	-3.7989	-87.5597
100.00	-0.1203	-0.4920	0.5065	-5.9079	-103.7429
125.89	-0.1797	-0.3255	0.3718	-8.5944	-118.8987
158.49	-0.1747	-0.1932	0.2604	-11.6856	-132.1122
199.53	-0.1427	-0.1068	0.1782	-14.9799	-143.2003
251.19	-0.1079	-0.0563	0.1217	-18.2944	-152.4281
316.23	-0.0796	-0.0286	0.0846	-21.4533	-160.2234
398.11	-0.0598	-0.0137	0.0613	-24.2463	-167.0540
501.19	-0.0478	-0.0055	0.0481	-26.3580	-173.4415
630.96	-0.0435	0.0001	0.0435	-27.2342	-180.1315
794.33	-0.0511	0.0079	0.0517	-25.7223	-188.7863
1000.00	-0.1049	0.0530	0.1175	-18.5981	-206.7919

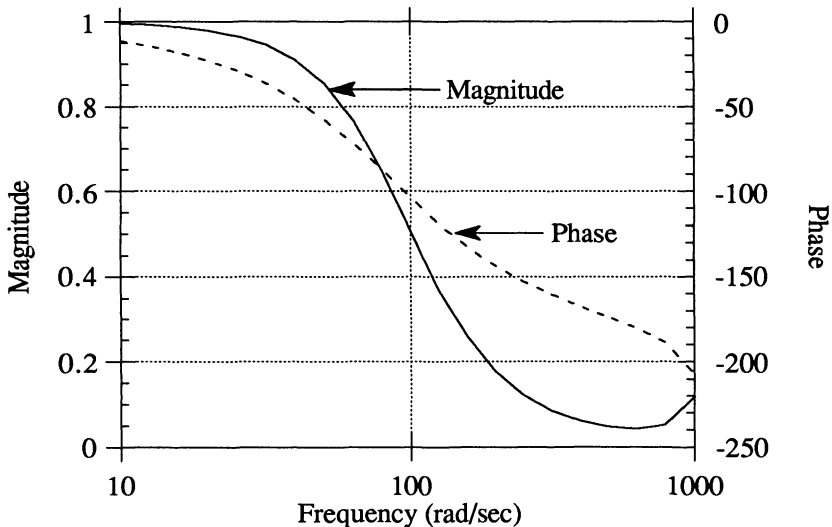
**Fig. 3.9-4** Magnitude and phase characteristics for $C(z)/R(z)$ used in Example 3.9-4.

Table 3.9-2 Frequency response characteristics for the open-loop transfer function $H(z)G(z)$ used in Example 3.9-4.

OMEGA	REAL	IMAGINARY	MAGNITUDE	DECIBELS	DEGREES
10.00	-0.3731	-4.9741	4.9881	13.9587	-94.2900
12.59	-0.3721	-3.9391	3.9567	11.9466	-95.3956
15.85	-0.3704	-3.1140	3.1360	9.9275	-96.7822
19.95	-0.3677	-2.4550	2.4824	7.8975	-98.5178
25.12	-0.3635	-1.9272	1.9612	5.8504	-100.6826
31.62	-0.3572	-1.5029	1.5447	3.7770	-103.3692
39.81	-0.3475	-1.1602	1.2111	1.6636	-106.6766
50.12	-0.3334	-0.8822	0.9431	-0.5091	-110.7000
63.10	-0.3132	-0.6563	0.7272	-2.7670	-115.5096
79.43	-0.2860	-0.4737	0.5533	-5.1406	-121.1194
100.00	-0.2517	-0.3286	0.4140	-7.6608	-127.4530
125.89	-0.2123	-0.2173	0.3038	-10.3482	-134.3234
158.49	-0.1711	-0.1363	0.2188	-13.1997	-141.4527
199.53	-0.1325	-0.0811	0.1553	-16.1765	-148.5382
251.19	-0.0997	-0.0458	0.1097	-19.1954	-155.3388
316.23	-0.0744	-0.0245	0.0783	-22.1217	-161.7420
398.11	-0.0566	-0.0122	0.0579	-24.7513	-167.7968
501.19	-0.0456	-0.0050	0.0459	-26.7636	-173.7419
630.96	-0.0417	0.0001	0.0417	-27.6039	-180.1260
794.33	-0.0487	0.0072	0.0492	-26.1558	-188.3554
1000.00	-0.0970	0.0433	0.1062	-19.4746	-204.0472

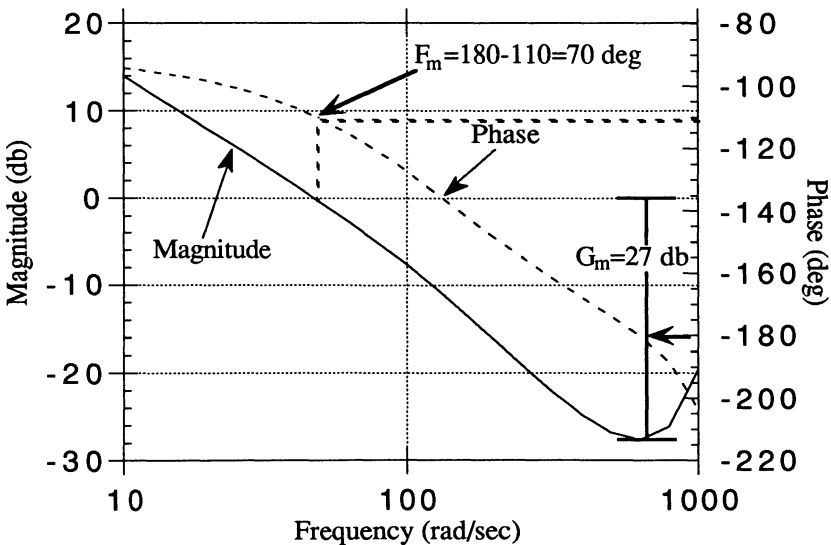


Fig. 3.9-5 Magnitude and phase characteristics for $H(z)G(z)$ used in Example 3.9-5.

Taking the $\mathcal{Z}$ transform on both sides of the equation gives

$$\mathcal{Z}[y'(k)] = \frac{(z-1)y(k)}{T} = \left(\frac{z-1}{T}\right)Y(z) \quad (3.10-2)$$

Expressing the **numerical differentiator** in transfer function form, we obtain

$$D_F(z) = \frac{z-1}{T} = \frac{1-z^{-1}}{Tz^{-1}} \quad (3.10-3)$$

where D and F denote differentiation (D) and forward (F) difference, respectively. If the input to the differentiator is a uniformly sampled signal $x(k)$, then, from the transfer function relationship, the numerical differentiation process can be implemented by the corresponding difference equation

$$y(k) = \frac{1}{T} [x(k+1) - x(k)] \quad (3.10-4)$$

where $y(k)$ is the output of the differentiator. Similarly, the second derivative of $y(t)$ can be approximated by two differentiators in cascade, as shown in Fig. 3.10-1. From the figure, the second derivative transfer function is given by

$$D_F^2(z) = \frac{(z-1)^2}{T^2} \quad (3.10-5)$$

For an n th derivative, the transfer function is

$$D_F^n(z) = \frac{(z-1)^n}{T^n} = \frac{(1-z^{-1})^n}{T^n z^{-n}} \quad (3.10-6)$$

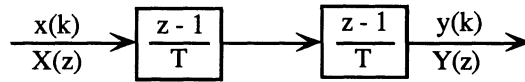


Fig. 3.10-1 Second derivative of $x(k)$ using forward differences.

If **numerical integration** rather than differentiation is desired, we simply inverse the derivative transfer function relationships. That is, the transfer function for integrating an input $x(k)$ n times is given by

$$I_F^n(z) = \frac{1}{D_F^n(z)} \quad (3.10-7)$$

where the symbol I denotes integration. For instance, the forward integration transfer function is given by

$$I_F(z) = \frac{T}{z-1} = \frac{Tz^{-1}}{1-z^{-1}} \quad (3.10-8)$$

If the integrator output is denoted by $y(k)$, the corresponding difference equation is

$$y(k) = y(k-1) + Tx(k-1) \quad ; k = 1, 2, \dots \quad (3.10-9)$$

Equivalently,

$$y(k+1) = y(k) + Tx(k) \quad ; k = 0, 1, 2, \dots \quad (3.10-10)$$

The above equations represent the **rectangular**, also known as **Euler's integration rule**. Euler's integration rule can be expressed in block diagram form as shown in Fig. 3.10-2.

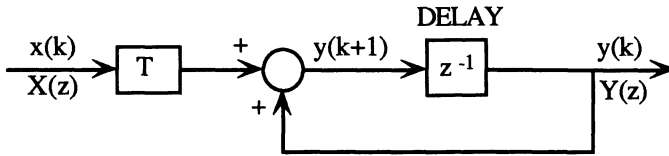


Fig. 3.10-2 Rectangular integration: $y(k+1) = y(k) + T x(k)$.

Like differentiators, integrators can also be connected in series (cascade). For instance, the double integral of the function $x(k)$ can be represented in block diagram form as shown in Fig. 3.10-3. The concepts are illustrated by means of an example.

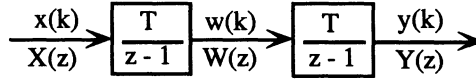


Fig. 3.10-3 Cascaded numerical integrators.

Example 3.10-1

If $x(k)$ represents the piecewise constant approximation of the unit-step function $u(k)$, determine the first integral $w(k)$, and the second integral $y(k)$, for the system shown in Fig. 3.10-3.

Since $x(k)=u(k)=1$, for $k = 0, 1, 2, \dots$, the output of the first integrator can be calculated using Euler's rule, that is,

$$w(k) = w(k-1) + Tx(k-1)$$

Similarly, the output of the second integrator can be determined from

$$y(k) = y(k-1) + Tw(k-1)$$

The integration results are shown in Table 3.10-1.

Table 3.10-1 Rectangular integration results for Example 3.10-1

k	kT	x(kT)	w(kT)	y(kT)	t ² /2
0	0	1	0	0	0
1	T	1	T	0	T ² /2
2	2T	1	2T	T ²	2T ²
3	3T	1	3T	3T ²	9T ² /2
4	4T	1	4T	6T ²	8T ²
5	5T	1	5T	10T ²	25T ² /2
:	:	:	:	:	:

Note that the integral of the unit-step function is the unit-ramp function t , and that the integral of t is the parabolic function $t^2/2$. Also, at the sampling instants, the first (forward) rectangular integrator integrates the piecewise constant function $x(k)$ exactly, giving the piecewise linear function $w(k)$.

The second rectangular integrator introduces an error, however, because it assumes that $w(k)$ can also be approximated by a piecewise constant function. To verify this, compare the $y(k)$ values in Table 3.10-1 with the values obtained by evaluating $t^2/2$ at the sampling instants, also shown in the table. To develop a rule that integrates the piecewise linear function $w(k)$ exactly, consider the following approach.

An alternative to the forward difference approximation for the derivative of $y(t)$ at $t=kT$ is the **backward difference** approximation. This approximation is given by

$$y'(k) \approx \frac{y(k) - y(k-1)}{T} \quad (3.10-11)$$

Applying $\mathcal{Z}$ transforms results in the following transfer function representation for the backward differentiator

$$D_B(z) = \frac{z-1}{Tz} = \frac{1-z^{-1}}{T} \quad (3.10-12)$$

where the subscript B denotes backward difference approximation. The corresponding difference equation is

$$y(k) = \frac{x(k) - x(k-1)}{T} \quad (3.10-13)$$

where $x(k)$ represents the input and $y(k)$ the output of the differentiator block. Using backward differences, the transfer function of an n th derivative can be written as

$$D_B^n(z) = \frac{(z-1)^n}{T^n z^n} = \frac{(1-z^{-1})^n}{T^n} \quad (3.10-14)$$

The analogous transfer function for an n th (backward) integral is

$$I_B^n(z) = \frac{T^n z^n}{(z-1)^n} = \frac{T^n}{(1-z^{-1})^n} \quad (3.10-15)$$

Therefore, the (backward) difference equation for integrating $x(k)$ once is

$$y(k) = y(k-1) + Tx(k) ; k = 0, 1, 2, \dots \quad (3.10-16)$$

A rule for integrating piecewise linear functions exactly is obtained when the results for the forward and backward integration rules are averaged. The new integration rule is called **trapezoidal**, because the integral of the approximation is the area of a trapezoid, rather than a rectangle. Using the forward and backward integration transfer functions, we obtain the transfer function for the trapezoidal integration rule as

$$I_T(z) = \frac{1}{2} [I_F(z) + I_B(z)] = \frac{1}{2} \left(\frac{T}{z-1} + \frac{Tz}{z-1} \right) = \frac{T}{2} \left(\frac{z+1}{z-1} \right) \quad (3.10-17)$$

where the subscript T denotes trapezoidal integration. The corresponding difference equation form of trapezoidal integration is

$$y(k) = y(k-1) + \frac{T}{2} [x(k) + x(k-1)] \quad (3.10-18)$$

The trapezoidal rule, also called **Tustin's transformation**, is equivalent to the bilinear transformation presented in Section 3.8-1.

Since the quantity $T[x(k)+x(k-1)]/2$ represents the area of a trapezoid, one may use this concept to derive the trapezoidal integration rule. Alternatively, the trapezoidal rule can be derived from the fundamental relationship

$$z = e^{sT}, \text{ or } s = \frac{1}{T} \ln z$$

Expanding $\ln z$ into its **Maclaurin series** gives

$$s = \frac{2}{T} \left[\frac{z-1}{z+1} + \frac{1}{3} \frac{(z-1)^3}{(z+1)^3} + \frac{1}{5} \frac{(z-1)^5}{(z+1)^5} + \dots \right] ; \text{ for } |z| > 0 \quad (3.10-19)$$

Tuncating the series after the first term and inverting the result, we obtain the trapezoidal integration transfer function

$$I_T(z) = \left(\frac{T}{2} \right) \left(\frac{z+1}{z-1} \right) \quad (3.10-20)$$

Example 3.10-2

Solve the problem of Example 3.10-1 by performing the second integration using trapezoidal integration.

The problem can be represented as shown in Fig. 3.10-4. The computed values are shown in Table 3.10-2.

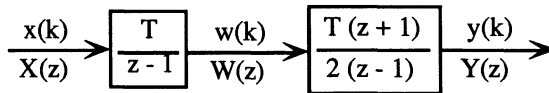


Fig. 3.10-4 Integration diagram for Example 3.10-2.

Table 3.10-2 *Integration results for Example 3.10-2*

k	kT	x(kT)	w(kT)	y(kT)	t ² /2
0	0	1	0	0	0
1	T	1	T	T ² /2	T ² /2
2	2T	1	2T	2T ²	2T ²
3	3T	1	3T	9T ² /2	9T ² /2
4	4T	1	4T	8T ²	8T ²
5	5T	1	5T	25T ² /2	25T ² /2
⋮	⋮	⋮	⋮	⋮	⋮

Note how the output values, denoted by $y(kT)$, match the exact values obtained by evaluating $t^2/2$ at $t = kT$. The example demonstrates that the trapezoidal rule will integrate a piecewise linear function exactly. This implies that to integrate exactly a piecewise constant function twice, we must first integrate it using (forward/Euler's) rectangular integration, and then integrate again using the trapezoidal rule. Using transfer function relations, the transfer function required to accomplish this is

$$I^2(z) = I_F(z)I_T(z) = \left(\frac{T}{z-1}\right)\left(\frac{T}{2}\right)\left(\frac{z+1}{z-1}\right) = \left(\frac{T^2}{2}\right)\frac{(z+1)}{(z-1)^2} \quad (3.10-21)$$

Consider the numerical integration and differentiation concepts presented earlier, but with a different approach. Recall that the output of the zero-order-hold (ZOH) element is a piecewise constant function. Thus, the simple integration process can be expressed in block diagram form as shown in Fig. 3.10-5.

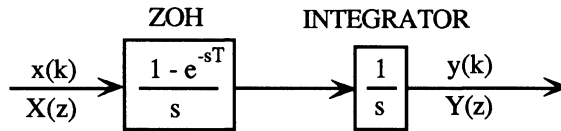


Fig. 3.10-5 Integration of a piecewise constant function.

From the block diagram, the pulse transfer function is

$$I_1(z) = \frac{Y(z)}{X(z)} = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{1}{s}\right] = \frac{T}{z-1} \quad (3.10-22)$$

which is the transfer function for (forward) rectangular or Euler integration rule.

Using this approach, we can show that to integrate exactly a piecewise constant function twice, the appropriate transfer function is

$$I_2(z) = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{1}{s^2}\right] = \frac{T^2}{2} \frac{(z+1)}{(z-1)^2} \quad (3.10-23)$$

For integrating the piecewise constant function exactly three times, the required transfer function is

$$I_3(z) = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{1}{s^3}\right] = \frac{T^3}{6} \frac{(z^2+4z+1)}{(z-1)^3} \quad (3.10-24)$$

This transfer function can be derived by using Theorem 6 and Table 3.5-1.

Next, consider the block diagram shown in Fig. 3.10-6. Assuming that $x(k)$ is a piecewise constant function, suppose that we wish to determine the transfer function $I_L(z)$ which integrates exactly the piecewise linear function $w(k)$. Using the expression for $I_2(z)$ derived above, we obtain

$$\frac{T}{(z-1)} I_L(z) = I_2(z) = \frac{T^2}{2} \frac{(z+1)}{(z-1)^2} \quad (3.10-25)$$

where the subscript L denotes linear integration. Solving for $I_L(z)$, we get

$$I_L(z) = \frac{T}{2} \frac{(z+1)}{(z-1)} \quad (3.10-26)$$

which is the same as the trapezoidal rule $I_T(z)$.

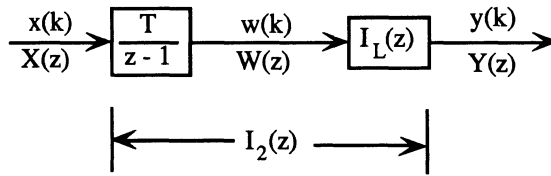


Fig. 3.10-6 Double integration of a piecewise constant function.

To carry this approach another step further, assume that we wish to integrate exactly a piecewise quadratic function. Working in a similar fashion as before, the problem is to determine $I_Q(z)$ from the equation

$$I_F(z)I_T(z)I_Q(z) = I_2(z)I_Q(z) = I_3(z) \quad (3.10-27)$$

where Q stands for **quadratic**, and the transfer functions $I_F(z)$ and $I_T(z)$ have already been defined. After substituting the known quantities, we obtain

$$\left(\frac{T^2}{2}\right) \frac{(z+1)}{(z-1)^2} I_Q(z) = \frac{T^3}{6} \frac{(z^2+4z+1)}{(z-1)^3} \quad (3.10-28)$$

Thus the transfer function for integrating piecewise quadratic signals exactly is

$$I_Q(z) = \frac{T}{3} \frac{(z^2+4z+1)}{(z-1)(z+1)} = \frac{T}{3} \frac{(z^2+4z+1)}{(z^2-1)} \quad (3.10-29)$$

This transfer function represents the well-known **Simpson's one-third (1/3) integration rule**. The corresponding difference equation form of Simpson's rule is

$$y(k) = y(k-2) + \frac{T}{3} [x(k) + 4x(k-1) + x(k-2)] \quad (3.10-30)$$

The previous method for developing numerical integration (or differentiation) rules can be extended indefinitely. One may prove, for instance, that the rule for integrating piecewise cubic signals exactly, call it the **one-fourth (1/4) integration rule**, is given by the transfer function

$$I_C(z) = \frac{T}{4} \frac{(z^2+10z+1)(z+1)}{(z-1)(z^2+4z+1)} = \frac{T}{4} \frac{(z^3+11z^2+11z+1)}{(z^3+3z^2-3z-1)} \quad (3.10-31)$$

where the subscript C denotes cubic. The corresponding difference equation form of the (1/4) integration rule is

$$y(k) = -3y(k-1)+3y(k-2)+y(k-3) + \frac{T}{4} [x(k)+11x(k-1)+11x(k-2)+x(k-3)] \quad (3.10-32)$$

3.10-2 Numerical Solution of Differential Equations

The objective in this section is to apply the numerical integration and differentiation rules developed earlier to generate a systematic procedure for approximating the solution of a differential equation by a recursive difference equation. The procedure is illustrated by the following example.

Example 3.10-3

Consider the standard-form second-order system described by the differential equation

$$y''(t) + 2\zeta\omega_n y'(t) + \omega_n^2 y(t) = \omega_n^2 x(t)$$

We wish to simulate the system on a computer using rectangular (forward) integration. The input sequence $x(k)$, damping ratio ζ , natural frequency ω_n , and appropriate initial conditions are specified.

First express the given equation in either derivative or integral operator form. Using the derivative operator 's', for instance, we get

$$(s^2 + 2\zeta\omega_n s + \omega_n^2) Y(s) = \omega_n^2 X(s)$$

Alternatively, if the integral operator $1/s$, or s^{-1} , is used, we obtain

$$(1 + 2\zeta\omega_n s^{-1} + \omega_n^2 s^{-2}) Y(s) = \omega_n^2 s^{-2} X(s)$$

Assuming that both $y(t)$ and $x(t)$ can be integrated by rectangular integration, we have

$$[1 + 2\zeta\omega_n \frac{T}{(z-1)} + \omega_n^2 \frac{T^2}{(z-1)^2}] Y(z) = \omega_n^2 \frac{T^2}{(z-1)^2} X(z)$$

Simplifying gives

$$[z^2 + (2\zeta\omega_n T - 2)z + (\omega_n^2 T^2 - 2\zeta\omega_n T + 1)] Y(z) = \omega_n^2 T^2 X(z)$$

Dividing both sides of the equation by z^2 and rearranging results in the difference equation

$$y(k) = 2(1 - \zeta\omega_n T)y(k-1) + (2\zeta\omega_n T - \omega_n^2 T^2 - 1)y(k-2) + \omega_n^2 T^2 x(k-2)$$

The second-order difference equation approximates the solution of the given differential equation at the sampling instants, with appropriate initial conditions. For instance, if the iteration begins with $k=1$, the required initial conditions are $y(0)$ and $y(-1)$. These initial conditions must then be expressed in terms of the differential equation initial conditions $y(0)$ and $y'(0)$. To determine $y(-1)$, recall that

$$y'(k) \approx \frac{y(k) - y(k-1)}{T} = \frac{(1 - z^{-1})}{T} Y(z)$$

If $k=0$, we obtain

$$y'(0) = \frac{y(0) - y(-1)}{T}$$

and the desired initial condition is

$$y(-1) = y(0) - Ty'(0)$$

Note that if a third initial condition is needed, the appropriate relationship to consider is

$$y''(k) = \frac{(1 - z^{-1})^2}{T^2} Y(z)$$

This leads to the initial condition

$$y(-2) = T^2 y''(0) - 2Ty'(0) + y(0)$$

A BASIC language simulation program that implements the difference equation is shown in Fig. 3.10-7. The program is set up to handle only step inputs, but it can be easily modified for other types of input sequences as well as for higher-order recursive relations.

The simulation scheme is based on a similar scheme introduced in Section 2.10-2 for continuous systems. The comments made in section 2.10 concerning selection of the integration time T apply here also. Specifically, if T is too large, the results will be unpredictable, and if T is too small, computational time and errors may become excessive.

Example 3.10-4

Use Tustin's method (trapezoidal integration) to determine the recursive relation required to simulate the differential equation

$$y''(t) + 2\zeta\omega_n y'(t) + \omega_n^2 y(t) = \omega_n^2 x(t)$$

Transforming the differential equation, and substituting the integration transfer function $I_T(z)$ developed earlier for trapezoidal integration, we obtain

```

100 PRINT"SIMULATION OF A SECOND-ORDER DIFFERENCE EQUATION:"
110 PRINT"y(k)=-B1*y(k-1)-B2*y(k-2)+A0*x(k)+A1*x(k-1)+A2*x(k-2).\"
140 PRINT"NOTE: FINTIM=FINISH TIME, TAU=INTEGRATION TIME,\"
150 PRINT"NUMOUT=NUMBER OF OUTPUT VALUES FOR SIMULATION.\"
170 INPUT"ENTER ZETA(Z), AND OMEGAN(W):\"; Z, W
180 INPUT"ENTER FINTIM, TAU, AND NUMOUT:\"; T2, T0, N0
190 INPUT"ENTER X0, THE SIZE OF THE STEP INPUT:\"; X0
200 INPUT"ENTER INITIAL CONDITIONS: y(0), y'(0):\"; Y1, Y2
210 Y2=Y1-T0*Y2
230 N1=T2/T0; N2=T2/N0
240 X1=0; X2=0
250 T1=N2-0.5*T0
260 B1=2*(Z*W*T0-1); B2=1+W*W*T0*T0-2*Z*W*T0
270 A0=0; A1=0; A2=W*W*T0*T0
280 T=0; Y0=-B1*Y1-B2*Y2+A0*X0+A1*X1+A2*X2
290 Y2=Y1; Y1=Y0; X2=X1; X1=X0
300 PRINT"Time", "Output\"
310 PRINT"=====\", \"=====\"
320 PRINT T, Y0
330 N=1
340 FOR K=1 TO N1+1
350     T=K*T0
360     Y0=-B1*Y1-B2*Y2+A0*X0+A1*X1+A2*X2
370     Y2=Y1; Y1=Y0; X2=X1; X1=X0
380     IF T<T1 THEN 420
390     N=N+1
400     PRINT T, Y0
410     T1=N*N2-0.5*T0
420 NEXT K
430 END

```

Fig. 3.10-7 Listing of program used to simulate a second-order difference equation.

$$\left[1 + 2\zeta\omega_n \frac{T}{2} \frac{(z+1)}{(z-1)} + \omega_n^2 \frac{T^2}{4} \frac{(z+1)^2}{(z-1)^2}\right] Y(z) = \omega_n^2 \frac{T^2}{4} \frac{(z+1)^2}{(z-1)^2} X(z)$$

After some manipulation, the resulting recursive relation is

$$y(k) = \frac{\left(2 - \frac{\omega_n^2 T^2}{2}\right)}{D} y(k-1) + \frac{\left(\zeta\omega_n T - 1 - \frac{\omega_n^2 T^2}{4}\right)}{D} y(k-2) + \frac{\omega_n^2 T^2}{4D} [x(k) + 2x(k-1) + x(k-2)]$$

where

$$D = 1 + \zeta\omega_n T + \frac{\omega_n^2 T^2}{4}$$

Example 3.10-5

A standard-form second-order system, described by the differential equation of Example 3.10-4, has damping ratio $\zeta=0.5$, and natural frequency $\omega_n=10$ rad/sec. Determine and

plot the unit-step response of the system if $T=0.04$ sec. Assume zero initial conditions. Solve the problem by each of the approximation methods: (a) forward difference, (b) backward difference, and (c) trapezoidal. For comparison, include a plot of the unit-step response of the system obtained analytically.

The general recursive expressions for the forward difference and trapezoidal approximations were developed earlier. It can be shown that the relation for the backward difference approximation case is given by

$$y(k) = \frac{2(1+\zeta\omega_n T)}{D} y(k-1) - \frac{1}{D} y(k-2) + \frac{\omega_n^2 T^2}{D} x(k)$$

where

$$D = (1+2\zeta\omega_n T + \omega_n^2 T^2)$$

The required step responses are shown plotted in Fig. 3.10-8.

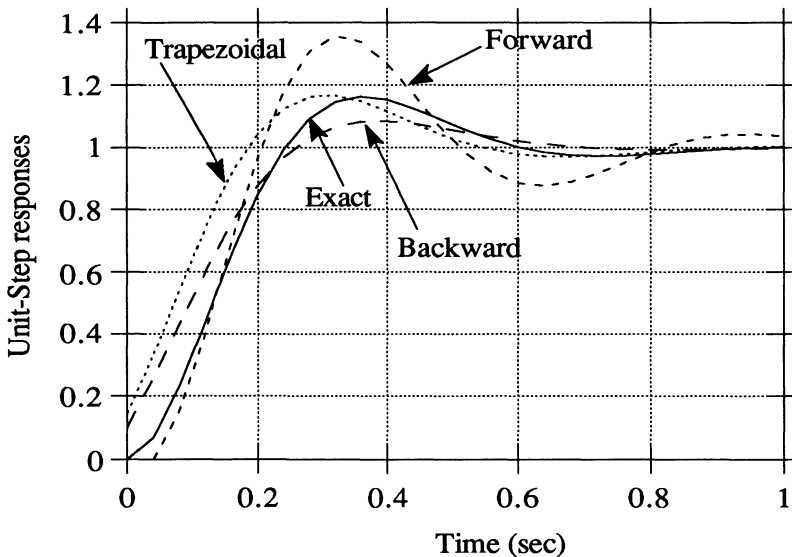


Fig. 3.10-8 Comparison of forward difference, backward difference, and trapezoidal integration methods for system used in Example 3.10-5.

Example 3.10-6

Determine a recursive relation to approximate the unit-step response of the system described by

$$y''(t) + 10y'(t) + 100y(t) = 100x(t)$$

assuming zero initial conditions. Assume that $x(t)$ and $y(t)$ are piecewise constant.

Using the derivative operator notation, we can rewrite the equation as

$$(s^2 + 10s + 100) Y(s) = 100 X(s)$$

Using integrators, we obtain

$$(1 + 10s^{-1} + 100s^{-2}) Y(s) = 100s^{-2} X(s)$$

Since we can integrate exactly a piecewise constant function twice, once by rectangular integration, and once by trapezoidal integration, the above equation can be written in the form

$$\left[1 + 10 \frac{T}{z-1} + 100 \frac{T^2}{2} \frac{(z+1)}{(z-1)^2}\right] Y(z) = 100 \left[\frac{T^2}{2} \frac{(z+1)}{(z-1)^2}\right] X(z)$$

Simplifying and rearranging, we obtain the desired recursive relation

$$y(k) = (2-50T^2-10T)y(k-1) + (10T-50T^2-1)y(k-2) + 50T^2[x(k-1)+x(k-2)]$$

where $x(k)=1$ for all $k \geq 0$.

3.10-3 Simulation of Discrete Systems Using SIMULAB

To illustrate the use of SIMULAB to simulate discrete and/or computer controlled (hybrid) systems, consider the following examples.

Example 3.10-7

Determine the unit-step response of a discrete system represented by the transfer function

$$G(z) = \frac{C(z)}{R(z)} = \frac{0.625z-0.5}{z^2-1.375z+0.5}$$

This is the transfer function of the digital PDFF control system considered in Example 3.7-6, with $T=0.005$ sec. The SIMULAB block diagram is shown in Fig. 3.10-9. The unit-step response plot was identical to that shown in Fig. 3.7-11.

Example 3.10-8

Determine the unit-step response of the PDFF computer controlled system represented by the block diagram of Fig. 3.7-10a. Use the parameter values $K_p=5$, $K_i=1.25$, $K=20$, and $T=0.005$ sec.

The SIMULAB simulation diagram is shown in Fig. 3.10-10. The simulation results (numerical) matched those of Fig. 3.7-11. Computer screens from the SIMULAB output are shown in Fig. 3.10-11. As can be seen from this figure, one can monitor any of the signals in the block diagram, for various types inputs.

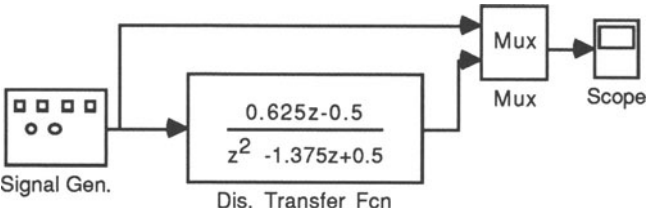


Fig. 3.10-9 SIMULAB block diagram for simulating the system in Example 3.10-7.

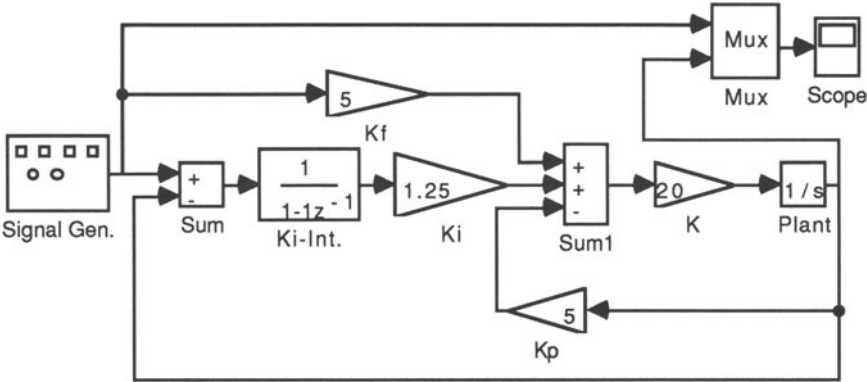


Fig. 3.10-10 SIMULAB block diagram for simulating the PDFF computer controlled system shown in Fig. 3.7-10.

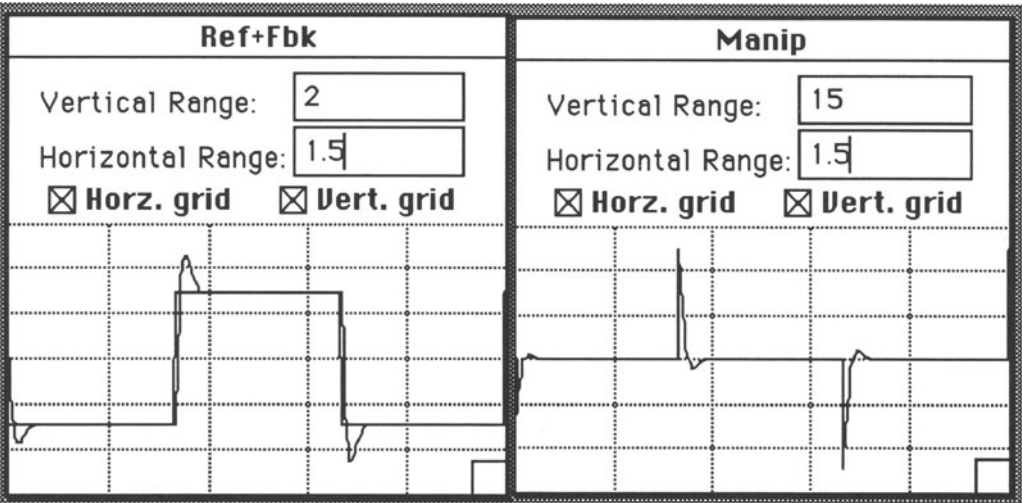


Fig. 3.10-11 SIMULAB output "screens" for PDFF system of Fig. 3.10-10. Plots are of reference and feedback (left), and manipulation (right).

PROBLEMS

- 3-1.** A discrete-time system is modeled by the difference equation

$$y(k+2) - 1.3205y(k+1) + 0.49659y(k) = 0.098327x(k+1) + 0.077779x(k) .$$

where $x(k)$ is the input and $y(k)$ the output. Determine and plot the unit-step response of the discrete system using the method of undetermined coefficients. Assume that $y(k)=0$ for $k \leq 0$.

- 3-2.** discrete system is represented by the difference equation

$$y(k) = y(k-2) + 2x(k)$$

where $x(k)$ represents the input and $y(k)$ the output. (a) Determine the transfer function of the system. (b) Determine and sketch the unit-impulse response of the system.

- 3--3.** A discrete-time system is modeled by the difference equation:

$$y(k+1) - y(k) = (1 + 2^k)x(k) ,$$

where $x(k)$ and $y(k)$ denote the system input and output, respectively. Determine the unit-step response of the system assuming zero initial conditions.

- 3-4.** A discrete-time system is represented by the transfer function

$$G(z) = \frac{Y(z)}{X(z)} = \frac{0.75z + 0.75}{z^2 - 0.7z + 1.2} .$$

(a) Determine the corresponding difference equation model. (b) Determine and sketch the unit-step response of the system.

- 3-5.** Using the transfer function of the system in Problem 3-1, determine and plot the unit-step response of the system using the following methods:

- (a) The $\mathcal{Z}$ transform method, that is, partial fractions, tables, etc.
- (b) The long division method - applied to the appropriate rational polynomial function in z .

- 3-6.** Assume that the spacing of the points (samples) for the difference equation in Problem 3-1 is $T=0.05\text{sec}$. Determine the differential equation that approximates the difference equation at the sampling instants $0, T, 2T, 3T, \dots$. Then, using the differential equation, determine the transfer function model for the corresponding continuous system.

- 3-7.** Determine the pulse transfer function of the continuous system determined in Problem 3-6. Assume that $T = 0.05 \text{ sec}$. Then, find the difference equation model of the system. Any surprises?

- 3-8. Determine and plot the unit-ramp response of the system of Problem 3-1 using the long division method.
- 3-9. Determine and plot frequency responses, that is, magnitude and phase versus frequency for the discrete system of Problem 3-1.
- 3-10. Derive the one-fourth (1/4) numerical integration rule given by Eq. (3.10-31).
- 3-11. Determine the recursion relation (difference equation) used to approximate the unit-step response of the system described by the differential equation:

$y''(t) + 14y'(t) + 100y(t) = 100x(t)$, assuming

- (a) rectangular (forward difference)
- (b) rectangular (backward differences), and
- (c) trapezoidal integration

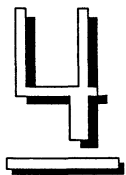
Assume zero initial conditions. Next, evaluate the respective unit-step responses using an appropriate integration time, but output results and plot them every 50 msec. Tabulate your results as follows:

Time	$y(k)-I_F(z)$	$y(k)-I_B(z)$	$y(k)-I_T(z)$
0.0			
:			
:			

$RMS(F) = ... \quad RMS(B) = ... \quad RMS(T) =$

The RMS error is given by $RMS = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} (y_a(i) - y_c(i))^2}$

where y_a represents exact and y_c calculated values/points.



STATE VARIABLE METHODS

for

Continuous & Discrete Systems

4.1 INTRODUCTION

The system analysis methods considered in previous chapters involved primarily linear, time-invariant, single-input-single-output (SISO) systems. To find the step response of a continuous system described by its differential equation, we used classical and Laplace transform methods. Similarly, to analyze a discrete system represented by its difference equation, we employed classical and $\mathcal{Z}$ transform techniques.

A continuous system described by an n th-order differential equation can also be represented by the corresponding s -domain transfer function. Similarly, an n th-order discrete system can be described either by a difference equation or by the corresponding z -domain transfer function. The differential (or difference) equation and transfer function representations are often referred to as conventional, and the methods of system analysis and design that use these representations are known as conventional, also.

The conventional representations are compact and popular for many well-established analysis and design techniques, but they also have limitations. For instance, the transfer function model is practical, because it enables us to work with algebraic equations, and is convenient for frequency domain analysis and design. However, the transfer function represents the relationship between one input variable and one output variable only; thus, it does not provide information concerning the internal variables of the system. In addition, the transfer function does not involve initial conditions and requires a root solver for analyzing higher-than-second-order systems. Finally, it may not be advantageous to apply conventional techniques to systems with multiple inputs and/or multiple outputs (MIMO), commonly known as **multivariable** systems. In general, Laplace transform techniques cannot be applied for time-varying or nonlinear systems.

An alternate method of system analysis and/or design is the **state space** or **state variable method**, which is synonymous with **modern system theory**. The state space method allows us to represent an n th-order, continuous system by a set of n

simultaneous, first-order differential equations. Similarly, an n th-order discrete system can be represented by a set of n simultaneous, first-order difference equations.

The state space representation offers several advantages over the conventional representation. First, it describes the relationships of the internal and external variables of the system enabling one to gain insight not only about its input-output characteristics, but also its total behavior. Second, this representation is more general and can be applied to systems with several inputs and/or outputs, as well as to certain types of nonlinear and time-varying systems. Finally, the state variable description of a system can be adapted to the compact matrix notation and matrix methods, which greatly facilitate the study of complex systems and lead to mathematical formulations that are suitable for computer solution.

The fundamental concepts of state space are presented in the following section. The state equations are formulated and solved using time domain and transform techniques, with emphasis the development of algorithms that are suitable for computerization.

4.2 STATE SPACE CONCEPTS: CONTINUOUS SYSTEMS

The state of a system is defined as the minimum amount of information required to determine its present and future response, for any present and future input.

For instance, the present and future behavior of a system represented by an n th-order differential (or difference) equation can be determined if in addition to its input, an appropriate set of n initial conditions is specified. This set of initial conditions represents the state of the system as it evolved from its past history.

From a mathematical standpoint, the state of a system at some initial time t_0 is simply the set of values of an appropriately chosen set of variables, called **state variables**. *The state variables of a system are defined as a minimal set of variables $x_1(t)$, $x_2(t)$, ..., $x_n(t)$ such that knowledge of these variables at any time $t=t_0$, plus knowledge of the input(s) for $t \geq t_0$ are sufficient to determine the state of the system for any time $t \geq t_0$.*

The number of state variables is equal to the order of a system, and if these n state variables are sufficient to completely describe the behavior of such a system, then these variables can be considered to be the components of a vector $\mathbf{x}(t)$, called the **state vector** and defined by

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \quad (4.2-1)$$

The state vector may be considered an element of the n -dimensional space, called **state space**, whose coordinate axes consist of the x_1 , x_2 , ..., and x_n axes. Thus, the state of the system at any time t can be represented by a point in the state space, defined by the state vector. Then, as time increases, the tip of the state vector defines a set of points, called the **vector trajectory** of the system.

The choice of state variables is not unique. As with block diagram representations, several alternative sets of state variables may be chosen to describe a system. Very often, state variables are chosen so that they correspond to physical quantities which are easy to measure.

In the case of mechanical systems, knowledge of relevant initial conditions, such as the initial positions and velocities of all masses, together with appropriate inputs are used to determine system behavior for present and future time.

To illustrate the concept of a set of state variables used to represent a mechanical system, consider the spring-mass translational system considered in Section 2.2-2 and shown in Fig. 2.2-3a. Two state variables sufficient to represent this system are the position $y(t)$ and velocity $y'(t)$. Therefore, the set of state variables can be defined as

$$x_1(t) = y(t) \quad (4.2-2)$$

and

$$x_2(t) = y'(t) \quad (4.2-3)$$

The second-order differential equation which describes the behavior of the system is given by Eq. (2.2-14) and is repeated here for convenience

$$My''(t) + By'(t) + Ky(t) = F_i(t) \quad (4.2-4)$$

where B represents the damping coefficient, M the mass, K the spring constant, and F_i the applied force or system input.

To express the above equation in terms of the state variables $x_1(t)$ and $x_2(t)$, substitute Eq. (4.2-2) and Eq. (4.2-3) into Eq. (4.2-4) to obtain

$$Mx_2'(t) + Bx_2(t) + Kx_1(t) = F_i(t) \quad (4.2-5)$$

Therefore, the state variable equations which describe the spring-mass system can be written as a set of first-order, simultaneous, differential equations

$$x_1'(t) = x_2(t) \quad (4.2-6)$$

and

$$x_2'(t) = -\frac{K}{M}x_1(t) - \frac{B}{M}x_2(t) + \frac{1}{M}F_i(t) \quad (4.2-7)$$

These equations can be written in vector-matrix form as

$$\begin{bmatrix} x_1'(t) \\ x_2'(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -K/M & -B/M \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/M \end{bmatrix} F_i(t) \quad (4.2-8)$$

In the case of electrical networks, knowledge of the initial voltages across capacitors and/or initial currents through inductors, together with the inputs, suffices to determine the behavior of a system. In other words, a selection of inductor currents and capacitor voltages gives a set of suitable state variables. Since inductors and capacitors are the energy storage elements of an electrical network, the stored energy of the system can also be determined in terms of these state variables.

To illustrate the state variable characterization of an electrical network, consider the RLC network example introduced in Section 2.2-1. The two energy storage elements are the inductor L and the capacitor C . Therefore, selecting the voltage across the capacitor and current through the inductor as the state variables, we can define the set of state variables as

$$x_1(t) = v_2(t) \quad (4.2-9)$$

and

$$x_2(t) = i(t) \quad (4.2-10)$$

Applying Kirchhoff's laws, the equations describing the dynamics of the system become

$$Cx_1'(t) = x_2(t) \quad (4.2-11)$$

and

$$Lx_2'(t) + Rx_2(t) + x_1(t) = v_1(t) \quad (4.2-12)$$

The output of the system can be the capacitor voltage, the inductor current, or a combination of both. The output can be formed from the state variables $x_1(t)$ and $x_2(t)$.

In writing the state equations, the first derivative of each state variable is set equal to a function of all state variables and inputs. The right-hand side should not contain any derivatives of either state variables or inputs. Thus, rearranging Eq. (4.2-11) and Eq. (4.2-12), the correct form for the state equations is

$$x_1'(t) = \frac{1}{C} x_2(t) \quad (4.2-13)$$

and

$$x_2'(t) = -\frac{1}{L} x_1(t) - \frac{R}{L} x_2(t) + \frac{1}{L} v_1(t) \quad (4.2-14)$$

Using vector-matrix notation, the equations are

$$\begin{bmatrix} x_1'(t) \\ x_2'(t) \end{bmatrix} = \begin{bmatrix} 0 & 1/C \\ -1/L & -R/L \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} v_1(t) \quad (4.2-15)$$

Eq. (4.2-15) is a state variable representation of the given electrical network. For any initial time t_0 , usually equal to zero, the dynamic behavior of the system is completely defined for $t \geq t_0$ if the initial current in the inductor, $x_2(t_0)$, the initial voltage across the capacitor, $x_1(t_0)$, and the input voltage, $v_1(t)$, are known. Thus, the state of the system for any $t \geq t_0$ is completely specified by the set of variables $x_1(t)$ and $x_2(t)$ and the input $v_1(t)$ for $t \geq t_0$.

For some systems, physical variables provide a natural choice for a set of state variables. In many cases, however, an alternate variable selection may be more appropriate for system analysis and design. In the case of block diagram representations, for instance, integrator outputs provide a convenient set of state variables. And even though the integrator outputs may or may not correspond directly to physical variables, their initial values, given the inputs, result in a unique system response.

To illustrate the procedure for obtaining a state variable description of a system from its block diagram, consider again the RLC network example but this time assume that the system is given in block diagram form as shown in Fig. 2.6-5b. A corresponding signal flow graph can be drawn as shown in Fig. 4.2-1.

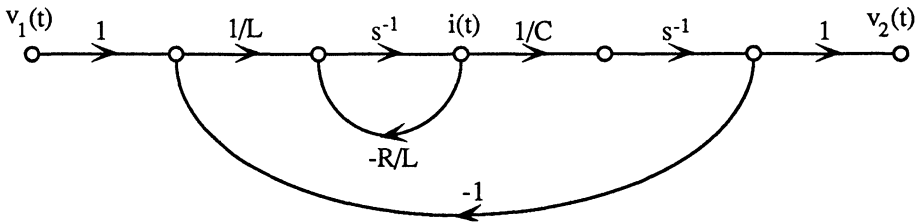


Fig. 4.2-1 Signal flow graph representation of an RLC network.

Since two integrators are present, this is a second-order system requiring two state variables $x_1(t)$ and $x_2(t)$. Before assigning state variables to integrator outputs, we must adopt an arbitrary convention for referring to integrators in a block diagram. For instance, let us number the integrators consecutively from output to input. For the system shown in Fig. 4.2-1, this means that the rightmost integrator is #1, and the one on the left is #2. The associated state variables are then assigned to the integrator outputs as shown in Fig. 4.2-2a.

The diagram in Fig. 4.2-2a also displays the derivatives of the state variables, that is, the integrator inputs. This diagram is called the **state diagram**, and it is used to derive the corresponding state variable and output equations for the system.

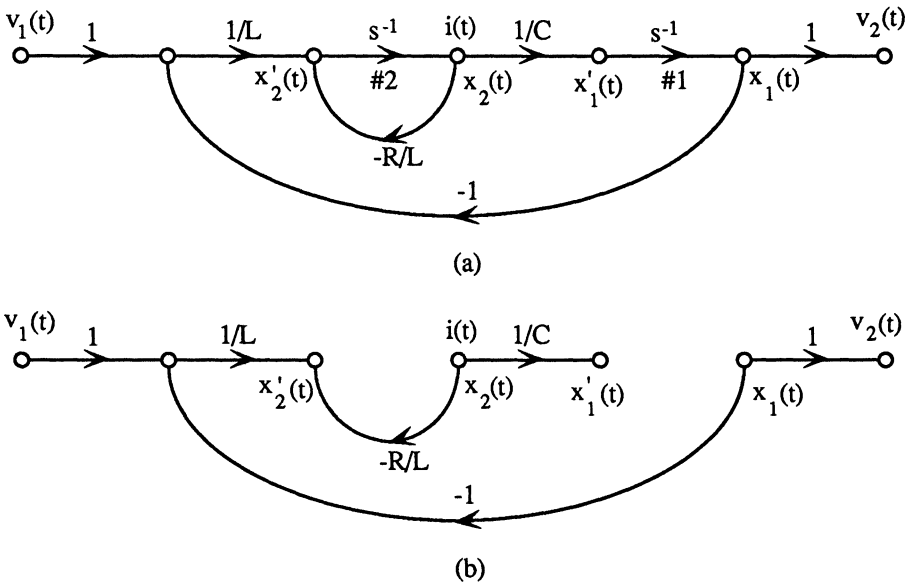


Fig. 4.2-2 State diagram for the RLC example system (a) with integrators included, and (b) with integrators removed.

To obtain the state and output equations from a state diagram, first remove the integrators, that is, the branches with gains s^{-1} , as shown in Fig. 4.2-2b. From the latter state diagram, viewing $x_1(t)$ and $x_2(t)$ as output nodes and $x_1'(t)$ and $x_2'(t)$ as input nodes, the state equations can be written as

$$x_1'(t) = \frac{1}{C} x_2(t) \quad (4.2-16)$$

$$x_2'(t) = -\frac{1}{L} x_1(t) - \frac{R}{L} x_2(t) + \frac{1}{L} v_1(t) \quad (4.2-17)$$

and the output equation as

$$v_2(t) = x_1(t) \quad (4.2-18)$$

Using vector-matrix notation, the state and output equations become

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1/C \\ -1/L & -R/L \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} v_1(t) \quad (4.2-19)$$

and
$$v_2(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad (4.2-20)$$

The above state space formulation is identical to that given by Eq. (4.2-15).

The set of equations formed from the state and output equations is generally referred to as the **dynamic equations**. The vector-matrix form of the respective state and output equations used to describe linear, time-invariant, multi-variable systems is commonly written as

$$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (4.2-21)$$

and
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \quad (4.2-22)$$

where, for an n th-order system with p inputs and q outputs,

$\mathbf{x}(t) = (n \times 1)$ **state variable vector**

$\mathbf{u}(t) = (p \times 1)$ **input vector**

$\mathbf{A} = (n \times n)$ **coefficient matrix or system matrix**

$\mathbf{B} = (n \times p)$ **driving or input or control matrix**

$\mathbf{y}(t) = (q \times 1)$ **output vector**

$\mathbf{C} = (q \times n)$ **output matrix**

$\mathbf{D} = (q \times p)$ **transmission matrix**

Care must be exercised when using the $\mathbf{u}(t)$ notation. For a single-input system, for instance, the input vector notation $\mathbf{u}(t)$ becomes identical to the popular unit-step function notation, which we use in this book. For state space considerations, the notation denotes system input(s) and not necessarily the unit-step function.

The state space model of an n th-order system with p inputs and q outputs can be represented in block diagram form as shown in Fig. 4.2-3. Observe that the input and output variables are shown as external to the system, while the state variables as internal. A block diagram representation of the vector-matrix form of the state and output equations is shown in Fig. 4.2-4.

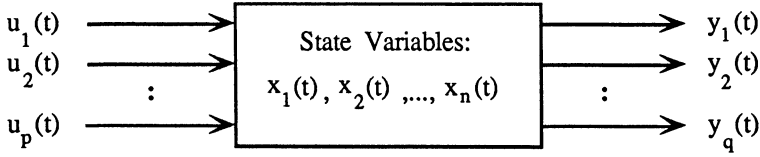


Fig. 4.2-3 State space representation of a continuous system.

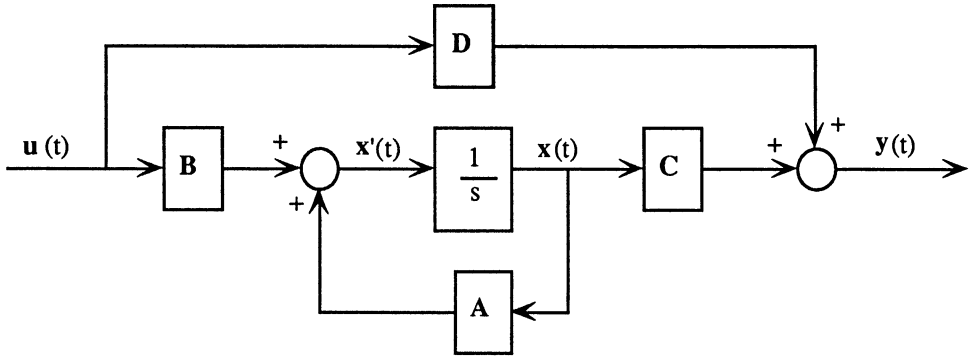


Fig. 4.2-4 Block diagram representation of continuous state and output equations.

4.2-1 State Transition Equation

Consider the n first-order differential equations given by Eq. (4.2-21). The equations in vector-matrix form are

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \quad (4.2-23)$$

where, for simplicity, $\mathbf{x}(t)$ has been replaced by $\mathbf{x}$ and $\mathbf{u}(t)$ by $\mathbf{u}$.

The objective is to solve Eq. (4.2-23) for the state variables $x_1(t), x_2(t), \dots, x_n(t)$, that is, to determine the state vector $\mathbf{x}$. Taking the Laplace transform on both sides of the equation, we obtain

$$s\mathbf{X}(s) - \mathbf{x}(0) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s) \quad (4.2-24)$$

rearranging and pre-multiplying by $(s\mathbf{I} - \mathbf{A})^{-1}$, we get

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}(0) + (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) \quad (4.2-25)$$

where $\mathbf{I}$ denotes the $(n \times n)$ identity matrix. Taking the inverse Laplace transform, results in the solution vector

$$\mathbf{x}(t) = \mathcal{L}^{-1}[(s\mathbf{I} - \mathbf{A})^{-1}]\mathbf{x}(0) + \mathcal{L}^{-1}[(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)] \quad (4.2-26)$$

Eq. (4.2-26) is called the **state transition equation** and represents the solution of the state equations. The term $\mathcal{L}^{-1}[(s\mathbf{I}-\mathbf{A})^{-1}]$, often denoted by $\phi(t)$, is called the **state transition** or **fundamental matrix**. Hence,

$$\mathcal{L}[\phi(t)] = \Phi(s) = (s\mathbf{I}-\mathbf{A})^{-1} \quad (4.2-27)$$

where

$$\phi(t) = \mathcal{L}^{-1}[(s\mathbf{I}-\mathbf{A})^{-1}] \quad (4.2-28)$$

is the state transition matrix. The state transition equation then is

$$\mathbf{x}(t) = \phi(t)\mathbf{x}(0) + \mathcal{L}^{-1}[(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)] \quad (4.2-29)$$

Using the convolution integral in place of Laplace domain multiplication,

$$\mathbf{x}(t) = \phi(t)\mathbf{x}(0) + \int_0^t \phi(t-\tau)\mathbf{B}\mathbf{u}(\tau)d\tau \quad (4.2-30)$$

where τ is a dummy variable.

For the homogeneous case, ($\mathbf{u}(t)=0$), the state transition matrix $\phi(t)$ represents the transient response of the system. As the name implies, *the state transition matrix defines the transition of the states from an initial time $t=0$ to any time t - with inputs equal to zero.* Notice that the second term on the right side of Eq. (4.2-29) represents the steady-state component of the response.

For an arbitrary initial time $t=t_0$, the state transition equation can be written in the general form

$$\mathbf{x}(t) = \phi(t-t_0)\mathbf{x}(t_0) + \int_{t_0}^t \phi(t-\tau)\mathbf{B}\mathbf{u}(\tau)d\tau \quad (4.2-31)$$

To determine the state transition matrix, it is necessary to evaluate the inverse of the matrix $(s\mathbf{I}-\mathbf{A})$. From linear algebra, the inverse of this matrix can be calculated from

$$(s\mathbf{I}-\mathbf{A})^{-1} = \frac{1}{|s\mathbf{I}-\mathbf{A}|} \text{Adj}(s\mathbf{I}-\mathbf{A}) \quad (4.2-32)$$

where "Adj" denotes the adjoint matrix, obtained by transposing the cofactor matrix.

The characteristic equation of a system represented by the matrix $\mathbf{A}$ can be determined by setting the determinant $|s\mathbf{I}-\mathbf{A}|$ equal to zero. That is, the characteristic equation is given by

$$|s\mathbf{I}-\mathbf{A}| = s^n + b_1s^{n-1} + b_2s^{n-2} + \dots + b_{n-1}s + b_n = 0 \quad (4.2-33)$$

The zeros of Eq. (4.2-33) are the characteristic roots or eigenvalues of $\mathbf{A}$ and, thus, of the system that $\mathbf{A}$ represents. The above concepts are illustrated next by an example.

Example 4.2-1

Assume that the parameter values for the RLC network described by Eq. (4.2-19) are $R=1$ Ohm, $L=0.1$ Henry, and $C=0.1$ Farad. Develop the analytical expressions for the state variables capacitor voltage (system output) and inductor current when the input is a step of 1 volt. Specifically, plot the two state variables $x_1(t)$ and $x_2(t)$ versus time. Also, plot $x_2(t)$ versus $x_1(t)$ in order to obtain the vector trajectory of the system.

Substituting the numerical values of R , L and C in Eq. (4.2-19), we obtain the state equation

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 10 \\ -10 & -10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 10 \end{bmatrix} v_1(t)$$

The system and input matrices are

$$\mathbf{A} = \begin{bmatrix} 0 & 10 \\ -10 & -10 \end{bmatrix}, \text{ and } \mathbf{B} = \begin{bmatrix} 0 \\ 10 \end{bmatrix}$$

First, let us determine the state transition matrix. The matrix $(s\mathbf{I}-\mathbf{A})$ is

$$(s\mathbf{I}-\mathbf{A}) = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} - \begin{bmatrix} 0 & 10 \\ -10 & -10 \end{bmatrix} = \begin{bmatrix} s & -10 \\ 10 & s+10 \end{bmatrix}$$

and the characteristic equation is

$$|s\mathbf{I}-\mathbf{A}| = s(s+10) + 100 = s^2 + 10s + 100 = 0$$

The inverse of the matrix $(s\mathbf{I}-\mathbf{A})$ can be calculated from the relationship

$$(s\mathbf{I}-\mathbf{A})^{-1} = \frac{1}{|s\mathbf{I}-\mathbf{A}|} \text{Adj}(s\mathbf{I}-\mathbf{A})$$

The cofactor matrix obtained from $(s\mathbf{I}-\mathbf{A})$ is

$$\begin{bmatrix} s+10 & -10 \\ 10 & s \end{bmatrix}$$

which, by transposing, gives the adjoint matrix

$$\text{Adj}(s\mathbf{I}-\mathbf{A}) = \begin{bmatrix} s+10 & -10 \\ 10 & s \end{bmatrix}^T = \begin{bmatrix} s+10 & 10 \\ -10 & s \end{bmatrix}$$

The transformed state transition matrix becomes

$$\Phi(s) = (s\mathbf{I}-\mathbf{A})^{-1} = \begin{bmatrix} \Phi_{11}(s) & \Phi_{12}(s) \\ \Phi_{21}(s) & \Phi_{22}(s) \end{bmatrix} = \frac{1}{s^2+10s+100} \begin{bmatrix} s+10 & 10 \\ -10 & s \end{bmatrix}$$

and the state transition matrix can then be calculated from the equation

$$\phi(t) = \mathcal{L}^{-1}[(s\mathbf{I}-\mathbf{A})^{-1}] = \begin{bmatrix} \phi_{11}(t) & \phi_{12}(t) \\ \phi_{21}(t) & \phi_{22}(t) \end{bmatrix}$$

The first term in the transformed matrix is

$$\Phi_{11}(s) = \frac{s+10}{s^2+10s+100} = \frac{s+5}{(s+5)^2+8.66^2} + 0.5774 \frac{8.66}{(s+5)^2+8.66^2}$$

Taking the inverse Laplace transform results in the expression

$$\phi_{11}(t) = e^{-5t}(\cos 8.66t + 0.5774 \sin 8.66t)$$

Similarly, the other terms of the state transition matrix are found as

$$\phi_{12}(t) = 1.1547e^{-5t}\sin 8.66t$$

$$\phi_{21}(t) = -1.1547e^{-5t}\sin 8.66t$$

and

$$\phi_{22}(t) = e^{-5t}(\cos 8.66t - 0.5774 \sin 8.66t)$$

If the external inputs are zero, the transient response of the system can be written in terms of the initial conditions and the above expressions. That is,

$$x_1(t) = \phi_{11}(t)x_1(0) + \phi_{12}(t)x_2(0)$$

and

$$x_2(t) = \phi_{21}(t)x_1(0) + \phi_{22}(t)x_2(0)$$

To determine the complete solution, however, we must also determine the steady-state component of the response. For a unit-step input, Eq. (4.2-26) gives

$$\begin{aligned} (s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) &= \frac{1}{s^2+10s+100} \begin{bmatrix} s+10 & 10 \\ -10 & s \end{bmatrix} \begin{bmatrix} 0 \\ 10 \end{bmatrix} \frac{1}{s} \\ &= \frac{1}{s^2+10s+100} \begin{bmatrix} \frac{100}{s} \\ 10 \end{bmatrix} = \begin{bmatrix} \frac{1}{s} - \frac{s+10}{s^2+10s+100} \\ \frac{10}{s^2+10s+100} \end{bmatrix} \end{aligned}$$

The inverse Laplace transform of the above matrix equation results in the steady-state solution components for $x_1(t)$ and $x_2(t)$. That is,

$$\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} 1 - e^{-5t}(\cos 8.66t + 0.5774 \sin 8.66t) \\ 1.1547e^{-5t}\sin 8.66t \end{bmatrix}, \text{ for } t \geq 0$$

The steady-state components are combined with the transient components and the system response is completely determined.

Plots of the capacitor voltage $x_1(t)$, and inductor current $x_2(t)$ versus time are shown in Fig. 4.2-5a. The state vector trajectory of the system, obtained by plotting $x_1(t)$ versus $x_2(t)$, is shown in Fig. 4.2-5b.

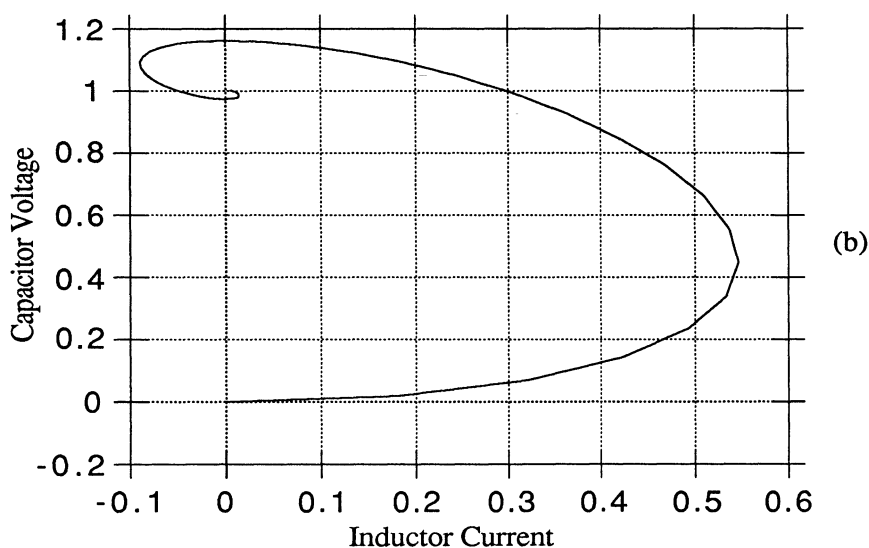
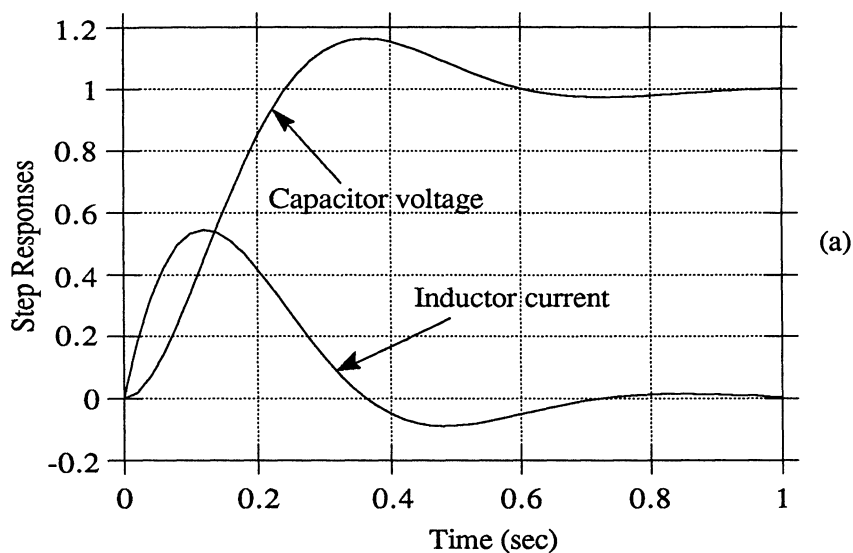


Fig. 4.2-5 (a) Capacitor voltage $x_1(t)$ and inductor current $x_2(t)$ vs. time; (b) state vector trajectory $x_1(t)$ vs. $x_2(t)$ for the RLC system of Example 4.2-1.

Leverrier's Algorithm. The analytical approach illustrated in the previous example for evaluating the inverse of the matrix $(s\mathbf{I}-\mathbf{A})$ becomes impractical for higher-order systems. A more systematic procedure which is also suitable for computerization is based on Leverrier's algorithm. To illustrate this method, consider the system determinant

$$|s\mathbf{I}-\mathbf{A}| = s^n + b_1s^{n-1} + b_2s^{n-2} + \dots + b_{n-1}s + b_n \quad (4.2-34)$$

and let us express the adjoint matrix of $(s\mathbf{I}-\mathbf{A})$ in the form

$$\text{Adj}(s\mathbf{I}-\mathbf{A}) = \mathbf{H}_1s^{n-1} + \mathbf{H}_2s^{n-2} + \mathbf{H}_3s^{n-3} + \dots + \mathbf{H}_n \quad (4.2-35)$$

where n is the order of the system. Then, according to Leverrier's algorithm, the b coefficients in Eq. (4.2-34) and the $(n \times n)$ $\mathbf{H}$ matrices in Eq. (4.2-35) can be evaluated from the relationships

$$\mathbf{H}_1 = \mathbf{I}, \quad b_1 = -\text{Trace}(\mathbf{A}) \quad (4.2-36)$$

$$\mathbf{H}_2 = \mathbf{A}\mathbf{H}_1 + b_1\mathbf{I}, \quad b_2 = -\frac{1}{2}\text{Trace}(\mathbf{A}\mathbf{H}_2) \quad (4.2-37)$$

and for $j = 3, 4, \dots, n$

$$\mathbf{H}_j = \mathbf{A}\mathbf{H}_{j-1} + b_{j-1}\mathbf{I}, \quad b_j = -\frac{1}{j}\text{Trace}(\mathbf{A}\mathbf{H}_j) \quad (4.2-38)$$

The **trace** of a matrix is equal to the sum of its main diagonal elements. The method outlined above can be programmed on a digital computer in order to evaluate the characteristic equation from the system matrix $\mathbf{A}$. When the method is coupled with a root solver, it results in a powerful tool for determining characteristic roots even for a high-order system.

Example 4.2-2

Apply Leverrier's algorithm to determine the inverse of $(s\mathbf{I}-\mathbf{A})$ for the system used in Example 4.2-1.

The system matrix is
$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -10 & -1 \end{bmatrix}$$

Therefore, for $n=2$, the relationships to be evaluated are

$$\mathbf{H}_1 = \mathbf{I}, \quad b_1 = -\text{Trace}(\mathbf{A})$$

$$\text{and} \quad \mathbf{H}_2 = \mathbf{A}\mathbf{H}_1 + b_1\mathbf{I}, \quad b_2 = -\frac{1}{2}\text{Trace}(\mathbf{A}\mathbf{H}_2)$$

Substituting the $\mathbf{A}$ matrix and b_1, b_2 values we get

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad b_1 = -(-10) = 10$$

$$\mathbf{H}_2 = \begin{bmatrix} 0 & 10 \\ -10 & -10 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + 10 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 10 & 10 \\ -10 & 0 \end{bmatrix}$$

and $\mathbf{A}\mathbf{H}_2 = \begin{bmatrix} 0 & 10 \\ -10 & -10 \end{bmatrix} \begin{bmatrix} 10 & 10 \\ -10 & 0 \end{bmatrix} = \begin{bmatrix} -100 & 0 \\ 0 & -100 \end{bmatrix}$

From Eq. (4.2-34), with $b_2 = -(-200)/2 = 100$, we obtain the characteristic equation

$$|s\mathbf{I} - \mathbf{A}| = s^2 + b_1s + b_2 = s^2 + 10s + 100$$

and, from Eq. (4.2-35), the adjoint matrix

$$\text{Adj}(s\mathbf{I} - \mathbf{A}) = \mathbf{H}_1s + \mathbf{H}_2 = \begin{bmatrix} (s+10) & 10 \\ -10 & s \end{bmatrix}$$

Thus, the desired result is

$$(s\mathbf{I} - \mathbf{A})^{-1} = \frac{1}{s^2 + 10s + 100} \begin{bmatrix} (s+10) & 10 \\ -10 & s \end{bmatrix}$$

which, as expected, agrees with the result obtained in Example 4.2-1.

State Transition and Exponential Matrices. The state transition matrix is defined as a matrix that satisfies the homogeneous state equation

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} \quad (4.2-39)$$

To find an alternative way for evaluating the state transition matrix, consider the following approach. We can verify by differentiation that the solution to the above equation is

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}(0) \quad (4.2-40)$$

where $e^{\mathbf{A}t}$ is the exponential matrix defined by

$$e^{\mathbf{A}t} = \mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2t^2}{2!} + \frac{\mathbf{A}^3t^3}{3!} + \dots = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k t^k}{k!} \quad (4.2-41)$$

Since from Eq. (4.2-30) and for zero input

$$\mathbf{x}(t) = \Phi(t)\mathbf{x}(0) \quad (4.2-42)$$

it is clear that $e^{\mathbf{A}t}$ is also the state transition matrix. That is,

$$\Phi(t) = e^{\mathbf{A}t} \quad (4.2-43)$$

Eq. (4.2-43) is valid for all t . For an arbitrary initial time t_0 ,

$$\Phi(t-t_0) = e^{\mathbf{A}(t-t_0)} \quad (4.2-44)$$

It follows from Eq. (4.2-43) that

$$\Phi(0) = \mathbf{I} \quad (4.2-45)$$

where $\mathbf{I}$ denotes the identity matrix.

Several other properties of the exponential matrix can be easily derived. For instance, multiplying each side of Eq. (4.2-43) on the right by $e^{-\mathbf{A}t}$ gives

$$\Phi(t)e^{-\mathbf{A}t} = e^{\mathbf{A}t}e^{-\mathbf{A}t} = \mathbf{I} \quad (4.2-46)$$

Then, multiplying both sides on the left by $\Phi^{-1}(t)$, we obtain

$$e^{-\mathbf{A}t} = \Phi^{-1}(t) \quad (4.2-47)$$

which implies the property

$$\Phi(-t) = \Phi^{-1}(t) = e^{-\mathbf{A}t} \quad (4.2-48)$$

Another property is

$$\Phi(t_2 - t_1)\Phi(t_1 - t_0) = \Phi(t_2 - t_0) \quad (4.2-49)$$

for any t_0 , t_1 and t_2 . This last property can be proven by substitution.

Caley-Hamilton Theorem. The main operation performed when evaluating the exponential matrix is raising the matrix $\mathbf{A}$ to a power, as shown by Eq. (4.2-41). This operation may be simplified by the use of the *Caley-Hamilton theorem which states that every square matrix of order n must satisfy its own characteristic equation*. That is, if the system characteristic equation is given by

$$\mathbf{P}(s) = s^n + b_1s^{n-1} + b_2s^{n-2} + \dots + b_{n-1}s + b_n = 0 \quad (4.2-50)$$

then

$$\mathbf{P}(\mathbf{A}) = \mathbf{A}^n + b_1\mathbf{A}^{n-1} + b_2\mathbf{A}^{n-2} + \dots + b_{n-1}\mathbf{A} + b_n\mathbf{I} = 0 \quad (4.2-51)$$

Eq. (4.2-51) allows the evaluation of $\mathbf{A}^n$ as a function of $\mathbf{A}^{n-1}$, $\mathbf{A}^{n-2}$, ..., $\mathbf{I}$.

Example 4.2-3

To illustrate the use of the Caley-Hamilton theorem, consider the system used in Example 4.2-1 where the $\mathbf{A}$ matrix is given by

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -10 & -10 \end{bmatrix}$$

and the characteristic equation is

$$\mathbf{P}(s) = |s\mathbf{I} - \mathbf{A}| = s^2 + 10s + 100 = 0$$

Applying the Caley-Hamilton theorem, we obtain

$$\mathbf{P}(\mathbf{A}) = \mathbf{A}^2 + 10\mathbf{A} + 100\mathbf{I} = 0$$

from which A^2 is calculated as

$$A^2 = -10A - 100I$$

We can calculate higher powers of A by applying the theorem again. For instance,

$$A^3 = A \cdot A^2 = -10A(A+10I) = -10(A^2+10A) = -10(-10A-100I+10A) = 1000I,$$

and so forth.

4.2-2 State Equations and Transfer Functions

Consider the state and output equations

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u \quad (4.2-52)$$

$$y = \mathbf{C}\mathbf{x} + \mathbf{D}u \quad (4.2-53)$$

Laplace transforming Eq. (4.2-52) and solving for $\mathbf{X}(s)$ yields

$$\mathbf{X}(s) = (s\mathbf{I}-\mathbf{A})^{-1}\mathbf{x}(0) + (s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}U(s) \quad (4.2-54)$$

Laplace transforming the output equation and substituting $\mathbf{X}(s)$ from Eq. (4.2-54), we obtain the output transform

$$Y(s) = \mathbf{C}(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{x}(0) + \mathbf{C}(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}U(s) + \mathbf{D}U(s) \quad (4.2-55)$$

Recall that the calculation of the transfer function requires zero initial conditions, that is, $\mathbf{x}(0) = 0$. Then, the last equation can be written as

$$Y(s) = [\mathbf{C}(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B} + \mathbf{D}]U(s) \quad (4.2-56)$$

Using matrix notation, the transfer function matrix becomes

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{I}-\mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \quad (4.2-57)$$

For a multivariable system with p inputs and q outputs, $\mathbf{G}(s)$ is a $(q \times p)$ matrix of transfer functions. Note that the transmission matrix $\mathbf{D}$ is zero for any system whose transfer function has more poles than zeros.

Example 4.2-4

Determine the transfer function for the system of Example 4.2-1 by using the matrix relationship expressed by Eq. (4.2-57). Find the poles of the system, also.

From Example 4.2-1, the relevant matrices are

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -10 & -10 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 10 \end{bmatrix}, \quad \mathbf{C} = [1 \quad 0], \quad \text{and} \quad \mathbf{D} = [0]$$

From the example, we also know that

$$(s\mathbf{I}-\mathbf{A})^{-1} = \frac{1}{s^2+10s+100} \begin{bmatrix} (s+10) & 10 \\ -10 & s \end{bmatrix}$$

Thus, from Eq. (4.2-57), the transfer function is

$$\begin{aligned} G(s) &= \frac{1}{s^2+10s+100} [1 \quad 0] \begin{bmatrix} (s+10) & 10 \\ -10 & s \end{bmatrix} \begin{bmatrix} 0 \\ 10 \end{bmatrix} \\ &= \frac{1}{s^2+10s+100} [(s+10) \quad 10] \begin{bmatrix} 0 \\ 10 \end{bmatrix} = \frac{100}{s^2+10s+100} \end{aligned}$$

The poles of the system are the zeros of the characteristic equation

$$s^2 + 10s + 100 = 0$$

which are

$$s_{1,2} = -5 \pm -j8.66$$

4.2-3 From Transfer Functions to State Equations

The objective in this section is to develop systematic procedures for obtaining a state space model of a system from its corresponding transfer function representation. The process of going from the transfer function to a state diagram and subsequently to a state space model is called **decomposition** or **programming** of the transfer function.

There are three basic methods used to decompose/program a transfer function. These methods are referred to as **direct**, **cascade** (or **series**) and **parallel**. They are developed and illustrated next by examples.

Direct Decomposition. It is called direct because it does not require the factoring of the numerator or denominator polynomials of the transfer function. Without loss of generality, this method can be illustrated by considering the second-order transfer function

$$\frac{Y(s)}{U(s)} = \frac{a_0 s^2 + a_1 s + a_2}{s^2 + b_1 s + b_2} \quad (4.2-58)$$

Higher-order systems can be handled in a similar fashion.

As a first step, we divide the numerator and denominator by the highest power of s in the denominator, in this case s^2 . This gives

$$\frac{Y(s)}{U(s)} = \frac{a_0 + a_1 s^{-1} + a_2 s^{-2}}{1 + b_1 s^{-1} + b_2 s^{-2}} \quad (4.2-59)$$

Next, we multiply the numerator and denominator by a dummy variable $E(s)$ to obtain

$$\frac{Y(s)}{U(s)} = \frac{(a_0 + a_1 s^{-1} + a_2 s^{-2})E(s)}{(1 + b_1 s^{-1} + b_2 s^{-2})E(s)} \quad (4.2-60)$$

Equating numerators and denominators in Eq. (4.2-60), we get

$$Y(s) = (a_0 + a_1s^{-1} + a_2s^{-2})E(s) \quad (4.2-61)$$

and

$$U(s) = (1 + b_1s^{-1} + b_2s^{-2})E(s) \quad (4.2-62)$$

To eliminate the dummy variable $E(s)$, we solve the second equation for $E(s)$. Thus,

$$E(s) = U(s) - b_1s^{-1}E(s) - b_2s^{-2}E(s) \quad (4.2-63)$$

Using Eq. (4.2-61) and Eq. (4.2-63), results in the state diagram shown in Fig. 4.2-6. The state variables are defined as the outputs of the integrators, numbered from output to input according to the convention established earlier. Finally, with the integrators removed, the state and output equations are written as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -b_2 & -b_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad (4.2-64)$$

and

$$y = [(a_2 - a_0b_2) \quad (a_1 - a_0b_1)] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [a_0]u \quad (4.2-65)$$

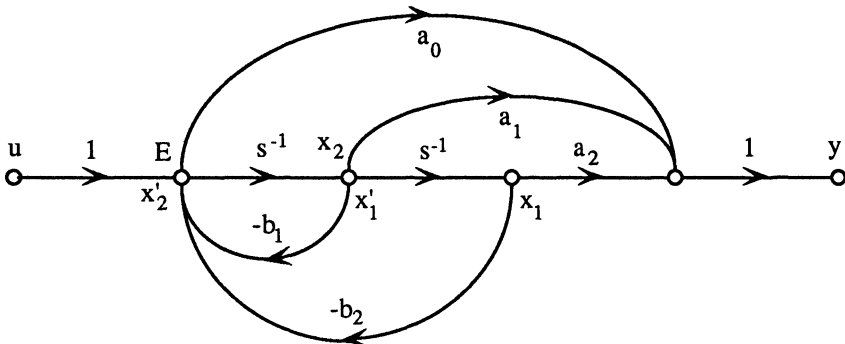


Fig. 4.2-6 State diagram drawn from Eq. (4.2-59) by direct decomposition.

Cascade Decomposition. This method of decomposition requires that the transfer function be in factored form. For example, consider the system

$$\frac{Y(s)}{U(s)} = \frac{K(s+a)(s+b)}{(s+c)(s+d)}, \quad (4.2-66)$$

where K , a , b , c , and d are real numbers. Viewing the transfer function as a product of two first-order transfer functions, the state diagram can be drawn as shown in Fig. 4.2-7. This can be verified by applying direct decomposition to the individual factors in the transfer function. From the state diagram, the respective state and output equations are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -d & (a-c) \\ 0 & -c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} K \\ K \end{bmatrix} u \quad (4.2-67)$$

and

$$y = [(b-d) \quad (a-c)] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [K]u \quad (4.2-68)$$

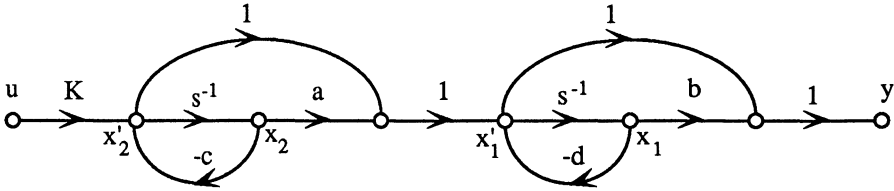


Fig. 4.2-7 State diagram drawn from Eq. (4.2-66) by cascade decomposition.

Parallel Decomposition. The parallel decomposition method requires that the transfer function be expanded into partial fractions. To illustrate the method, consider a transfer function expanded into partial fractions as

$$\frac{Y(s)}{U(s)} = \frac{K_1}{s+a} + \frac{K_2}{s+b} \quad (4.2-69)$$

The state diagram for this system is shown in Fig. 4.2-8. The state diagram is drawn by combining the state diagrams corresponding to the two terms in the partial fraction expansion. The state and output equations are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -a & 0 \\ 0 & -b \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \quad (4.2-70)$$

and

$$y = [K_1 \quad K_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (4.2-71)$$

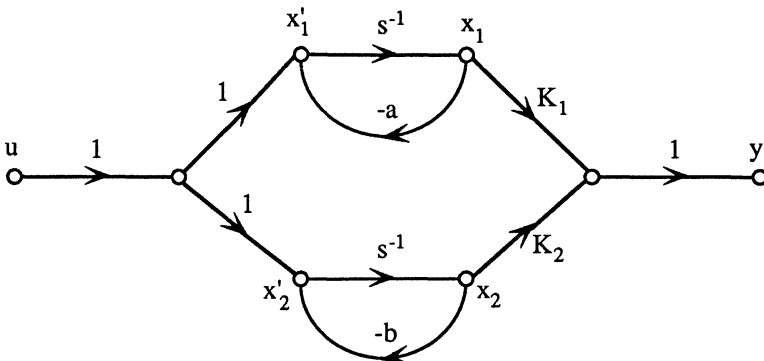


Fig. 4.2-8 State diagram drawn from Eq. (4.2-69) by parallel decomposition.

If the system transfer function has only simple poles, the A matrix that results by parallel decomposition is always a diagonal matrix. One of the advantages of a diagonal A is that the eigenvalues of the system are the main diagonal elements of A . Care must be exercised when applying parallel decomposition to systems whose transfer function has multiple poles. Some of the integrators in the state diagram may have to be "shared" so that the number of integrators used does not exceed the order of the system.

Example 4.2-5

A system is described by the transfer function

$$G(s) = \frac{Y(s)}{U(s)} = \frac{2}{s^2 + 3s + 2}$$

Determine state-space models when (a) direct, (b) cascade and (c) parallel decomposition schemes are applied.

(a) Direct Decomposition

The transfer function can be written as

$$\frac{Y(s)}{U(s)} = \frac{2}{s^2 + 3s + 2} = \frac{2s^{-2}E(s)}{(1 + 3s^{-1} + 2s^{-2})E(s)}$$

where $E(s)$ is a dummy variable. The relevant equations are then obtained by equating the numerators and denominators. That is,

$$Y(s) = 2s^{-2}E(s)$$

and

$$E(s) = U(s) - 3s^{-1}E(s) - 2s^{-2}E(s)$$

From these equations, the state diagram is drawn as shown in Fig. 4.2-9.

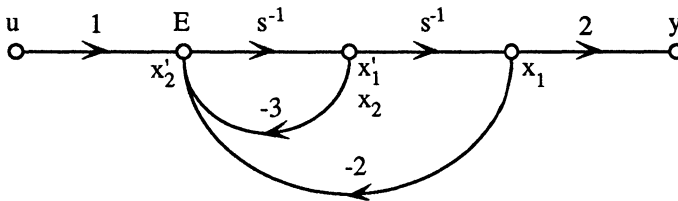


Fig. 4.2-9 State diagram for direct decomposition of transfer function used in Example 4.2-5.

The corresponding state and output equations are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

and

$$y = [2 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

(b) Cascade or Series Decomposition

For this case, the transfer function can be expressed in the factored form

$$\frac{Y(s)}{U(s)} = \left(\frac{1}{s+1} \right) \left(\frac{2}{s+2} \right)$$

The state diagram, obtained by cascading the state diagrams of the two first-order transfer functions, is shown in Fig. 4.2-10. The corresponding state and output equations are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -2 & 2 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

and

$$y = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

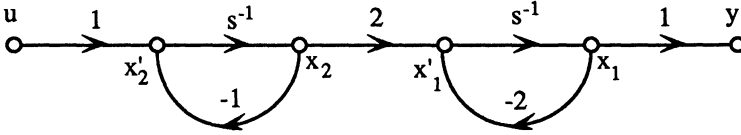


Fig. 4.2-10 State diagram for cascade decomposition of transfer function used in Example 4.2-5.

(c) Parallel Decomposition

Using partial fraction expansion, we can write the transfer function in the form

$$G(s) = \frac{Y(s)}{U(s)} = \frac{2}{(s+1)(s+2)} = \frac{2}{s+1} - \frac{2}{s+2}$$

The state diagram is obtained by combining the two first-order state diagrams in parallel, as shown in Fig. 4.2-11. The state and output equations are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix} u$$

and

$$y = [1 \quad -1] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Note that the poles are real and simple and that matrix **A** is diagonal. Furthermore, the diagonal elements are the system eigenvalues $s_1 = -1$ and $s_2 = -2$.

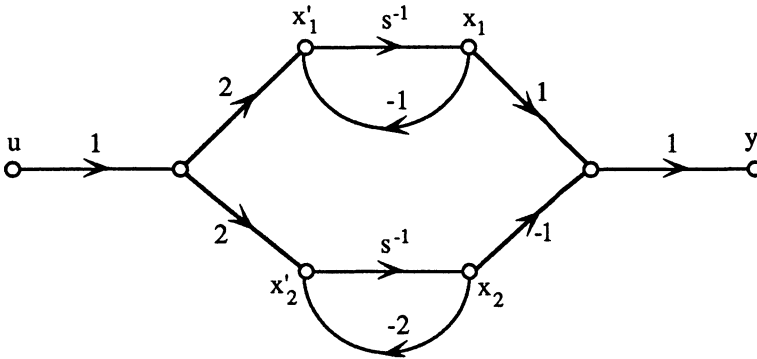


Fig. 4.2-11 State diagram for parallel decomposition of transfer function used in Example 4.2-5.

4.2-4 Phase-variable Canonical Form

When an n th-order transfer function of the form

$$G(s) = \frac{a_0 s^n + a_1 s^{n-1} + a_2 s^{n-2} + \dots + a_n}{s^n + b_1 s^{n-1} + b_2 s^{n-2} + \dots + b_n} \quad (4.2-72)$$

is decomposed by the direct programming method, the resulting system matrix is given by

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -b_n & -b_{n-1} & -b_{n-2} & \dots & -b_1 \end{bmatrix} \quad (4.2-73)$$

and the input matrix by

$$B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 1 \end{bmatrix} \quad (4.2-74)$$

If matrices **A** and **B** are as given by Eq. (4.2-73) and Eq. (4.2-74), respectively, then the state equations are said to be in **phase-variable canonical form**, which is particularly useful in the design of control systems. If a system described by the state equation

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u \quad (4.2-75)$$

is not in phase-variable canonical form, there exists a transformation relationship that transforms Eq. (4.2-75) into phase-variable canonical form if the **controllability test matrix** given by

$$\Gamma = [\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B} \quad \dots \quad \mathbf{A}^{n-1}\mathbf{B}] \quad (4.2-76)$$

is nonsingular. Consider a system represented by the state equation

$$\mathbf{x}' = \mathbf{Ax} + \mathbf{Bu} \quad (4.2-77)$$

where the matrices $\mathbf{A}$ and $\mathbf{B}$ are in arbitrary form. It is possible to transform the above system of equations into controllable, phase-variable canonical form. The structure of the system matrix, call it $\mathbf{A}_o$, in controllable canonical form, and the input (control) matrix, call it $\mathbf{B}_o$, are of the form

$$\mathbf{A}_o = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -b_n & -b_{n-1} & -b_{n-2} & \dots & -b_1 \end{bmatrix} \quad (4.2-78)$$

and

$$\mathbf{B}_o = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \quad (4.2-79)$$

To find the desired transformation matrix $\mathbf{H}$, consider a transformation such that

$$\mathbf{e} = \mathbf{Hx} \quad (4.2-80)$$

Then,

$$\mathbf{e}' = \mathbf{Hx}' = \mathbf{H}(\mathbf{Ax} + \mathbf{Bu}), \text{ or } \mathbf{e}' = \mathbf{HAH}^{-1}\mathbf{e} + \mathbf{HBu} \quad (4.2-81)$$

It can be seen that, for a system matrix $\mathbf{A}$, the transformation is

$$\mathbf{A}_o = \mathbf{HAH}^{-1} \quad (4.2-82)$$

and for the input matrix $\mathbf{B}$, the transformation is

$$\mathbf{B}_o = \mathbf{HB} \quad (4.2-83)$$

The structure of $\mathbf{B}_o$ permits the evaluation of the transformation matrix $\mathbf{H}$. The steps involved in finding $\mathbf{H}$ are indicated by the following algorithm:

Algorithm for finding $\mathbf{H}$:

- (1) Compute the controllability matrix Γ

$$\Gamma = [\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B} \quad \dots \quad \mathbf{A}^{n-1}\mathbf{B}] \quad (4.2-84)$$

- (2) Find Γ^{-1} , the inverse of Γ
- (3) Find $\mathbf{H}_1$, the last row of Γ^{-1}
- (4) Finally, form the desired transformation matrix $\mathbf{H}$ from the relationship

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_1 \mathbf{A} \\ \vdots \\ \mathbf{H}_1 \mathbf{A}^{n-1} \end{bmatrix} \quad (4.2-85)$$

$$\text{where} \quad \mathbf{H}_1 = [0 \ 0 \ \dots \ 1] [\mathbf{B} \ \mathbf{AB} \ \mathbf{A}^2\mathbf{B} \ \dots \ \mathbf{A}^{n-1}\mathbf{B}]^{-1} \quad (4.2-86)$$

Representing a system in phase-variable canonical form has several advantages. One such advantage is that it can be easily transformed into diagonal form. If a system has distinct eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$, and it is in phase-variable canonical form, the system matrix $\mathbf{A}_0$ can be diagonalized by the transformation

$$\mathbf{P}^{-1}\mathbf{A}_0\mathbf{P} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix} \quad (4.2-87)$$

where the matrix $\mathbf{P}$, known as **modal matrix**, is given by

$$\mathbf{P} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ \lambda_1 & \lambda_2 & \lambda_3 & \dots & \lambda_n \\ \lambda_1^2 & \lambda_2^2 & \lambda_3^2 & \dots & \lambda_n^2 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \lambda_1^{n-1} & \lambda_2^{n-1} & \lambda_3^{n-1} & \dots & \lambda_n^{n-1} \end{bmatrix} \quad (4.2-88)$$

Diagonalization, when possible, has practical implications and offers advantages in system design as illustrated by the following example.

Example 4.2-6

Consider a system described by the state equations

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -2 & 2 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

First, determine the phase-variable canonical form. Then, if possible, diagonalize the new system matrix $\mathbf{A}$.

From Eq. (4.2-84), we obtain the Γ matrix as

$$\Gamma = [\mathbf{B} \ \mathbf{AB}] = \begin{bmatrix} 0 & 2 \\ 1 & -1 \end{bmatrix}$$

which is nonsingular because its determinant is not zero. This implies that the system can be transformed into phase-variable canonical form. The inverse matrix is

$$\Gamma^{-1} = [\mathbf{B} \quad \mathbf{AB}]^{-1} = \begin{bmatrix} 0.5 & 1 \\ 0.5 & 0 \end{bmatrix}$$

From the last row of the above matrix, we obtain

$$\mathbf{H}_1 = [0.5 \quad 0]$$

From Eq. (4.2-85), the transformation matrix is

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_1 \mathbf{A} \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ -1 & 1 \end{bmatrix}$$

Thus,

$$\mathbf{H}^{-1} = \begin{bmatrix} 2 & 0 \\ 2 & 1 \end{bmatrix}$$

The new system matrix is

$$\mathbf{A}_o = \mathbf{H} \mathbf{A} \mathbf{H}^{-1} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

and the new input matrix is

$$\mathbf{B}_o = \mathbf{H} \mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Using the new matrices, we obtain the modal matrix as

$$\mathbf{P} = \begin{bmatrix} 1 & 1 \\ \lambda_1 & \lambda_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & -2 \end{bmatrix}$$

because the eigenvalues are known to be -1 and -2, respectively. Then, from Eq. (4.2-87), the diagonal form of the system matrix is

$$\mathbf{P}^{-1} \mathbf{A}_o \mathbf{P} = \begin{bmatrix} 2 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ -1 & -2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix}$$

4.3 STATE SPACE CONCEPTS: DISCRETE SYSTEMS

In Section 3.10, we developed numerical integration techniques and applied them to approximate the solution of an n th-order differential equation by an n th-order difference equation. The algebra involved makes this approach tedious and impractical for relatively high-order systems. This difficulty is alleviated, however, when n simultaneous, first-order differential equations are used to describe the system - instead of a single n th-order differential equation.

To illustrate the concept, consider the transfer function of the second-order system given by

$$G(s) = \frac{Y(s)}{U(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (4.3-1)$$

Applying the direct decomposition scheme developed earlier, we can show that the state and output equations are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} u \quad (4.3-2)$$

and

$$y = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (4.3-3)$$

By multiplying out the above matrix equations, we obtain

$$\dot{x}_1(t) = x_2(t) \quad (4.3-4)$$

$$\dot{x}_2(t) = -\omega_n^2 x_1(t) - 2\zeta\omega_n x_2(t) + \omega_n^2 u(t) \quad (4.3-5)$$

and

$$y(t) = x_1(t) \quad (4.3-6)$$

To find the corresponding difference equations, assume that the rectangular integration rule of Eq. (3.10-8) is applied. After some manipulation, the discrete state equations are

$$x_1(k+1) = x_1(k) + T x_2(k) \quad (4.3-7)$$

$$x_2(k+1) = -\omega_n^2 T x_1(k) + (1 - 2\zeta\omega_n T) x_2(k) + \omega_n^2 T u(k) \quad (4.3-8)$$

and the output equation is

$$y(k) = x_1(k) \quad (4.3-9)$$

The above equations can be readily programmed on a digital computer by simply modifying the program shown in Fig. 3.10-7.

Eq. (4.3-7) and Eq. (4.3-8) are simultaneous, first-order difference equations which can also be written in vector-matrix form as

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T \\ -\omega_n^2 T & (1 - 2\zeta\omega_n T) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n^2 T \end{bmatrix} u(k) \quad (4.3-10)$$

where $x_1(k)$ and $x_2(k)$ are the discrete state variables. The output equation can be written as

$$y(k) = [1 \quad 0] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \quad (4.3-11)$$

In general, a n th-order discrete (discrete-time), linear, time-invariant, multivariable system can be represented by a set of state equations

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}\mathbf{u}(k) \quad (4.3-12)$$

and a set of output equations

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) \quad (4.3-13)$$

where the notation used in the discrete system case is analogous to the notation used in the continuous system case. That is, for an n th-order discrete system with p inputs and q outputs,

$\mathbf{x}(k) = (n \times 1)$ state variable vector

$\mathbf{u}(k) = (p \times 1)$ input vector

$\mathbf{W} = (n \times n)$ coefficient matrix or system matrix

$\mathbf{V} = (n \times p)$ driving or input or control matrix

$\mathbf{y}(k) = (q \times 1)$ output vector

$\mathbf{C} = (q \times n)$ output matrix

$\mathbf{D} = (q \times p)$ transmission matrix

A block diagram representation of the discrete state space equations is shown in Fig. 4.3-1.

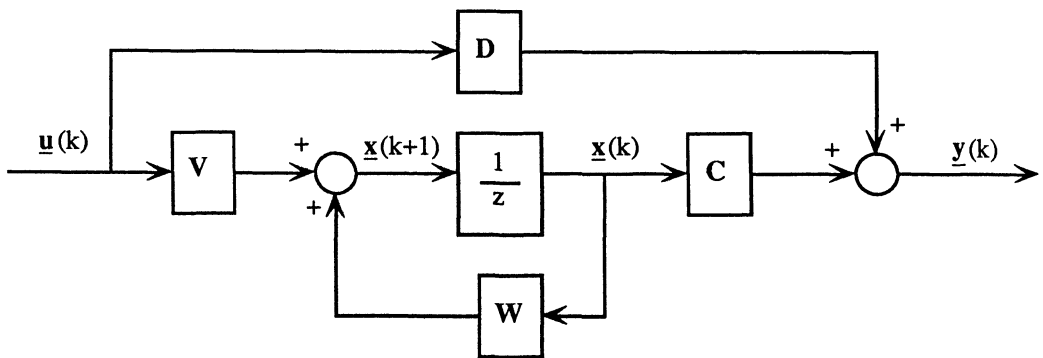


Fig. 4.3-1 Block diagram representation of the discrete state and output equations.

In general, the state space concepts developed for continuous systems can be carried over and applied to discrete systems. For instance, *the discrete state variables are defined as a minimal set of variables, $\mathbf{x}(k)$, such that knowledge of these variables at some initial discrete time $k=k_0$, together with knowledge of the inputs, $\mathbf{u}(k)$, for $k \geq k_0$, are sufficient to completely determine the state variables $\mathbf{x}(k)$ and outputs $\mathbf{y}(k)$, for $k \geq k_0$.*

4.3-1 Solution of Discrete State Equations

Consider the vector-matrix form of the discrete state equation

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}\mathbf{u}(k) \quad (4.3-14)$$

Taking the $\mathcal{Z}$ transform on both sides of the equation gives

$$z\mathbf{X}(z) - z\mathbf{x}(0) = \mathbf{W}\mathbf{X}(z) + \mathbf{V}\mathbf{U}(z)$$

Rearranging yields

$$\mathbf{X}(z) = (z\mathbf{I} - \mathbf{W})^{-1}z\mathbf{x}(0) + (z\mathbf{I} - \mathbf{W})^{-1}\mathbf{V}\mathbf{U}(z) \quad (4.3-15)$$

The inverse $\mathcal{Z}$ transform of Eq. (4.3-15) is the **discrete state transition equation**

$$\mathbf{x}(k) = \mathcal{Z}^{-1}[(z\mathbf{I} - \mathbf{W})^{-1}z]\mathbf{x}(0) + \mathcal{Z}^{-1}[(z\mathbf{I} - \mathbf{W})^{-1}\mathbf{V}\mathbf{U}(z)] \quad (4.3-16)$$

and the **discrete state transition matrix** is given by

$$\phi(k) = \mathcal{Z}^{-1}[(z\mathbf{I} - \mathbf{W})^{-1}z] \quad (4.3-17)$$

The expression for the state transition matrix given by Eq. (4.3-17) can be evaluated analytically using matrix inverse concepts and tables of $\mathcal{Z}$ transforms. Alternatively, we can make use of the following. The $\mathcal{Z}$ transform of the system matrix $\mathbf{W}$ raised to the k th power is

$$\mathcal{Z}[\mathbf{W}^k] = \sum_{k=0}^{\infty} \mathbf{W}^k z^{-k} = \mathbf{I} + \mathbf{W}z^{-1} + \mathbf{W}^2 z^{-2} + \dots \quad (4.3-18)$$

Premultiplying both sides of the equation by $\mathbf{W}z^{-1}$, we obtain

$$\mathbf{W}z^{-1}\mathcal{Z}(\mathbf{W}^k) = \mathbf{W}z^{-1} + \mathbf{W}^2 z^{-2} + \mathbf{W}^3 z^{-3} + \dots \quad (4.3-19)$$

Subtract Eq. (4.3-19) from Eq. (4.3-18), we get

$$\mathcal{Z}[\mathbf{W}^k] - \mathbf{W}z^{-1}\mathcal{Z}(\mathbf{W}^k) = \mathcal{Z}[\mathbf{W}^k](\mathbf{I} - \mathbf{W}z^{-1}) = \mathcal{Z}[\mathbf{W}^k](z\mathbf{I} - \mathbf{W})z^{-1} = \mathbf{I}$$

from which, taking the inverse $\mathcal{Z}$ transform and solving for $\mathbf{W}^k$ yields

$$\mathbf{W}^k = \mathcal{Z}^{-1}[(z\mathbf{I} - \mathbf{W})^{-1}z] \quad (4.3-20)$$

Thus, from Eq. (4.3-17), the state transition matrix becomes

$$\phi(k) = \mathbf{W}^k = \mathcal{Z}^{-1}[(z\mathbf{I} - \mathbf{W})^{-1}z] \quad (4.3-21)$$

Similarly, we can prove that the second term in Eq. (4.3-16) is given by

$$\mathcal{Z}^{-1}[(z\mathbf{I}-\mathbf{W})^{-1}\mathbf{V}\mathbf{U}(z)] = \sum_{j=0}^{k-1} \mathbf{W}^{k-j-1} \mathbf{V} \mathbf{u}(j) \quad (4.3-22)$$

which is a discrete time convolution. Finally, substituting Eq. (4.3-21) and Eq. (4.3-22) into Eq. (4.3-16), we obtain the discrete state transition equation in the alternate form

$$\mathbf{x}(k) = \mathbf{W}^k \mathbf{x}(0) + \sum_{j=0}^{k-1} \mathbf{W}^{k-j-1} \mathbf{V} \mathbf{u}(j) \quad (4.3-23)$$

Using an approach parallel to that taken in Section 4.2-1, we can evaluate the inverse of the matrix $(z\mathbf{I}-\mathbf{W})$ from the relationship

$$(z\mathbf{I}-\mathbf{W})^{-1} = \frac{1}{|z\mathbf{I}-\mathbf{W}|} \text{Adj}(z\mathbf{I}-\mathbf{W}) \quad (4.3-24)$$

where $|z\mathbf{I}-\mathbf{W}|$ is the determinant of the discrete system. The characteristic equation is given by

$$|z\mathbf{I}-\mathbf{W}| = z^n + b_1 z^{n-1} + b_2 z^{n-2} + \dots + b_n = 0 \quad (4.3-25)$$

The adjoint matrix can be evaluated using cofactors or, better yet, by Leverrier's algorithm implemented on a digital computer. The concepts are illustrated by the following examples.

Example 4.3-1

A discrete system is described by the state and output equations

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.99094 & 0.08611 \\ -0.17221 & 0.73262 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.00453 \\ 0.08611 \end{bmatrix} u(k)$$

and
$$y(k) = \begin{bmatrix} 2 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

Determine the unit-step response of the above system, assuming zero initial conditions. Then, using a sampling time of $T=0.1$ sec, determine an approximating continuous unit-step response as well as the corresponding continuous transfer function.

Since the initial conditions are zero, that is, $\mathbf{x}(0)=0$, Eq. (4.3-15) gives

$$\mathbf{X}(z) = (z\mathbf{I}-\mathbf{W})^{-1} \mathbf{V} \mathbf{U}(z)$$

and from the output equation

$$y(k) = 2x_1(k)$$

Using matrix operations, we find that

$$(z\mathbf{I}-\mathbf{W})^{-1} = \frac{1}{|z\mathbf{I}-\mathbf{W}|} \text{Adj}(z\mathbf{I}-\mathbf{W}) = \frac{1}{z^2 - 1.72357z + 0.74082} \begin{bmatrix} (z-0.73262) & 0.08611 \\ -0.17221 & (z-0.99094) \end{bmatrix}$$

Thus, the matrix of the transforms of the state variables is

$$\begin{aligned} \begin{bmatrix} X_1(z) \\ X_2(z) \end{bmatrix} &= \frac{1}{z^2 - 1.72357z + 0.74082} \begin{bmatrix} (z - 0.73262) & 0.08611 \\ -0.17221 & (z - 0.99094) \end{bmatrix} \begin{bmatrix} 0.00453 \\ 0.08611 \end{bmatrix} \begin{bmatrix} z \\ z-1 \end{bmatrix} \\ &= \frac{z}{(z-1)(z^2 - 1.72357z + 0.74082)} \begin{bmatrix} (0.00453z + 0.0041) \\ (0.08611z - 0.08611) \end{bmatrix} \end{aligned}$$

The output transform then becomes

$$Y(z) = 2X_1(z) = \frac{2z(0.00453z + 0.0041)}{(z-1)(z^2 - 1.72357z + 0.74082)} = \frac{0.00906z^2 + 0.0082}{(z-1)(z^2 - 1.72357z + 0.74082)}$$

The partial fraction expansion of $Y(z)$ gives

$$Y(z) = \frac{z}{z-1} + \frac{z}{z-0.81873} - 2 \frac{z}{z-0.90484}$$

from which, taking the inverse $\mathcal{Z}$ transform, we obtain the unit-step response

$$y(k) = 1 + (0.81873)^k - 2(0.90484)^k, \text{ for } k = 0, 1, 2, \dots$$

The above response can also be written in the equivalent form

$$y(k) = 1 + e^{-0.2k} - 2e^{-0.1k}; \text{ for } k = 0, 1, 2, \dots$$

At the sampling instants, we can substitute the continuous time variable using the relationship $t=kT$. This leads to the step response

$$y(t) = 1 + e^{-0.2t/T} - 2e^{-0.1t/T}; \text{ for } t = 0, T, 2T, \dots$$

For $T = 0.1$ sec, then, the continuous unit-step response used to approximate the discrete unit-step response at the sampling instants is

$$y(t) = 1 + e^{-2t} - 2e^{-t}, \text{ for } t \geq 0$$

Now, since $y(t)$ represents the unit-step response of the system, $y'(t)$ is its unit-impulse response. The transfer function of the system can be obtained by taking the Laplace transform of $y'(t)$. Thus,

$$y'(t) = -2e^{-2t} + 2e^{-t}$$

and the continuous transfer function becomes

$$G(s) = 2 \frac{1}{s+1} - 2 \frac{1}{s+2} = \frac{2}{s^2 + 3s + 2}$$

4.3-2 Discrete State Equations and Transfer Functions

Consider the discrete dynamic equations

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}u(k) \quad (4.3-26)$$

$$\text{and} \quad \mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}u(k) \quad (4.3-27)$$

In the last section, it was shown that

$$\mathbf{X}(z) = (z\mathbf{I} - \mathbf{W})^{-1}z\mathbf{x}(0) + (z\mathbf{I} - \mathbf{W})^{-1}\mathbf{V}U(z) \quad (4.3-28)$$

Taking the $\mathcal{Z}$ transform on each side of Eq. (4.3-27) and substituting $\mathbf{X}(z)$ from Eq. (4.3-28), we obtain

$$\mathbf{Y}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{W})^{-1}z\mathbf{x}(0) + \mathbf{C}(z\mathbf{I} - \mathbf{W})^{-1}\mathbf{V}U(z) + \mathbf{D}U(z) \quad (4.3-29)$$

To determine the system transfer function, simply let $\mathbf{x}(0)=0$ in the above equation and solve for the ratio of output-to-input transforms. This gives

$$\mathbf{G}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{W})^{-1}\mathbf{V} + \mathbf{D} \quad (4.3-30)$$

where for a multivariable system $\mathbf{G}(z)$ is a matrix of transfer functions.

Example 4.3-2

Determine the transfer function of the discrete system described by the dynamic equations

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$

$$y(k) = [-2 \quad 4] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + [1]u(k)$$

From Eq. (4.3-30),

$$\frac{\mathbf{Y}(z)}{U(z)} = \mathbf{C}(z\mathbf{I} - \mathbf{W})^{-1}\mathbf{V} + \mathbf{D}$$

Evaluating the matrix inverse quantity first, we obtain

$$(z\mathbf{I} - \mathbf{W})^{-1} = \frac{1}{z^2 - 3z + 2} \begin{bmatrix} (z-3) & 1 \\ -2 & z \end{bmatrix}$$

Therefore, the discrete transfer function is

$$\frac{\mathbf{Y}(z)}{U(z)} = [-2 \quad 4] \frac{1}{z^2 - 3z + 2} \begin{bmatrix} (z-3) & 1 \\ -2 & z \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} + 1 = \frac{z(z+1)}{z^2 - 3z + 2}$$

4.3-3 From Discrete Transfer Functions to State Equations

Assume that the transfer function or the difference equation model of a discrete system is available. Then, in order to obtain a state variable description for the discrete system, we can apply any of the transfer function decomposition techniques developed for continuous systems (see Section 4.2-3).

When comparing continuous and discrete state diagrams, the main difference is that integrators (s^{-1} elements) are replaced by "delayers" (z^{-1} elements). In addition, the outputs of the delay elements are selected to represent discrete state variables, and the inputs of the delay elements represent first-order difference terms, $x(k+1)$, rather than first-order derivative terms, $x'(t)$.

The direct, cascade, and parallel schemes used to decompose a discrete transfer function $G(z)$ are illustrated by means of examples.

Example 4.3-3 Direct Decomposition

Determine a state space model for the discrete transfer function

$$G(z) = \frac{Y(z)}{U(z)} = \frac{z^2 + z}{z^2 - 1.5z + 0.5}$$

Dividing the numerator and denominator by z^2 , we can write the transfer function as

$$\frac{Y(z)}{U(z)} = \frac{(1+z^{-1})E(z)}{(1-1.5z^{-1}+0.5z^{-2})E(z)}$$

where $E(z)$ is a dummy variable. Equating the numerators and denominators, we obtain

$$Y(z) = (1 + z^{-1}) E(z)$$

and

$$E(z) = U(z) + 1.5z^{-1}E(z) - 0.5z^{-2}E(z)$$

The discrete state diagram is shown in Fig 4.3-2. The corresponding discrete dynamic equations can now be written, assuming first that the delay elements have been removed from the state diagram. The state and output equations in vector-matrix form are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.5 & 1.5 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$

and

$$y(k) = [-0.5 \quad 2.5] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + [1]u(k)$$

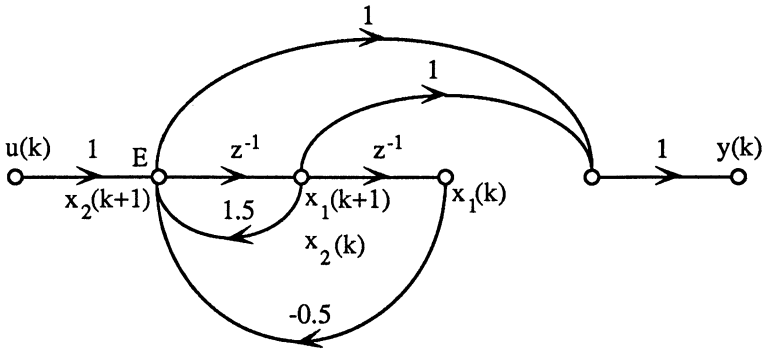


Fig. 4.3-2 Discrete state diagram for system of Example 4.3-3.

Example 4.3-4 Cascade Decomposition

Determine the discrete dynamic equations for the system of Example 4.3-3 using cascade decomposition.

The transfer function can be written in factored form as

$$\frac{Y(z)}{U(z)} = \left(\frac{z+1}{z-0.5} \right) \left(\frac{z}{z-1} \right)$$

The corresponding state diagram is shown on Fig. 4.3-3. From the state diagram, the state and output equations are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 1.5 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u(k)$$

and

$$y(k) = [1 \quad 1.5] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + [1]u(k)$$

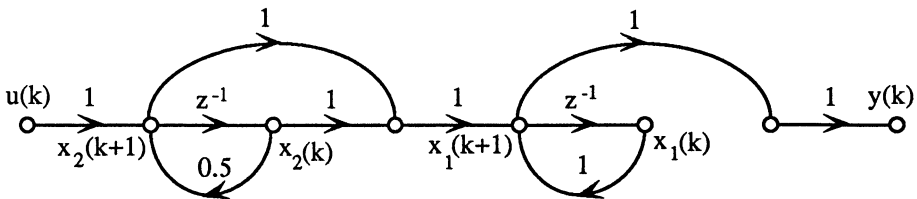


Fig. 4.3-3 Discrete state diagram for system of Example 4.3-4.

Example 4.3-5 Parallel Decomposition

Determine the discrete dynamic equations for the system of Example 4.3-3 using parallel decomposition.

After performing partial fraction expansion, the transfer function can be expressed in the form

$$\frac{Y(z)}{U(z)} = \frac{4z}{z-1} - \frac{3z}{z-0.5}$$

The state diagram is shown in Fig. 4.3-4. The corresponding state and output equations are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u(k)$$

and

$$y(k) = [4 \quad -1.5] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + [1]u(k)$$

Note that the diagonal form of the system matrix has the system eigenvalues 1 and 0.5 as the main diagonal elements.

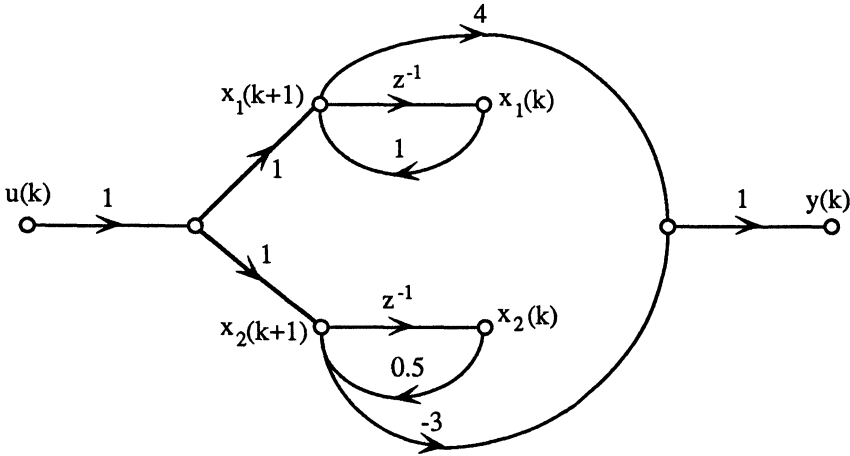


Fig. 4.3-4 Discrete state diagram for system of Example 4.3-5.

Example 4.3-6 Parallel Decomposition, Multiple Poles

As another example of parallel decomposition, consider the third-order system described by the transfer function

$$\frac{Y(z)}{U(z)} = \frac{z(2z-3)}{(z-2)(z-1)^2}$$

The above transfer function has repeated real poles at $z=1$, and was expanded into partial fractions in Example 3.5-9. Thus, it can be written as

$$\frac{Y(z)}{U(z)} = \frac{z}{z-2} - \frac{z}{z-1} + \frac{z}{(z-1)^2}$$

Even though this last expression seems to imply four delay elements in the discrete state diagram, the correct number of delay elements is governed by the order of the system, in this case three. Therefore, one delay element must be shared; otherwise, a dependent set of equations will result. To make things easier to visualize, rewrite the previous equation in the form

$$\frac{Y(z)}{U(z)} = \frac{z}{z-2} + \frac{z}{z-1} \left(\frac{1}{z-1} - 1 \right)$$

and think of the $z/(z-1)$ term as the shared term. This leads to the state diagram shown in Fig. 4.3-5. From this diagram, we obtain the discrete dynamic equations

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+2) \\ x_3(k+3) \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 2 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix}$$

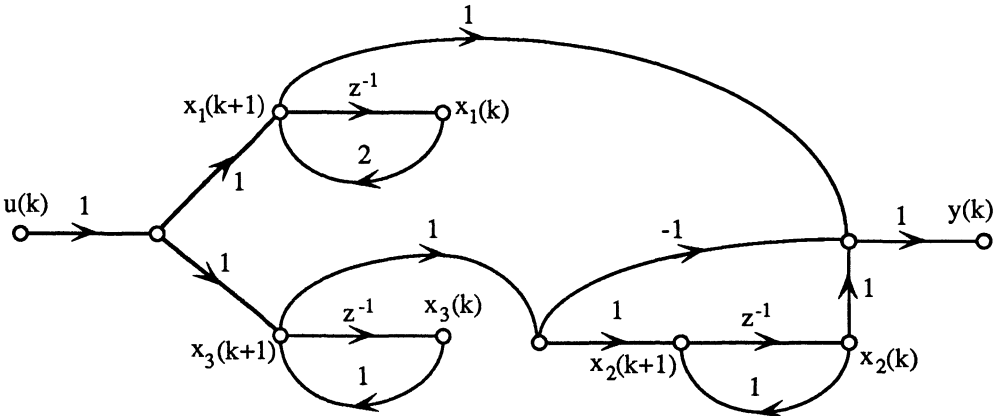


Fig. 4.3-5 Discrete state diagram for system of Example 4.3-6.

Coupled Decomposition. Another form of decomposition suitable for a second-order transfer function with complex poles is the **coupled** form. The complex conjugate poles p_1 and p_1^* are shown in Fig. 4.3-6. The denominator of the transfer function, that is, the characteristic polynomial, is given by

$$\begin{aligned}(z-p_1)(z+p_1^*) &= z^2 - (p_1+p_1^*)z + p_1p_1^* = z^2 - 2\text{Re}(p_1)z + [\text{Re}^2(p_1)+\text{Im}^2(p_1)] \\ &= z^2 - 2rz\cos\theta + (r^2\cos^2\theta + r^2\sin^2\theta) = z^2 - 2rz\cos\theta + r^2\end{aligned}\quad (4.3-1)$$

A state diagram (structure) which has the above characteristic equation is shown in Fig. 4.3-7. From the state diagram, the state equations are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} r\cos\theta & r\sin\theta \\ -r\sin\theta & r\cos\theta \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} g_2 \\ g_1 \end{bmatrix} u(k)\quad (4.3-2)$$

and the output equation

$$y(k) = [1 \quad 0] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + [a_o]u(k)\quad (4.3-3)$$

The corresponding transfer function $Y(z)/U(z)$ can be obtained using the above equations and $\mathcal{Z}$ transforms, or from the state diagram using Mason's gain formula. Either approach leads to the transfer function

$$\frac{Y(z)}{U(z)} = a_o + \frac{g_2(z - r\cos\theta) + g_1r\sin\theta}{z^2 - 2rz\cos\theta + r^2}\quad (4.3-4)$$

For the general case, a second-order transfer function can be written in the form

$$G(z) = \frac{a_0z^2 + a_1z + a_2}{z^2 + b_1z + b_2}\quad (4.3-5)$$

Equating coefficients between Eq. (4.3-5) and Eq. (4.3-4), after cross-multiplication to normalize it, we obtain the coefficients for the coupled form decomposition as

$$r\cos\theta = -\frac{b_1}{2}\quad (4.3-6)$$

$$r\sin\theta = \sqrt{b_2 - \frac{b_1^2}{4}}\quad (4.3-7)$$

$$g_1 = \frac{a_2 + a_0\left(\frac{b_1^2}{2} - b_2\right) - \frac{b_1a_1}{2}}{\sqrt{b_2 - \frac{b_1^2}{4}}}\quad (4.3-8)$$

$$g_2 = a_1 - a_0b_1\quad (4.3-9)$$

with a_o staying the same.

There are a couple of advantages of the coupled form implementation over the direct form implementation. When the parameters are quantized, the pole/zero locations are also quantized. The coupled form realization leads to z -plane poles located near $z=1$, which result in reduced quantization error. Also, the transfer function denominator parameters

for the coupled form are all between 0 and 1, making the software implementation simpler.

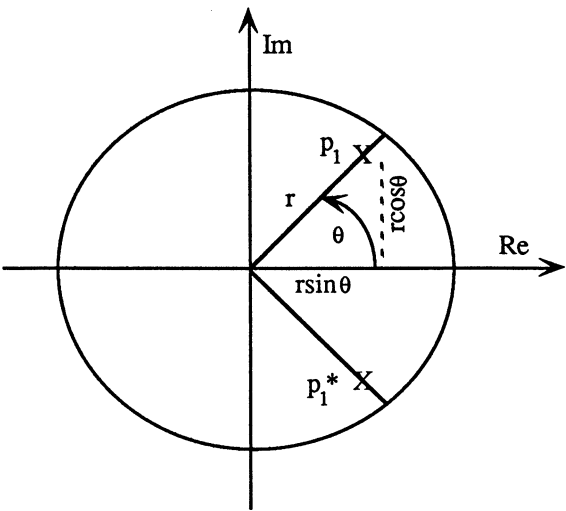


Fig. 4.3-6 z-plane diagram showing complex pair of poles.

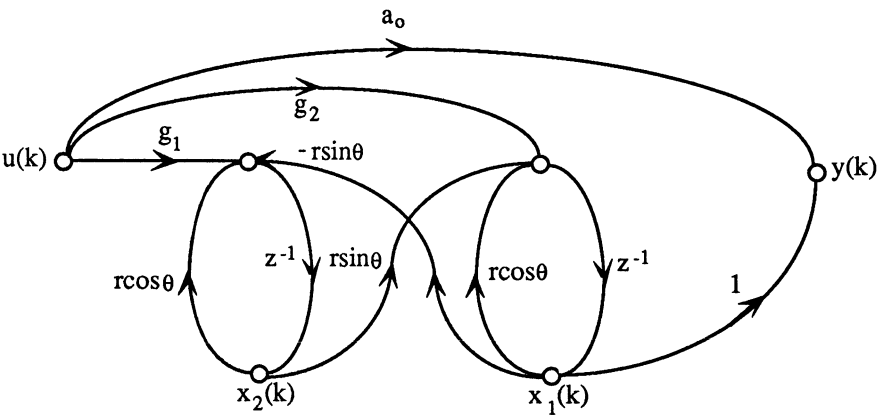


Fig. 4.3-7 State diagram for coupled form decomposition.

Example 4.3-7 Coupled Decomposition, Complex Poles

Determine the relevant parameters to implement the following transfer function using the coupled decomposition form.

$$G(z) = \frac{Y(z)}{U(z)} = \frac{0.65z}{z^2 - 0.6z + 0.25}$$

From Eq. (4.3-5, the given values are $a_0=a_2=0$, $a_1=0.65$, $b_1=-0.6$, and $b_2=0.25$. Then, using equations (4.3-6)-(4.3-9), the parameters values for the coupled structure of Fig. 4.3-7) are

$$r\cos=0.3, r\sin=0.4, g_1=0.4875, \text{ and } g_2=0.65. \text{ Also, } r=\sqrt{b_2} = 0.5.$$

4.4 STATE EQUATIONS: DISCRETE FROM CONTINUOUS

The vector-matrix form for the continuous state equations is

$$\dot{\mathbf{x}}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (4.4-1)$$

and for the discrete state equations the form is

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}\mathbf{u}(k) \quad (4.4-2)$$

Multiplying both sides of Eq. (4.4-1) by $e^{-\mathbf{A}t}$ and rearranging gives

$$e^{-\mathbf{A}t}\dot{\mathbf{x}}'(t) - e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t) = e^{-\mathbf{A}t}\mathbf{B}\mathbf{u}(t) \quad (4.4-3)$$

Eq. (4.4-3) can be written as

$$\frac{d}{dt} [e^{-\mathbf{A}t}\mathbf{x}(t)] = e^{-\mathbf{A}t}\mathbf{B}\mathbf{u}(t) \quad (4.4-4)$$

Integrating Eq. (4.4-4) from kT to $(k+1)T$ results in

$$e^{-\mathbf{A}(k+1)T}\mathbf{x}(k+1) - e^{-\mathbf{A}kT}\mathbf{x}(k) = \int_{kT}^{(k+1)T} e^{-\mathbf{A}\tau}\mathbf{B}\mathbf{u}(\tau)d\tau \quad (4.4-5)$$

where τ is a dummy variable. For responses at the sampling instants only, the above equation becomes

$$\mathbf{x}(k+1) = e^{\mathbf{A}T}\mathbf{x}(k) + e^{\mathbf{A}(k+1)T} \int_{kT}^{(k+1)T} e^{-\mathbf{A}\tau}\mathbf{B}\mathbf{u}(\tau)d\tau \quad (4.4-6)$$

In Eq. (4.4-6), T represents the sampling time and the term $e^{\mathbf{A}T}$ is the familiar state transition matrix defined by

$$\phi(T) = e^{\mathbf{A}T} = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k T^k}{k!} \quad (4.4-7)$$

Assuming that the input is piecewise constant in the interval of integration, we can write Eq. (4.4-6) as

$$\mathbf{x}(k+1) = e^{\mathbf{A}T}\mathbf{x}(k) + \left\{ \int_{kT}^{(k+1)T} e^{\mathbf{A}[(k+1)T-\tau]} d\tau \right\} \mathbf{B}\mathbf{u}(k) \quad (4.4-8)$$

Substituting Eq. (4.4-7) into Eq. (4.4-8) gives

$$\mathbf{x}(k+1) = \left[\sum_{k=0}^{\infty} \frac{\mathbf{A}^k \mathbf{T}^k}{k!} \right] \mathbf{x}(k) + \left\{ \int_0^{\mathbf{T}} \sum_{k=0}^{\infty} \frac{\mathbf{A}^k [(k+1)\mathbf{T}-\tau]^k}{k!} d\tau \right\} \mathbf{B} \mathbf{u}(k) \quad (4.4-9)$$

Interchanging summation and integration in the last expression and evaluating the integral leads to the result

$$\begin{aligned} \mathbf{x}(k+1) &= \left[\sum_{k=0}^{\infty} \frac{\mathbf{A}^k \mathbf{T}^k}{k!} \right] \mathbf{x}(k) + \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{k!} \left\{ \int_0^{\mathbf{T}} [(k+1)\mathbf{T}-\tau]^k d\tau \right\} \mathbf{B} \mathbf{u}(k) \\ &= \left(\sum_{k=0}^{\infty} \frac{\mathbf{A}^k \mathbf{T}^k}{k!} \right) \mathbf{x}(k) + \left[\sum_{k=0}^{\infty} \frac{\mathbf{A}^k \mathbf{T}^{k+1}}{(k+1)!} \right] \mathbf{B} \mathbf{u}(k) \end{aligned} \quad (4.4-10)$$

Equating terms in Eq. (4.4-10) and Eq. (4.4-2), defines the discrete state equation matrices in terms of the continuous state equation matrices. That is, the discrete system matrix $\mathbf{W}$ is expressed in terms of the continuous system matrix $\mathbf{A}$ by

$$\mathbf{W} = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k \mathbf{T}^k}{k!} \quad (4.4-11)$$

Also, the discrete input matrix $\mathbf{V}$ is given as a function of the continuous matrices $\mathbf{A}$ and $\mathbf{B}$ by

$$\mathbf{V} = \left[\sum_{k=0}^{\infty} \frac{\mathbf{A}^k \mathbf{T}^{k+1}}{(k+1)!} \right] \mathbf{B} \quad (4.4-12)$$

The $\mathbf{W}$ and $\mathbf{V}$ matrices given by Eq. (4.4-11) and Eq. (4.4-12), respectively, can be evaluated for a given $\mathbf{T}$ and then used to determine the response of the continuous system described by Eq. (4.4-1).

An important implication of the state variable formulation using the above $\mathbf{W}$ and $\mathbf{V}$ matrices is that the transfer function which results from Eq. (4.3-30) represents the pulse transfer function of the system. This is expected, because in the development of the $\mathbf{W}$ and $\mathbf{V}$ relationships, it was assumed that the input is piecewise constant, which is equivalent to assuming that the continuous system is preceded by a zero-order-hold (ZOH) element. This last assumption satisfies the definition of the pulse transfer function. The pulse transfer function can be evaluated on the computer using Leverrier's algorithm.

Example 4.4-1

Evaluate the discrete matrix $\mathbf{W}$ for the continuous system described by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Assume a sampling time of $\mathbf{T}=0.1$ sec.

Using only three terms in the summation of Eq. (4.4-11), we obtain

$$\begin{aligned} \mathbf{W} &= \mathbf{I} + \mathbf{T}\mathbf{A} + \frac{\mathbf{T}^2\mathbf{A}^2}{2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + 0.1 \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \\ &+ 0.005 \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} = \begin{bmatrix} 0.99 & 0.085 \\ -0.17 & 0.735 \end{bmatrix} \end{aligned}$$

This is the same system used in Example 4.3-1. Note that even though we only used three terms to approximate the $\mathbf{W}$ matrix, the results approximate rather well the values obtained in Example 4.3-1.

4.5 DIGITAL SIMULATION USING STATE SPACE MODELS

Programming Eq. (4.4-11) and Eq. (4.4-12) results in a powerful scheme for simulating continuous systems on a digital computer. The common terms in the respective $\mathbf{W}$ and $\mathbf{V}$ matrices suggest the following computational algorithm.

$$\mathbf{W} = [\mathbf{I} \quad + \quad \mathbf{AT} \quad + \quad \frac{\mathbf{A}^2\mathbf{T}^2}{2} \quad + \quad \dots] \quad (4.5-1)$$

$$\mathbf{V} = [\mathbf{IT} \quad + \quad \frac{\mathbf{AT}^2}{2} \quad + \quad \frac{\mathbf{A}^2\mathbf{T}^3}{3!} \quad + \quad \dots] \mathbf{B} \quad (4.5-2)$$

For efficiency, the first term $\mathbf{IT}$ in the $\mathbf{V}$ matrix can be calculated by multiplying the first term $\mathbf{I}$ in the $\mathbf{W}$ matrix by $\mathbf{T}$. The $\mathbf{AT}$ term in the $\mathbf{W}$ matrix is calculated next by multiplying the $\mathbf{IT}$ term by $\mathbf{A}$, the $\mathbf{AT}$ term is multiplied by $\mathbf{T}/2$ to obtain $\mathbf{AT}^2/2$, and so forth. Thus, the time it takes to compute $\mathbf{V}$ is practically the same as the time it takes to compute $\mathbf{W}$.

In general, the above scheme results in excellent simulation accuracy with about 20-30 terms in the series expansion. $\mathbf{T}$ is primarily selected for overall response resolution. That is, to describe the response adequately, one selects the spacing of the points $\mathbf{T}$, and the number of points necessary to describe the response. In cases of frequency dependent inputs, care must be exercised in choosing $\mathbf{T}$ so that the "twice-the-highest-frequency" rule for sampling the input signal, as dictated by the sampling theorem, is not violated.

Even though the choice of $\mathbf{T}$ does not appear to be critical, certain logical steps should be followed. For instance, if the time constants of the system being simulated are available, the sampling time $\mathbf{T}$ should not be longer than the shortest time constant and the simulation should run at least four times the longest time constant. Furthermore, if the poles of the system include one or more complex pairs, the response contains oscillations and $\mathbf{T}$ should be chosen to adequately describe the response by providing enough points per cycle of oscillation.

In the case of a second-order system, the frequency of oscillation is the damped natural frequency and about ten points per cycle should describe the response adequately. However, when a system time constant indicates that the response settles before an oscillation cycle is complete, $\mathbf{T}$ should be chosen to adequately describe the response in four or five time constants.

Example 4.5-1

Determine a discrete state space model for the RLC network used in Example 4.2-1. Assume a sampling time $T=0.04$ sec. Also, determine the corresponding pulse transfer function for the system.

The continuous state equations for the RLC network are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 10 \\ -10 & -10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 10 \end{bmatrix} v_1(t)$$

and the output equation is

$$v_2(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Recall that the state variables $x_1(t)$ and $x_2(t)$ were conveniently selected to represent the important physical quantities of capacitor voltage and inductor current, respectively. Using Eq. (4.4-11) and Eq. (4.4-12), we obtain the discrete state equations

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.93059 & 0.32098 \\ -0.32098 & 0.60961 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} 0.06941 \\ 0.32098 \end{bmatrix} v_1(k)$$

and the output equation

$$v_2(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

Simulation results for a unit-step function input are shown in Table 4.5-1. Analytical expressions for the $x_1(t)$ and $x_2(t)$ continuous state variables were obtained in Example 4.2-1. Plots of these variables are shown in Fig. 4.2-5.

The pulse transfer function of the network can be calculated by either using Eq. (4.3-30) and the above matrices, or by applying Leverrier's algorithm. Using the latter method on the computer, results in the pulse transfer function

$$G(z) = \frac{0.06941z + 0.06715}{z^2 - 1.54019z + 0.67032}$$

Note that for this RLC network, the continuous transfer function was evaluated in previous examples as

$$G(s) = \frac{10}{s^2 + 10s + 100}$$

Thus, the pulse transfer function for $T=0.04$ sec can also be determined from the relationship

$$G_h G(z) = \mathcal{Z} \left[\frac{1-e^{-sT}}{s} \frac{10}{s^2 + 10s + 100} \right]$$

Table 4.5-1 Computer simulation results for Example 4.5-1.

Time	$x_1(t)$	$x_2(t)$	Output
0.00	0.00000	0.00000	0.00000
0.04	0.06941	0.32098	0.06941
0.08	0.23704	0.49437	0.23704
0.12	0.44868	0.54627	0.44868
0.16	0.66229	0.50997	0.66229
0.20	0.84943	0.41928	0.84943
0.24	0.99446	0.30393	0.99446
0.28	1.09240	0.18705	1.09240
0.32	1.14603	0.08437	1.14603
0.36	1.16297	0.00456	1.16297
0.40	1.15312	-0.04953	1.15312
0.44	1.12660	-0.07934	1.12660
0.48	1.09234	-0.08900	1.09234
0.52	1.05736	-0.08390	1.05736
0.56	1.02645	-0.06956	1.02645
0.60	1.00229	-0.05089	1.00229
0.64	0.98580	-0.03176	0.98580
0.68	0.97659	-0.01480	0.97659
0.72	0.97346	-0.00151	0.97346
0.76	0.97482	0.00760	0.97482
0.80	0.97901	0.01272	0.97901
0.84	0.98455	0.01449	0.98455
0.88	0.99027	0.01379	0.99027
0.92	0.99537	0.01153	0.99537
0.96	0.99939	0.00852	0.99939
1.00	1.00217	0.00539	1.00217

4.6 CONTROLLABILITY AND OBSERVABILITY

Two important concepts associated with modern control theory are the concepts of **controllability** and **observability**. Controllability is concerned with determining the state vector from the input vector and, thus, involves the system and input matrices. Observability, on the other hand, is concerned with determining the state vector from the output vector and, thus, involves the system and output matrices.

By definition, a continuous or discrete system is said to be **controllable at some initial time**, if there exists some input that can drive the initial state to any final state in finite time. If this is true for all initial times and all initial states, the system is said to be **completely state controllable** or simply **state controllable**.

For an n th-order, continuous system described by the dynamic equations

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (4.6-1)$$

and

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) \quad (4.6-2)$$

the necessary and sufficient condition for complete controllability is that the rank of the so-called controllability matrix is equal to n . That is,

$$\text{Rank}[\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A^2B} \quad \dots \quad \mathbf{A^{n-1}B}] = n \quad (4.6-3)$$

Equivalently, for a single-input system to be completely controllable, the above matrix must be nonsingular, that is, its determinant must not be zero.

Observe that the matrix of Eq. (4.6-3) was first encountered as Eq. (4.2-76). In that discussion, the nonsingularity property of the matrix was stated as a condition that must be satisfied before a system described by its dynamic equations can be transformed into phase-variable canonical form. It follows from the above discussion that if a system is completely controllable, only then can it be transformed into phase-variable canonical form.

Consider the discrete dynamic equations

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}\mathbf{u}(k) \quad (4.6-4)$$

and

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) \quad (4.6-5)$$

where, for simplicity, we have assumed that the $\mathbf{D}$ matrix is zero. Again, any system whose transfer function has more poles than zeros can be expressed in this form.

For the above system to be controllable, there must exist an input sequence $\mathbf{u}(k)$ that can drive the system from some arbitrary initial state $\mathbf{x}(0)$ to the specific final state $\mathbf{x}(n)$ in n sampling intervals. From Eq. (4.3-23), we have

$$\mathbf{x}(k) = \mathbf{W}^k\mathbf{x}(0) + \sum_{j=0}^{k-1} \mathbf{W}^{k-j-1}\mathbf{V}\mathbf{u}(j) \quad (4.6-6)$$

and in n sampling intervals

$$\mathbf{x}(n) = \mathbf{W}^n\mathbf{x}(0) + \sum_{j=0}^{n-1} \mathbf{W}^{n-j-1}\mathbf{V}\mathbf{u}(j) \quad (4.6-7)$$

The last equation can be rewritten as

$$\mathbf{x}(n) - \mathbf{W}^n\mathbf{x}(0) = \mathbf{W}^{n-1}\mathbf{V}\mathbf{u}(0) + \mathbf{W}^{n-2}\mathbf{V}\mathbf{u}(1) + \dots + \mathbf{W}\mathbf{V}\mathbf{u}(n-2) + \mathbf{V}\mathbf{u}(n-1) \quad (4.6-8)$$

which in matrix form becomes

$$\mathbf{x}(n) - \mathbf{W}^n\mathbf{x}(0) = [\mathbf{V} \quad \mathbf{WV} \quad \dots \quad \mathbf{W}^{n-2}\mathbf{V} \quad \mathbf{W}^{n-1}\mathbf{V}] \begin{bmatrix} \mathbf{u}(n-1) \\ \mathbf{u}(n-2) \\ \vdots \\ \mathbf{u}(0) \end{bmatrix} \quad (4.6-9)$$

Since Eq. (4.6-9) represents a set of n simultaneous equations, for a solution to exist, the rank of the matrix of the coefficients must be the same as the number of unknowns. That is,

$$\text{Rank}[\mathbf{V} \quad \mathbf{WV} \quad \dots \quad \mathbf{W}^{n-2}\mathbf{V} \quad \mathbf{W}^{n-1}\mathbf{V}] = n \quad (4.6-10)$$

If the condition stated by Eq. (4.6-10) is satisfied, it means that there exists an input sequence $\mathbf{u}(k)$ that can drive the system defined by the $\mathbf{W}$ and $\mathbf{V}$ matrices from an initial state to a desired final state in n steps. The problem of finding an appropriate input sequence, known as the **control sequence**, is considered in Chapter 6. If, however,

the $\mathbf{W}$ and $\mathbf{V}$ matrices do not satisfy the condition for controllability, the system is **uncontrollable** and no controlling sequence can be found to drive it from an initial state to a final state in finite time.

Observability is viewed as the dual concept of controllability. By definition, *a linear system is said to be observable if every initial state can be determined from the observation of the output over a finite interval.* Thus, the discrete system described by Eq. (4.6-4) and Eq. (4.6-5) is observable provided that the initial state $\mathbf{x}(0)$ can be calculated from measurements $\mathbf{y}(0), \mathbf{y}(1), \dots, \mathbf{y}(n-1)$, where n is finite.

To derive the appropriate criteria for observability, consider the discrete dynamic equations defined by Eq. (4.6-4) and Eq. (4.6-5). Substituting the state vector equation given by Eq. (4.3-23) in the output equation of Eq. (4.6-5), leads to the expression

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) = \mathbf{C}\mathbf{W}^k\mathbf{x}(0) + \mathbf{C}\sum_{j=0}^{k-1}\mathbf{W}^{k-j-1}\mathbf{V}\mathbf{u}(j) \quad (4.6-11)$$

Rewriting the above equation in the form

$$\mathbf{y}(k) - \mathbf{C}\sum_{j=0}^{k-1}\mathbf{W}^{k-j-1}\mathbf{V}\mathbf{u}(j) = \mathbf{C}\mathbf{W}^k\mathbf{x}(0) \quad (4.6-12)$$

and evaluating it for $k = 0, 1, 2, \dots, n-1$, results in the equations

$$\mathbf{y}(0) = \mathbf{C}\mathbf{x}(0) \quad (4.6-13)$$

$$\mathbf{y}(1) - \mathbf{C}\mathbf{V}\mathbf{u}(0) = \mathbf{C}\mathbf{W}\mathbf{x}(0) \quad (4.6-14)$$

$$\mathbf{y}(2) - \mathbf{C}\mathbf{W}\mathbf{V}\mathbf{u}(0) - \mathbf{C}\mathbf{V}\mathbf{u}(1) = \mathbf{C}\mathbf{W}^2\mathbf{x}(0) \quad (4.6-15)$$

$$\vdots$$

$$\mathbf{y}(n-1) - \mathbf{C}\sum_{j=0}^{n-2}\mathbf{W}^{n-2-j}\mathbf{V}\mathbf{u}(j) = \mathbf{C}\mathbf{W}^{n-1}\mathbf{x}(0) \quad (4.6-16)$$

To simplify the derivation of the observability condition, assume that $\mathbf{u}(k)=0$. Then, the above equations can be written in matrix form as

$$\begin{bmatrix} \mathbf{C} \\ \mathbf{C}\mathbf{W} \\ \mathbf{C}\mathbf{W}^2 \\ \vdots \\ \mathbf{C}\mathbf{W}^{n-1} \end{bmatrix} \mathbf{x}(0) = \begin{bmatrix} \mathbf{y}(0) \\ \mathbf{y}(1) \\ \vdots \\ \mathbf{y}(n-1) \end{bmatrix} \quad (4.6-17)$$

For the above set of n equations in n unknowns to have a unique solution, the rank of the matrix of coefficients must be n . Thus, for a discrete system to be observable, it is necessary and sufficient that

$$\text{Rank} \begin{bmatrix} C \\ C W \\ C W^2 \\ \vdots \\ C W^{n-1} \end{bmatrix} = n \quad (4.6-18)$$

The matrix of coefficients is often referred to as the **observability matrix**. Thus, if the rank of the observability matrix is n , then a solution vector $\mathbf{x}(0)$ can be found and this vector will be unique.

In the case of a continuous system, the necessary and sufficient condition for observability becomes

$$\text{Rank} \begin{bmatrix} C \\ C A \\ C A^2 \\ \vdots \\ C A^{n-1} \end{bmatrix} = n \quad (4.6-19)$$

The controllability and observability concepts are illustrated next by examples.

Example 4.6-1

Determine the controllability and observability of a continuous system described by the dynamic equations

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

The controllability matrix is given by

$$[B \quad AB] = \begin{bmatrix} 0 & 1 \\ 1 & -3 \end{bmatrix}$$

The value of the determinant is -1 , not zero, and hence the rank is 2. Thus, the system is completely controllable. The observability matrix is given by

$$\begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix}$$

which is singular, because the value of the determinant is zero. The rank of the above matrix is 1, which is less than n . Therefore, the system is unobservable. This means that there are nonzero initial states that cannot be determined from the measurement of the output $y(t)$.

The system of the previous example has the transfer function

$$\frac{Y(s)}{U(s)} = \frac{s+1}{s^2+3s+2} = \frac{s+1}{(s+1)(s+2)} = \frac{1}{s+2}$$

which has an identical pole and zero at $s=-1$. The relationship between controllability and observability and the pole-zero cancellation of a transfer function is defined by the following theorem, which is stated without proof.

Theorem. If the transfer function has pole-zero cancellation, the system will be either uncontrollable or unobservable, depending on how the state variables are defined. If the transfer function does not have a pole-zero cancellation, however, the system can always be represented by dynamic equations as a completely controllable and observable system.

Example 4.6-2

Determine the controllability and observability of the system described by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

and
$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

From Eq. (4.6-3), the controllability matrix is

$$[\mathbf{B} \quad \mathbf{AB}] = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$$

which is singular and, thus, the system is uncontrollable.

From Eq. (4.6-19), the observability matrix is

$$\begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

which is of rank 2. Hence the system is observable.

Example 4.6-3

Determine the controllability and observability of the system described by the discrete dynamic equations of Example 4.5-1. The equations are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.93059 & 0.32098 \\ -0.32098 & 0.60961 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.06941 \\ 0.32098 \end{bmatrix} u(k)$$

and
$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

From Eq. (4.6-10), the discrete observability matrix is

$$[\mathbf{V} \quad \mathbf{WV}] = \begin{bmatrix} 0.06941 & 0.16700 \\ 0.32098 & 0.17339 \end{bmatrix}$$

which is nonsingular and, thus, the system is completely controllable. From Eq. (4.6-18), the observability matrix is

$$\begin{bmatrix} C \\ CW \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0.93059 & 0.32098 \end{bmatrix}$$

which is nonsingular and implies a completely observable system.

4.7 CASE STUDY: ANALYSIS OF A MOTOR DRIVE SYSTEM

The objective of this study is to illustrate how some of the mathematical tools developed this far can be applied to analyze the motor drive system shown in block diagram form in Fig. 4.7-1. The system consists of a **direct current (DC) motor** driven by a **pulse-width-modulated (PWM)** power supply. A DAC and a digital tach or pulse generator (PG) are used as interfaces so that the analog plant can communicate with the digital computer for testing and/or control purposes.

The DC motor is an electromechanical device which converts electrical into mechanical energy and delivers it to a load. The motor and voltage or current amplifier combination constitutes what is commonly referred to as a **motor drive plant**, which provides the power element in many closed-loop control applications.

In determining a practical model for the DC motor drive plant shown in Fig. 4.7-1, all linear parameters have been assumed, and secondary effects such as hysteresis and the voltage drop across the brushes have been neglected. The following nomenclature is used:

$i_r(t)$	= reference or command current (volts)
$i(t)$	= armature current (amps)
$v_a(t)$	= armature voltage (volts)
$d(t)$	= torque disturbance
$\omega(t)$	= speed/velocity or shaft rate (rad/sec)
$\omega(k)$	= speed/velocity (pulses/T)
$\theta(t)$	= shaft position/displacement (rad)
$\theta(k)$	= shaft position/displacement (pulses)
T	= sampling time
$m(k)$	= manipulation or controller output (pulses/T)
L	= armature inductance
R	= armature resistance
J_M	= motor inertia
J_L	= load inertia
J_T	= total inertia ($J_M + J_L$)
F	= motor friction
K_t	= motor torque constant
K_b	= back electromotive force (EMF) gain
K_c	= current feedback gain
K_1, K_2	= current-loop (PI) controller gains
K_{da}	= Digital-to-Analog Converter (DAC) gain
K_{dt}	= digital tachometer gain (pulses/rev)

It is required to carry out the following two-part investigation:

A. Modeling of a Motor Drive System

- (1) Draw an equivalent state diagram for the motor drive plant and determine its state and output equations. Assume that both $i_r(t)$ and $d(t)$ inputs present.
- (2) Determine the position-to-current and position-to-disturbance transfer functions $\Theta(s)/I_r(s)$ and $\Theta(s)/D(s)$.

B. Analysis of a Motor Drive System

Perform (1) and (2), below, assuming the following parameters settings:

$$\begin{aligned}
 R &= 0.294 \text{ (ohms)}, & L &= 0.002 \text{ (henries)} \\
 JM &= 0.29 \text{ (lb-in-sec}^2\text{)}, & JL &= 0.1 \text{ (lb-in-sec}^2\text{)} \\
 K_b &= 0.75 \text{ (volt/rad/sec)}, & K_c &= 0.07574 \text{ (volt/amp)} \\
 K_1 &= 3.24, & K_2 &= 540, & K_t &= 6.67 \text{ (lb-in)/amp} \\
 K_{da} &= \left(\frac{10}{2047}\right) \left(\frac{\text{volts}}{\text{pulses}}\right), \text{ for a 12-bit, } \pm 10 \text{ volt DAC} \\
 K_{dt} &= (4 \cdot 2000) \text{ (pulses/revolution)} = (8000/2\pi) \text{ (pulses/rad)} \\
 T &= 0.005 \text{ sec, and } F=0
 \end{aligned}$$

- (1) Evaluate the state variables of the system, including the discrete position $\theta(k)$ in pulses, when the input (manipulation variable) is a step of 200 pulses; that is, when $m(k)=200u(k)$ is applied by the computer.
- (2) Determine the discrete transfer functions $\Theta(z)/M(z)$ and $\Omega(z)/M(z)$. Visualize the complete system of Fig. 1 as shown in Fig. 2. That is, assume that the motor drive plant is represented by the transfer function $G_p(s)$. This makes the approach general and, thus, applicable to other types of plants.

From Fig. 4.7-2, the position-to-manipulation discrete transfer function, $\Theta(z)/M(z)$, can be obtained from the relationship

$$\frac{\Theta(z)}{M(z)} = K_{da} G_h G_p(z) K_{dt} \quad (4.7-1)$$

where $G_h G_p(z)$ is the pulse transfer function defined by

$$G_h G_p(z) = \mathcal{Z}[G_h(s) G_p(s)] \quad (4.7-2)$$

The state diagram for the motor drive plant is shown in Fig. 4.7-3. The respective state and output equations can be expressed in vector-matrix form as

$$\begin{bmatrix} x'_1(t) \\ x'_2(t) \\ x'_3(t) \\ x'_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{F}{J} & \frac{K_t}{J} & 0 \\ 0 & \frac{K_b}{L} & -\frac{R+K_1 K_c}{L} & \frac{1}{L} \\ 0 & 0 & -K_2 K_c & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{J} \\ \frac{K_1}{L} & 0 \\ K_2 & 0 \end{bmatrix} \begin{bmatrix} i_r(t) \\ d(t) \end{bmatrix} \quad (4.7-3)$$

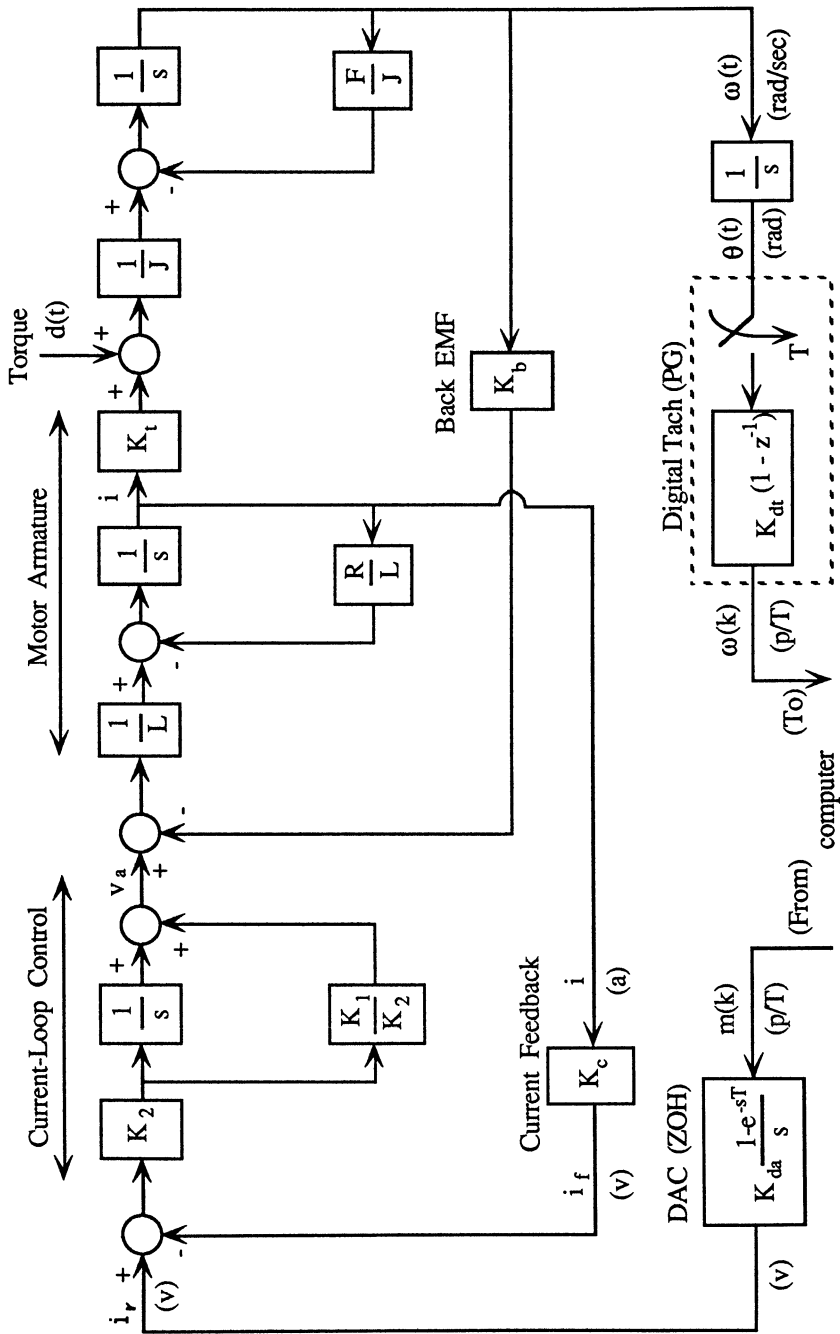


Fig. 4-7-1 Block diagram of a DC motor drive system, with DAC and PG interfaces.

$$\text{and} \quad \theta(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} \quad (4.7-4)$$

From the labeling used in the state diagram of Fig. 4.7-2, $x_1(t)$ represents the position variable in rad, $x_2(t)$ the velocity in rad/sec, and $x_3(t)$ the armature current in amps. Note that to obtain position in pulses, replace the '1' in the output matrix of Eq. (4.7-4) with the PG gain K_{dt} . Also, for a manipulation input/command from the computer in pulses/T, just multiply the first column of the input matrix in Eq. (4.7-3) by K_{da} . Thus, a computer command of $m(k)=204$ pulses, is equivalent to 1 volt at the current reference i_r because, for our 12-bit DAC,

$$204 \text{ (pulses)} * \left(\frac{10}{2047} \right) \left(\frac{\text{volts}}{\text{pulses}} \right) \approx 1 \text{ (volt)} \quad (4.7-5)$$

To obtain the required transfer functions from the above state equations, we can use matrix methods or we can Laplace transform the equations, assuming zero initial conditions, and work algebraically. Either way, the output (position) transform is

$$\Theta(s) = \frac{\frac{1}{J} [(s^2 + \frac{R+K_1K_c}{L}s + \frac{K_2K_c}{L})D(s) + (\frac{K_1K_t}{L}s + \frac{K_2K_t}{L})I_r(s)]}{s[s^3 + (\frac{F}{J} + \frac{R+K_1K_c}{L})s^2 + \frac{FR+FK_1K_c+JK_2K_c+K_bK_t}{JL}s + \frac{FK_2K_c}{JL}]} \quad (4.7-6)$$

The required $\Theta(s)/I_r(s)$ and $\Theta(s)/D(s)$ transfer functions can be obtained from Eq. (4.7-6) by setting $D(s)$ and $I_r(s)$ to zero, respectively. This leads to the position-to-current transfer function

$$\frac{\Theta(s)}{I_r(s)} = \frac{\frac{1}{J} (\frac{K_1K_t}{L}s + \frac{K_2K_t}{L})}{s[s^3 + (\frac{F}{J} + \frac{R+K_1K_c}{L})s^2 + \frac{FR+FK_1K_c+JK_2K_c+K_bK_t}{JL}s + \frac{FK_2K_c}{JL}]} \quad (4.7-7)$$

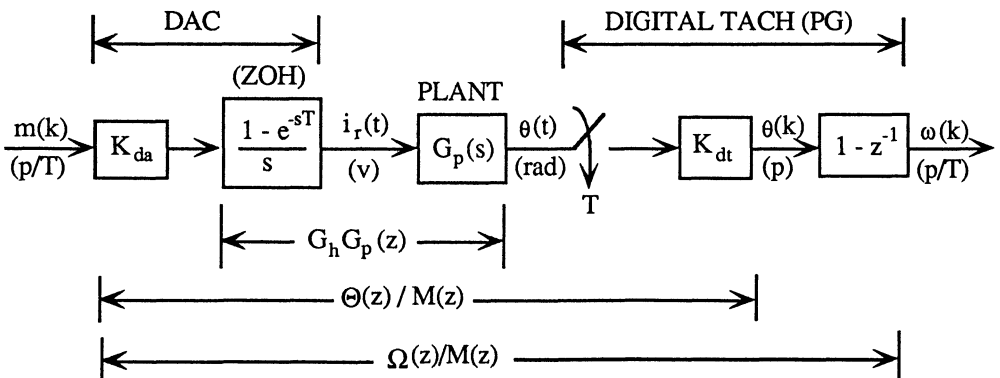


Fig. 4.7-2 Block diagram representation of an analog plant $G_p(s)$ and interfaces.

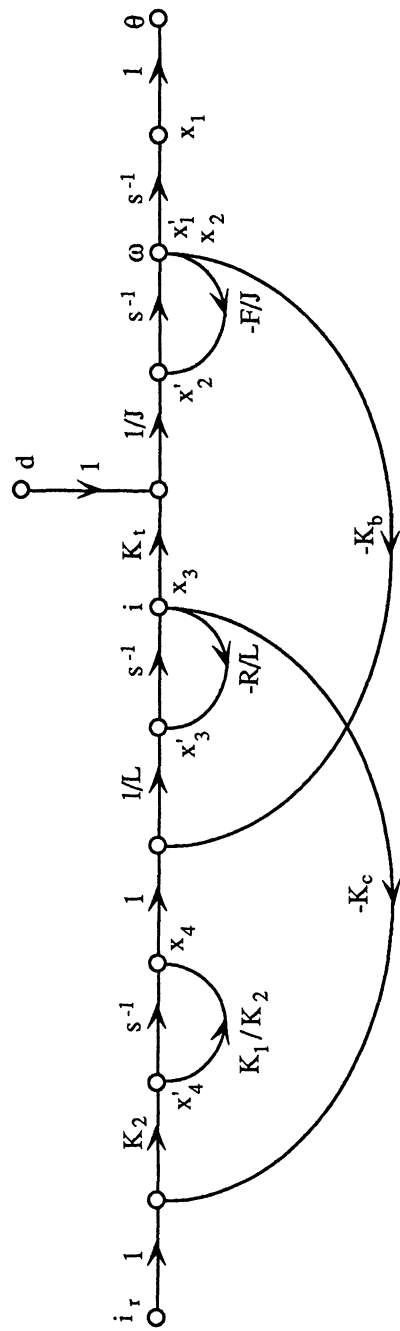


Fig. 4.7-3 State diagram of motor drive plant shown in Fig. 4.7-1.

and to the position-to-disturbance transfer function

$$\frac{\Theta(s)}{D(s)} = \frac{\frac{1}{J} (s^2 + \frac{R+K_1K_c}{L} s + \frac{K_2K_c}{L})}{s[s^3 + (\frac{F}{J} + \frac{R+K_1K_c}{L})s^2 + \frac{FR+FK_1K_c+JK_2K_c+K_bK_t}{JL} s + \frac{FK_2K_c}{JL}]} \quad (4.7-8)$$

Considering only the current input, substitute the given parameter values in the state and output equations given by Eq. (4.7-3) and Eq. (4.7-4). In addition, incorporate the K_{da} and K_{dt} gains in the input and output matrices, respectively. Since $K_{da} * K_1/L =$, $K_{da} * K_2 =$, and $K_{dt} = 8000/2\pi =$, the state and output equations are

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 17.1 & 0 \\ 0 & -375 & -269.7 & 500 \\ 0 & 0 & -40.9 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 7.914 \\ 2.638 \end{bmatrix} m(t) \quad (4.7-9)$$

and $\theta(t) = [1273.2 \quad 0 \quad 0 \quad 0] \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} \quad (4.7-10)$

These continuous equations can be "discretized" using the scheme developed in Section 4.5. Using $T=0.005$ sec, the resulting discrete state and output equations become

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ x_4(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 4.9045E-3 & 1.3591E-4 & 1.2738E-4 \\ 0 & 0.9490 & 0.4201 & 0.0680 \\ 0 & -0.9212 & 0.1240 & 1.2283 \\ 0 & 0.1219 & -0.1005 & 0.8375 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_4(k) \end{bmatrix} + \begin{bmatrix} 2.4660E-6 \\ 1.4116E-3 \\ 2.9923E-2 \\ 9.8140E-3 \end{bmatrix} m(k) \quad (4.7-11)$$

and

$$\theta(k) = [477.465 \quad 0 \quad 0 \quad 0] \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_4(k) \end{bmatrix} \quad (4.7-12)$$

If the input from the computer is $m(k)=204$ pulses (i.e., $i_r(t)=1$ volt), the above equations give the results shown in Table 4.7-1. Again, note that the output $\theta(k)$ is the position variable in pulses, $x_1(t)$ represents position in rad, $x_2(t)$ velocity in rad/sec, and $x_3(t)$ armature current in amps. To find velocity in pulses/T, multiply the velocity in rad/sec column by K_{dt} .

The position-to-manipulation transfer function is

$$\frac{\Theta(z)}{M(z)} = \frac{0.0031397z^3 + 0.009586z^2 - 0.002344z - 0.00105}{z^4 - 2.9105z^3 + 3.0806z^2 - 1.4298z + 0.25963} \quad (4.7-13)$$

which can be written in factored form as

$$\frac{\Theta(z)}{M(z)} = \frac{0.0031397z^3 + 0.009586z^2 - 0.002344z - 0.00105}{(z-1.01)(z-0.99)[(z-0.4554)^2 + 0.2288^2]} \quad (4.7-14)$$

The double-integrator nature of the positional system is evident by the double pole of its transfer function at $z=1$. This conclusion can also be drawn by substituting $F=0$ in the continuous transfer function of Eq. (4.7-7).

The velocity-to-manipulation transfer function, $\Omega(z)/M(z)$, can be obtained from Eq. (4.7-8) by first letting $F=0$ in the equation, and differentiating it, that is, multiplying it by s . Then, to account for the DAC and PG interfaces, we multiply the equation by K_{da} and K_{dt} . This leads to the continuous velocity-to-current transfer function

$$\frac{\Omega(s)}{I_r(s)} = \frac{\frac{1}{J} (K_{da}K_{dt}) \left(\frac{K_1K_t}{L} s + \frac{K_2K_t}{L} \right)}{s \left[s^2 + \frac{R+K_1K_c}{L} s + \frac{JK_2K_c + K_bK_t}{JL} \right]} \quad (4.7-15)$$

Substituting the given parameter values, we get

$$\frac{\Omega(s)}{I_r(s)} = \frac{172333.2(s+166.667)}{s(s^2+269.634s+26852.46)} \quad (4.7-16)$$

Using $T=0.005$ sec, we can discretize (approximate) the above function by obtaining its pulse transfer function. This results in the discrete velocity-to-current transfer function

$$\frac{\Omega(z)}{I_r(z)} = \frac{1.7976z^2 + 0.72197z - 0.65283}{z^3 - 1.9107z^2 + 1.1704z - 0.25972} \quad (4.7-17)$$

By factoring the denominator, the transfer function can be written in the form

$$\frac{\Omega(z)}{I_r(z)} = \frac{1.7976z^2 + 0.72197z - 0.65283}{(z-1)[(z-0.4553)^2 + 0.2289^2]} \quad (4.7-18)$$

An important parameter for the velocity and position transfer functions is their respective steady-state gain. Since the velocity response is given in pulses/ T , the **velocity plant gain**, call it K_ω , in pulses/sec can be obtained from Eq. (4.7-15) using the relation

$$K_\omega = \lim_{s \rightarrow 0} s \frac{\Omega(s)}{I_r(s)} T = \frac{172333.2 * 166.667 * 0.005}{26852.46} = 5.3482 \quad (4.7-19)$$

The value of K_ω represents the slope of the plain integrator (K_ω/s) to a unit-step input, after the transient effects of the poles and zero in Eq. (4.7-15) have died out.

Table 4.7-1 *Simulation results for motor drive system; input step is $m(k)=204$ pulses/T.*

TIME (sec)	$x_1(t)$ (θ in rad)	$x_2(t)$ (ω - rad/sec)	$x_3(t)$ (i - amps)	$x_4(t)$	OUTPUT (θ - pulses)	ω (pulses/T)
0.000	0	0	0	0	0.00	0.00
0.005	0.0005	0.288	6.1043	2.0021	0.64	0.64
0.010	0.0035	0.9538	9.055	3.1006	4.46	3.82
0.015	0.0103	1.7843	10.1568	3.8053	13.13	8.67
0.020	0.0214	2.6667	10.3939	4.3859	27.28	14.16
0.025	0.037	3.5535	10.3237	4.956	47.09	19.80
0.030	0.0569	4.4308	10.1982	5.5485	72.51	25.42
0.035	0.0813	5.2985	10.1022	6.1643	103.48	30.97
0.040	0.1099	6.1598	10.0474	6.7954	139.95	36.47
0.045	0.1429	7.0177	10.0223	7.4345	181.90	41.95
0.050	0.1801	7.8743	10.0138	8.0767	229.30	47.40
0.055	0.2216	8.7305	10.0125	8.7199	282.15	52.85
0.060	0.2674	9.5868	10.0136	9.363	340.46	58.30
0.065	0.3175	10.4431	10.0149	10.0059	404.21	63.76
0.070	0.3718	11.2996	10.0158	10.6485	473.42	69.21
0.075	0.4305	12.1561	10.0163	11.291	548.08	74.66
0.080	0.4934	13.0126	10.0165	11.9335	628.19	80.11
0.085	0.5606	13.8692	10.0165	12.5759	713.75	85.56
0.090	0.6321	14.7257	10.0165	13.2183	804.77	91.02
0.095	0.7079	15.5822	10.0165	13.8607	901.24	96.47
0.100	0.7879	16.4388	10.0165	14.5031	1003.17	101.92
0.105	0.8722	17.2953	10.0165	15.1455	1110.54	107.38
0.110	0.9609	18.1519	10.0165	15.7879	1223.37	112.83
0.115	1.0538	19.0084	10.0165	16.4303	1341.65	118.28
0.120	1.1509	19.865	10.0165	17.0727	1465.38	123.73
0.125	1.2524	20.7215	10.0165	17.7151	1594.57	129.19
0.130	1.3582	21.578	10.0165	18.3575	1729.21	134.64
0.135	1.4682	22.4346	10.0165	18.9999	1869.30	140.09
0.140	1.5825	23.2911	10.0165	19.6423	2014.85	145.55
0.145	1.7011	24.1477	10.0165	20.2847	2165.85	151.00
0.150	1.824	25.0042	10.0165	20.9271	2322.30	156.45
0.155	1.9511	25.8607	10.0165	21.5695	2484.20	161.90
0.160	2.0826	26.7173	10.0165	22.2119	2651.55	167.36
0.165	2.2183	27.5738	10.0165	22.8543	2824.36	172.81
0.170	2.3583	28.4304	10.0165	23.4967	3002.62	178.26
0.175	2.5026	29.2869	10.0165	24.1391	3186.34	183.71
0.180	2.6512	30.1434	10.0165	24.7816	3375.51	189.17
0.185	2.8041	31	10.0165	25.424	3570.12	194.62
0.190	2.9612	31.8565	10.0165	26.0664	3770.20	200.07
0.195	3.1226	32.7131	10.0165	26.7088	3975.72	205.52
0.200	3.2883	33.5696	10.0165	27.3512	4186.70	210.98

The positional response is given in pulses and the **positional plant gain**, call it K_θ , can be obtained from the velocity plant gain K_ω using the relation

$$K_\theta = \frac{K_\omega}{T} \quad (4.7-20)$$

For our case, this gives the value

$$K_\theta = \frac{5.3482}{0.005} = 1069.6 \quad (4.7-21)$$

Alternatively, K_θ can be obtained directly from Eq. (4.7-15), that is,

$$K_\theta = \lim_{s \rightarrow 0} s^2 \left[\frac{\Omega(s)}{s I_r(s)} \right] = \lim_{s \rightarrow 0} \frac{172333.2(s+166.667)}{(s^2+269.634s+26852.46)} = 1069.6 \quad (4.7-22)$$

The velocity and position plant gains can also be obtained from the respective discrete transfer functions. From Eq. (4.7-18), for instance, the velocity plant gain is

$$K_\omega = \lim_{z \rightarrow 1} \left[\frac{(z-1)}{T} \right] \left[\frac{\Omega(z)}{I_r(z)} \right] T = \lim_{z \rightarrow 1} (z-1) \left[\frac{\Omega(z)}{I_r(z)} \right] = 5.3474 \quad (4.7-23)$$

Using the discrete position plant transfer function given by Eq. (4.7-14), on the other hand, the positional plant gain is obtained as

$$K_\theta = \lim_{z \rightarrow 1} \left[\frac{(z-1)^2}{T^2} \right] \left[\frac{\Theta(z)}{I_r(z)} \right] \approx \frac{0.0093}{0.3489*0.005*0.005} = 1066.2 \quad (4.7-24)$$

The significance of Eq. (4.7-20) is its simplicity and practicality. The means that given a velocity step response, analog or digital, the velocity plant gain K_ω is the slope of the resulting ramp function. This step response can be readily obtained and recorded in the laboratory. To obtain the positional plant gain K_θ , simply divide K_ω by T or, equivalently, multiply K_ω by the sampling frequency ($1/T$).

The motor drive plant analyzed above is used in Chapter 7 as one of the plants to be controlled by a digital computer. Reduced-order (simplified) models for the motor drive plant using a systematic method is the subject of the Case Study in Chapter 8.

PROBLEMS

4-1. A dynamic system is represented by the differential equation

$$c''(t) + 3c'(t) + 2c(t) = r(t), \text{ with } c(0)=1, \text{ and } c'(0)=0$$

represent the system in state space form

4-2. Determine the unit-step response of the system in Problem 4-1.

4-3. The transfer function of a system is given by

$$G(s) = \frac{C(s)}{R(s)} = \frac{100}{(s+5)(s+10)}$$

Determine state space representations using (a) direct , (b) cascade, and (c) parallel decomposition.

- 4 - 4.** A discrete-time system is represented by the following dynamic equations, that is, state and output equations:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.90167 & 0.03449 \\ -3.449 & 0.41881 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.09833 \\ 3.44900 \end{bmatrix} r(k)$$

and $c(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$

where $r(k)$ and $c(k)$ represent the system input and output.

- (a) Determine the corresponding discrete transfer function $C(z)/R(z)$
- (b) Determine and plot the unit-step response of the system.

- 4 - 5.** A discrete-time system is described by the transfer function

$$\frac{C(z)}{R(z)} = \frac{(0.09833z + 0.07778)}{(z^2 - 1.320z + 0.49659)}$$

Determine a discrete state-space model using direct decomposition. Express the dynamic equations in vector-matrix form.

- 4 - 6.** A continuous system is described by the following state and output equations

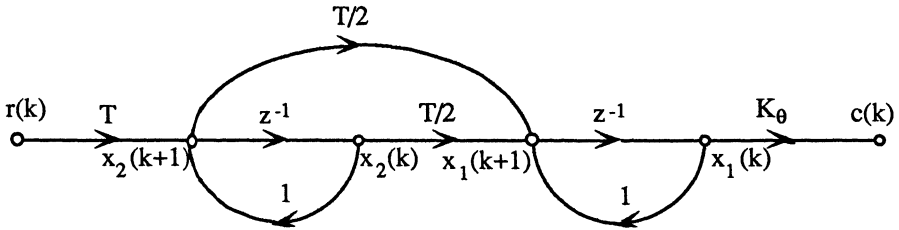
$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -100 & -14 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(t)$$

and $c(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$

where $r(t)$ and $c(t)$ represent the system input and output.

Determine a discrete state-space model, that is, a set of first-order difference and output equations by approximating the above equations using the forward-difference integration/differentiation scheme.

- 4 - 7.** A system is represented by the following state diagram. Determine the corresponding state space equations. Then, find the transfer function model of the system. If $K_0=1070$, and $T=0.005$ sec, determine and plot the unit-step response of the system.



- 4-8. The transfer function of a discrete system is given by

$$G(z) = \frac{C(z)}{R(z)} = \frac{z^2}{(z^2 - 0.9z + 0.2)}$$

Determine state space representations using (a) direct , (b) cascade and (c) parallel decomposition.

- 4-9. A linear recursion equation which characterizes the input-output relationship of a digital computer is given by

$$y(k) = \sum_{i=0}^n a_i x(k-i) - \sum_{i=1}^n b_i y(k-i) .$$

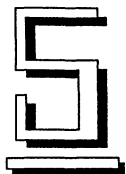
Determine the corresponding digital computer transfer function $Y(z)/X(z)$.

- 4-10. Determine the observability and controllability of the analog plant described by the transfer function

$$G(s) = \frac{K_0}{s^2}$$

- 4-11. Determine the observability and controllability of the discrete system given in Problem 4-7.

- 4-12. Develop a computer program that implements the simulation scheme suggested by Eq. (4.5-1) and (4.5-2). That is, write a computer program that determines the discrete state equations from the continuous ones. Then, evaluate the resulting difference equations to simulate any given system.



DIGITAL SIGNAL PROCESSING

Design and Implementation of Digital Filters

5.1 INTRODUCTION

In many fields of science and engineering, there is often a requirement to transform a signal, continuous or discrete, into a form which is suitable for a particular application. In a signal processing application, for instance, a given signal may contain undesirable disturbances (noise) mixed in with desirable information. It is then necessary to process the signal in order to pass (extract) only the useful information and block (reject) the noise. The system or device that accomplishes this is called a **filter**. A radio receiver, for instance, uses filters in order to pass the band of frequencies transmitted by the radio station we want to hear and to reject noise and frequencies transmitted by other radio stations. A control system, for another example, uses filters to modify the dynamics of a given process and thus make the performance of the overall system satisfactory.

In the past, most of the transformations implied by filters and compensators have been accomplished by analog hardware. However, the increased speed and decreased size and cost of modern microprocessors and microcomputers have permitted the economical use of digital processing of signals to implement filtering and control algorithms in real time.

The software implementation offers several advantages. First, it offers flexibility, because the characteristics of the transformation algorithm can be easily changed by simply reading in a new set of parameters. Second, a software implementation results in a more reliable system, because there are no physical components to age or deteriorate with temperature variations. This in turn implies consistency in system performance and independence from bulky component problems, usually associated with low-frequency analog realizations. Finally, a greater degree of accuracy can be attained with programming the filter or control algorithm, because the parameters are controlled by the designer who can set them to obtain an efficient and cost-effective system that meets problem specifications.

5.2 DIGITAL FILTER CONCEPTS

A digital filter is a discrete system which operates on a discrete signal (sequence of numbers) input to produce a discrete signal output according to some computational process or algorithm. By use of ADC and DAC interfaces, the input to a digital filter can be a uniformly sampled signal, thus, allowing digital filters to be used for processing analog signals, too.

Like any linear, time-invariant, discrete system, a digital filter can be represented by the familiar n th-order difference equation

$$y(k) = \sum_{i=0}^n a_i x(k-i) - \sum_{i=1}^n b_i y(k-i) \quad (5.2-1)$$

or, equivalently, by the transfer function

$$G(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{i=0}^n a_i z^{-i}}{1 + \sum_{i=1}^n b_i z^{-i}} \quad (5.2-2)$$

If state space techniques are employed, the filter can be also described by the dynamic equations

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}\mathbf{u}(k) \quad (5.2-3)$$

and

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) \quad (5.2-4)$$

Thus, in the case of a multiple-input and/or multiple-output system, the matrix of transfer functions, $G(z)$, can be obtained by substituting the above coefficient matrices in Eq. (4.3-30).

Thus far in this book, the main emphasis has been on system analysis. That is, given a mathematical description of a system, we concentrated on solving the equations to determine and evaluate its performance due to certain types of inputs. In this chapter, however, the main objective is the design of digital filters. In other words, given a set of problem specifications, the goal is to determine the difference equation which satisfies these specifications, at least approximately. The digital filter is then realized using appropriate structures.

In many applications, a digital filter, just like an analog filter, may be designed to perform standard frequency operations such as low-pass filtering, high-pass filtering, band-pass filtering, and band-reject filtering. In other cases, a digital filter may perform functions such as differentiation, prediction, integration, smoothing, or any combination of these operations.

A **low-pass filter** is a system which passes sinusoidal inputs whose frequencies, ω , fall in the range $0 \leq \omega \leq \omega_c$ and rejects sinusoidal inputs whose frequencies are greater than ω_c , where ω_c is the cutoff frequency.

A **high-pass filter** is a system which rejects sinusoidal inputs whose frequencies are smaller than ω_c , and passes sinusoidal inputs whose frequencies are greater than ω_c .

A **band-reject filter** is a system which rejects sinusoidal inputs whose

frequencies lie in the range $\omega_1 \leq \omega \leq \omega_2$, and passes sinusoidal inputs whose frequencies fall outside this range.

A **band-pass filter** is a system which passes sinusoidal inputs whose frequencies lie in the range $\omega_1 \leq \omega \leq \omega_2$, and rejects sinusoidal inputs whose frequencies fall outside this range.

The ideal frequency amplitude or magnitude response characteristics of the previous digital filters are shown in Fig. 5.2-1. In this figure, ω_s represents the **sampling frequency** (rad/sec), and ω_o denotes the **folding frequency** which is equal to $\omega_s/2$.

Note that in the frequency range from $\omega=0$ to the folding frequency, ω_o , the ideal amplitude response characteristics of the digital filters are identical to the ideal amplitude response characteristics of the equivalent analog filters. However, like all discrete systems, the frequency response characteristics of the digital filters are periodic, that is, they repeat every $\omega_s/2$ rad/sec.

Digital filters are also classified according to the duration of their impulse response, and the type of structure or realization used for actual implementation. A digital filter whose impulse response has an infinite number of samples is known as an **Infinite Impulse Response (IIR) digital filter**. On the other hand, a digital filter whose impulse response is of finite duration is called a **Finite Impulse Response (FIR) filter**. Generally, IIR filters are implemented using structures with feedback while FIR filter structures do not use feedback.

If at least one b_i and one a_i coefficients are nonzero in Eq. (5.2-1), the digital filter is said to be of the **recursive type**, that is, the present value of the output depends both on the present and previous values of the input as well as on the previous values of the output. An IIR digital filter is generally implemented by a recursive realization. If all the b_i coefficients are zero, then the digital filter is said to be of the **nonrecursive type**, that is, the output depends only on the present and past values of the input.

A FIR digital filter is usually implemented by a nonrecursive structure and can be represented by the equation

$$y(k) = \sum_{i=0}^n a_i x(k-i) \quad (5.2-5)$$

FIR filters operate on a predetermined number of input samples, considered to be a **data window**. Note that the data window moves down the time axis at the sampling rate, and it takes $(n+1)$ sampling times for all of the points to be aware of the change. Thus, the digital filter has a transient response which lasts $(n+1)$ sampling periods. The difference equation implied by the backward derivative approximation operator

$$y(k) = \frac{x(k) - x(k-1)}{T} \quad (5.2-6)$$

is an example of a FIR digital filter. The two-point signal averager

$$y(k) = \frac{x(k) + x(k-1)}{2} \quad (5.2-7)$$

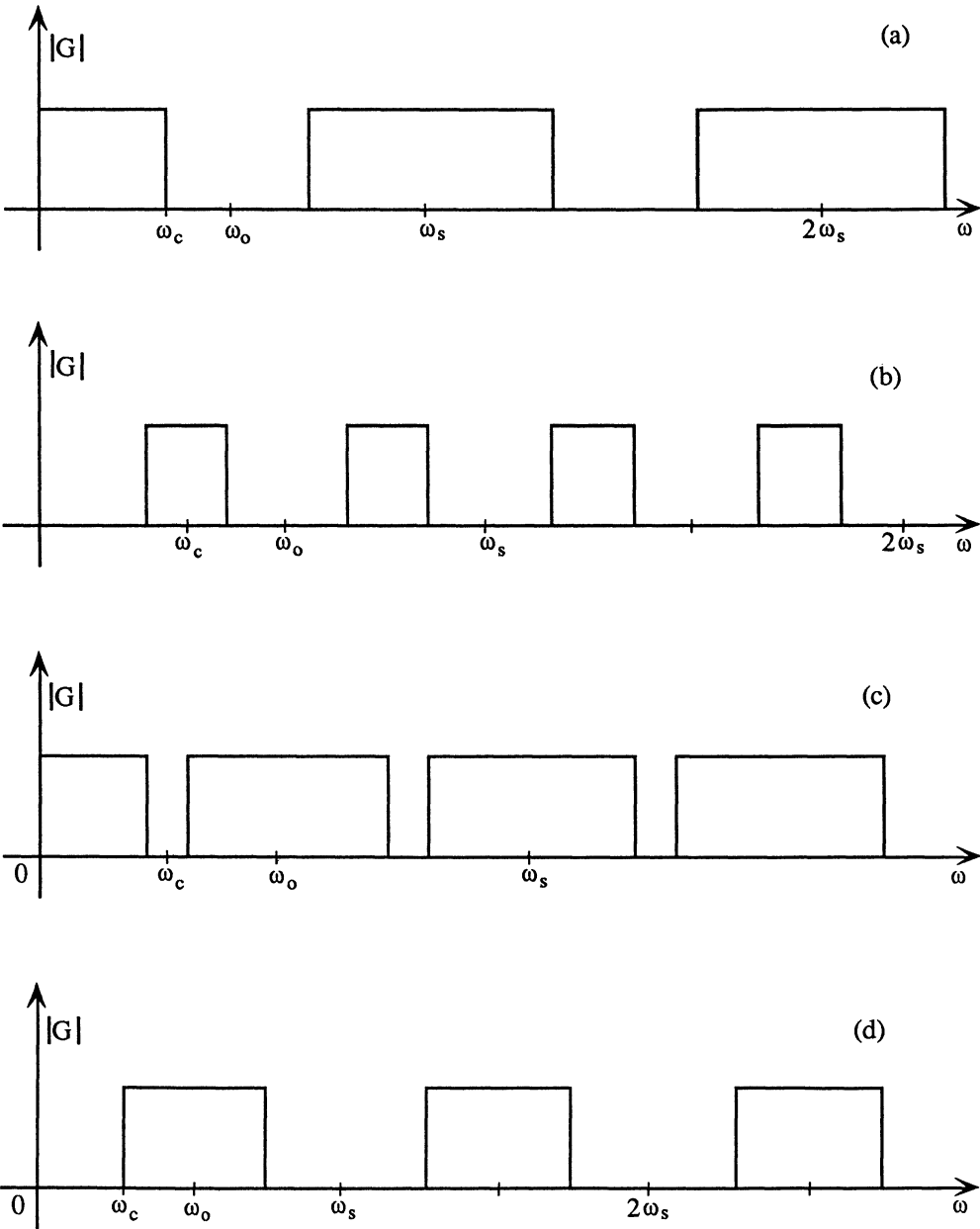


Fig. 5.2-1 Ideal amplitude/magnitude response characteristics for (a) low-pass, (b) band-pass, (c) band-reject, and (d) high-pass digital filters.

is another example. Any of the numerical integration rules developed in Chapter 3 would be an example of an IIR digital filter. For instance, the trapezoidal integration rule

$$y(k) = y(k-1) + \frac{T}{2} [x(k) + x(k-1)] \quad (5.2-8)$$

is an IIR digital filter.

The design of digital filters is generally carried out by one of two methods. The first method consists of designing an **analog prototype** filter that meets the problem requirements. Once this is accomplished, the analog filter transfer function is converted to a digital filter transfer function by some approximation technique. The main advantage of this approach is that the design of analog filters is well established and, thus, analog filter transfer functions suitable for many applications are readily available. Any limitations are generally related to the choice of the conversion technique and/or of the sampling frequency.

The second method is to design the digital filter directly in the discrete frequency (or discrete time) domain using appropriate techniques. The advantage of this approach is directness, because the performance of the digital filter depends on the exact placement of its discrete poles and zeros. The disadvantage of the method is the difficulty encountered by a designer who wishes to visualize the exact location of the z-plane poles and zeros in order to achieve required frequency response characteristics.

In this chapter, the design of digital IIR filters using the first method is emphasized. That is, an analog filter with desirable characteristics is selected and approximated with a digital filter that has similar characteristics. An example of designing a digital FIR filter using the method of least-squares approximation is also included.

The filter design concepts presented in this chapter are also relevant in the design of digital controllers, which is the subject of Chapter 7.

5.3 ANALOG FILTER CONCEPTS

Consider the block diagram representation of an analog filter shown in Fig. 5.3-1. In this diagram, the input consists of some useful information $x(t)$, and some disturbance or noise $d(t)$. The filter output is $y(t)$. By definition, the objective of the filter is to eliminate the undesirable noise while preserving the useful signal.

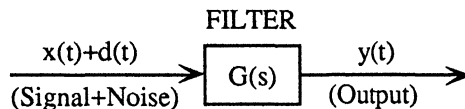


Fig. 5.3-1 Block diagram of an analog filter.

Assuming an ideal filter, the operation can be expressed mathematically by

$$y(t) = Kx(t-T) \quad (5.3-1)$$

where the constant K is an amplitude modifier and T represents a system delay. Taking the Laplace transform on both sides of Eq. (5.3-1) gives

$$Y(s) = Ke^{-sT}X(s) \quad (5.3-2)$$

The filter transfer function then is

$$G(s) = \frac{Y(s)}{X(s)} = Ke^{-sT} \quad (5.3-3)$$

Expressing $G(s)$ in terms of magnitude and phase, we obtain

$$G(j\omega) = Ke^{-j\omega T} = K \angle -\omega T \quad (5.3-4)$$

Eq. (5.3-4) implies that in the passband the amplitude response of the ideal filter is constant, and the phase response is a linear function of frequency.

It is, of course, impossible to build an ideal filter, because it would have to be of infinite order and, thus, require an infinite number of components. Fortunately, the ideal filter can be approximated by practical-order filters that approach the ideal characteristics sufficiently close to satisfy the requirements of many real world applications.

There are many types of analog filters available. There are also many sophisticated techniques and computer programs for carrying out complex filter designs on digital computers. In our discussions, we will emphasize the Butterworth, ITAE, Binomial, and Bessel types of filters. The reason for this is that these filters have attractive frequency as well as transient response characteristics, which make them desirable in both signal processing as well as digital control applications.

The transfer function form for the Butterworth, ITAE, Binomial, and Bessel filters is given by

$$G(s) = \frac{Y(s)}{X(s)} = \frac{b_n}{s^n + b_1 s^{n-1} + b_2 s^{n-2} + \dots + b_{n-1} s + b_n} \quad (5.3-5)$$

where the coefficients $b_1, b_2, \dots, b_n$ assume different values. Eq. (5.3-5) is often referred to as a **zero-displacement error (ZDE) system**, because it can track step inputs with zero steady-state error.

5.3-1 Butterworth Filter Forms

The magnitude squared function of the low-pass **Butterworth**, also known as the **maximally-flat** filter is given by

$$|G(j\omega)|^2 = \frac{1}{1 + (\omega/\omega_c)^{2n}} \quad (5.3-6)$$

where n represents the order of the transfer function, and ω_c is the cutoff frequency. As the value of n increases, the filter structure becomes more complicated, but the cutoff of signals having frequencies above ω_c becomes sharper. Equivalently, the higher the order of the filter, the closer we get to the frequency response of the ideal low-pass filter.

The poles of the Butterworth filter can be obtained from the characteristic equation

$$1 + (s/j\omega_c)^{2n} = 0 \quad (5.3-7)$$

which can be written as

$$(s/j\omega_c)^{2n} = -1 = e^{j\pi(1+2k)} \quad (5.3-8)$$

Solving the last equation for s , we obtain

$$s = j\omega_c e^{j\pi(1+2k)/2n} = \omega_c e^{j\pi[1+(1+2k)/n]/2} \quad (5.3-9)$$

where $k = 0, 1, \dots, (2n-1)$. Eq. (5.3-9) represents a circle of radius ω_c and it implies that the poles of the filter transfer function lie on this circle, equally-spaced. Because of stability considerations, only the left-hand side poles are used, while the right-hand side poles are discarded.

Example 5.3-1

Determine the poles and the transfer function of the third-order Butterworth filter, assuming $\omega_c=1$ rad/sec.

With $n=3$ in Eq. (5.3-8), we can determine the left-side poles by letting k take the values 0, 1, and 2. Thus, since $\omega_c=1$, the respective poles are

$$s_1 = e^{j2\pi/3} = -0.5 + j0.866$$

$$s_2 = -1$$

and

$$s_3 = e^{j4\pi/3} = -0.5 - j0.866$$

The filter transfer function then becomes

$$G(s) = \frac{1}{(s+1)[(s+0.5)^2 + 0.866^2]} = \frac{1}{s^3 + 2s^2 + 2s + 1}$$

The unit-step and frequency-amplitude responses are shown in Fig. 5.3-2a, and Fig. 5.3-2b, respectively. Note that the time response is plotted versus $\omega_c t$, and the frequency response is plotted versus ω/ω_c , where ω_c is the cutoff frequency.

If ω_c were not equal to 1 in the above example, the third-order Butterworth filter transfer function would be of the form

$$G(s) = \frac{\omega_c^3}{s^3 + 2\omega_c s^2 + 2\omega_c^2 s + \omega_c^3} \quad (5.3-10)$$

Comparing the two transfer functions, we see that the second can be derived from the first by simply replacing s by s/ω_c . The characteristic polynomials of the first five Butterworth filters are shown in Table 5.3-1. Higher-order filters can be derived using Eq. (5.3-9).

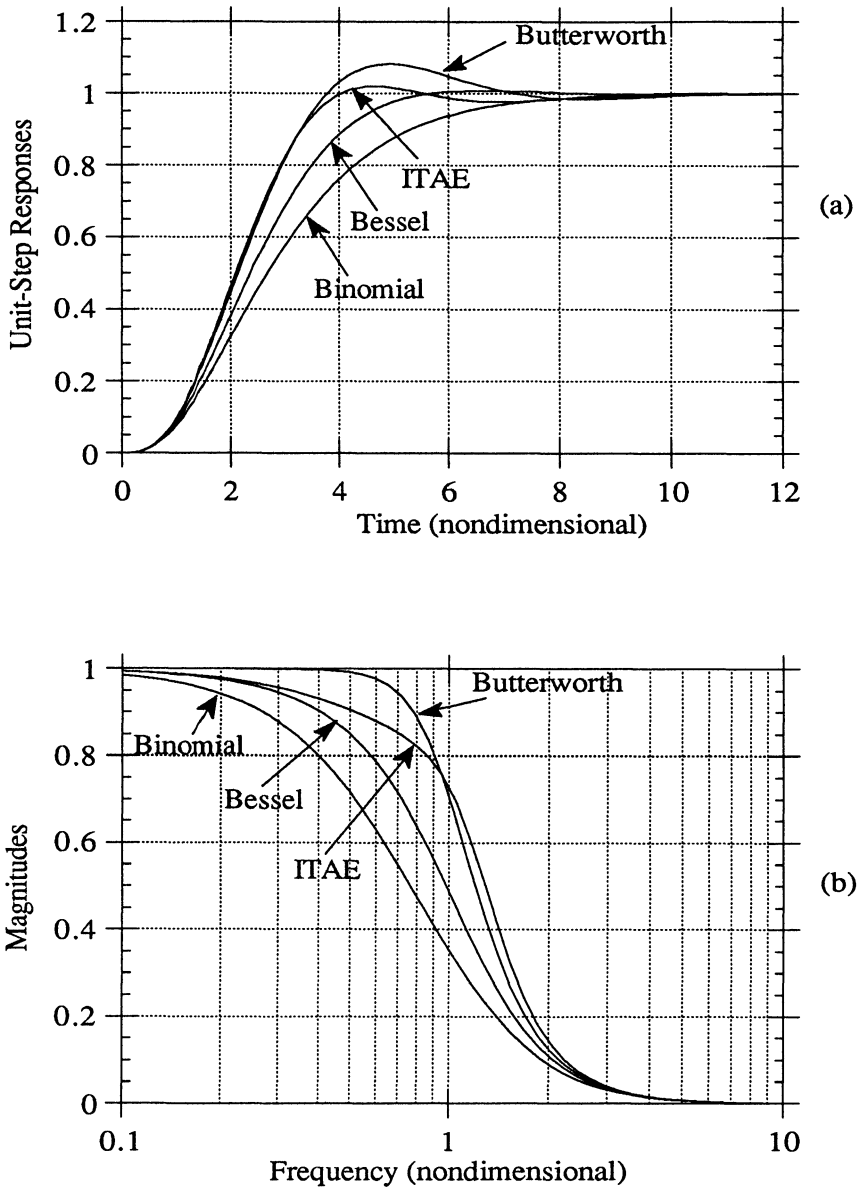


Fig. 5.3-2 Unit-step responses (a), and frequency magnitude responses (b) of 3rd-order Butterworth, ITAE, Bessel, and Binomial low-pass filters.

Table 5.3-1 *Low-pass Butterworth filter forms*

n	Denominators
1	$s + 1$
2	$s^2 + \sqrt{2}s + 1$
3	$s^3 + 2s^2 + 2s + 1$
4	$s^4 + 2.613s^3 + 3.414s^2 + 2.613s + 1$
5	$s^5 + 3.24s^4 + 5.24s^3 + 5.24s^2 + 3.24s + 1$

5.3-2 Binomial Filter Forms

A **Binomial** filter has all its poles located at the $-\omega_c$ point. Thus, with all real poles, the transient response of the filter exhibits no overshoot or ringing but is relatively slow. For instance, the third-order Binomial filter is given by

$$G(s) = \frac{\omega_c^3}{(s + \omega_c)^3} = \frac{\omega_c^3}{s^3 + 3\omega_c s^2 + 3\omega_c^2 s + \omega_c^3} \quad (5.3-11)$$

Normalized time and frequency responses for the third-order Binomial filter are shown in Fig. 5.3-2a and Fig. 5.3-2b. The characteristic polynomials of the first five Binomial filters are shown in Table 5.3-2.

Table 5.3-2 *Low-pass Binomial filter forms*

n	Denominators
1	$s + 1$
2	$s^2 + 2s + 1$
3	$s^3 + 3s^2 + 3s + 1$
4	$s^4 + 4s^3 + 6s^2 + 4s + 1$
5	$s^5 + 5s^4 + 10s^3 + 10s^2 + 5s + 1$

5.3-3 ITAE and Bessel Filter Forms

To evaluate system performance, a performance index is usually chosen and measured. The system parameters are then adjusted so that the index is minimized. The coefficients, that will minimize the **Integral of Time-multiplied Absolute-value Error** (ITAE) performance criterion for a step input for the transfer function of Eq. (5.3-5), have been determined by Graham and Lathrop. Normalized time and frequency responses for the ITAE filter are shown in Fig. 5.3-2a and Fig. 5.3-2b, respectively. The characteristic polynomials of the first five ITAE filters are shown in Table 5.3-3.

Another type of low-pass filter with response characteristics related to the previous three types is the Bessel filter. Its unit-step and frequency responses are shown in Fig. 5.3-2a and Fig. 5.3-2b, respectively. The Bessel filter coefficients are shown in Table 5.3-4.

Table 5.3-3 Low-pass ITAE filter forms

n	Denominators
1	$s + 1$
2	$s^2 + \sqrt{2}s + 1$
3	$s^3 + 1.75s^2 + 2.15s + 1$
4	$s^4 + 2.1s^3 + 3.4s^2 + 2.7s + 1$
5	$s^5 + 2.8s^4 + 5s^3 + 5.5s^2 + 3.4s + 1$

Table 5.3-4 Low-pass Bessel filter forms

n	Denominators
1	$s + 1$
2	$s^2 + \sqrt{3}s + 1$
3	$s^3 + 2.433s^2 + 2.4661s + 1$
4	$s^4 + 3.124s^3 + 4.3919s^2 + 3.2014s + 1$
5	$s^5 + 3.8108s^4 + 6.777s^3 + 6.8867s^2 + 3.9365s + 1$

Comparing the four unit-step responses in Fig. 5.3-2a, we see that the Binomial response is relatively slow, the Butterworth response is faster than the Binomial but oscillatory, and the ITAE and Bessel responses are faster than the Binomial but less oscillatory than the Butterworth. The same conclusions can be deduced from the corresponding frequency responses shown in Fig. 5.3-2b. Observe the relative "flatness" of the Butterworth frequency response, which is the reason this filter type is also referred to as maximally flat. On the other extreme, note the higher attenuation associated with the Binomial response, even at relatively low frequencies.

5.3-4 Other Types of Analog Filters

The filters considered so far are of the low-pass type. It is relatively easy to design other filter types by using appropriate transformation relations.

High-Pass Filters. To design a high-pass Butterworth filter from a low-pass filter, the transformation relation is

$$G_{hp}(s) = G(s) \Big|_{s=1/s} \quad (5.3-12)$$

where $G(s)$ and $G_{hp}(s)$ are the low-pass and high-pass transfer functions, respectively. For example, the first-order, high-pass filter that results from the above relation is

$$G_{hp}(s) = \frac{1}{s+1} \Big|_{s=1/s} = \frac{s}{s+1} \quad (5.3-13)$$

As another example, the third-order, high-pass Butterworth filter is

$$G_{hp}(s) = \frac{1}{s^3+2s^2+2s+1} \Big|_{s=1/s} = \frac{s^3}{s^3+2s^2+2s+1} \quad (5.3-14)$$

Thus, for a cutoff frequency of 60Hz, or 377 rad/sec, the high-pass filter transfer function is determined by replacing s by $s/377$ in the last equation. The result is the high-pass filter transfer function

$$G_{hp}(s) = \frac{s^3}{s^3 + 754s^2 + 284258s + 53582633} \quad (5.3-15)$$

Band-Pass Filters. An analog band-pass filter can be obtained from a low-pass filter using the relation

$$G_{bp}(s) = G(s) \Big|_{s=(s^2+\omega_c^2)/Bs} \quad (5.3-16)$$

where ω_c is now called the center frequency (in a logarithmic sense) of the band-pass filter and B is the bandwidth. That is, if ω_1 and ω_2 are the respective lower and upper frequencies, then, from

$$\ln \omega_c = \frac{\ln \omega_1 + \ln \omega_2}{2} \quad (5.3-17)$$

$$\omega_c = \sqrt{\omega_1 \omega_2} \quad (5.3-18)$$

$$\text{and} \quad B = \omega_2 - \omega_1 \quad (5.3-19)$$

For example, we can design a second-order, band-pass filter from a first-order, low-pass filter using the relation

$$G_{bp}(s) = \frac{1}{s+1} \Big|_{s=(s^2+\omega_c^2)/Bs} = \frac{Bs}{s^2 + Bs + \omega_c^2} \quad (5.3-20)$$

Band-Reject Filters. Band-reject or **notch** filters can be designed from high-pass filters from the relation

$$G_{br}(s) = G_{hp}(s) \Big|_{s=(s^2+\omega_c^2)/Bs} \quad (5.3-21)$$

where ω_c is the center (or notch) frequency of the stop band and B is the bandwidth. For example, consider the first-order, high-pass filter given by Eq. (5.3-13). The corresponding band-reject filter is given by

$$G_{br}(s) = \frac{s}{s+1} \Big|_{s=(s^2+\omega_c^2)/Bs} = \frac{s^2 + \omega_c^2}{s^2 + Bs + \omega_c^2} \quad (5.3-22)$$

This last transfer function looks similar to the transfer function of the *twin-T* or notch filter network shown in Fig. 2.6-3. The network transfer function is given by Eq. (2.6-13), which after substituting ω_c for $1/CR$ and rearranging we get

$$G(s) = \frac{s^2 + \omega_c^2}{s^2 + 4\omega_c s + \omega_c^2} \quad (5.3-23)$$

The difference between the two band-reject transfer functions is the bandwidth parameter which can be adjusted in Eq. (5.3-22), but not in Eq. (5.3-23).

5.4 DESIGN OF IIR DIGITAL FILTERS

Compared to FIR filters, IIR filters tend to require a smaller number of terms to obtain a relatively sharp cutoff but may exhibit poor phase characteristics. For the design of IIR digital filters, as was stated earlier, the starting point is the analog filter transfer function $G(s)$. The problem is to determine a digital filter transfer function, $G(z)$, which approximates the performance of the analog filter.

Before introducing specific methods for converting $G(s)$ into $G(z)$, we should emphasize the importance of selecting the sampling time T . The choice of T depends on the frequency content of the input signal and, in addition, T must be small enough so that the performance of the digital filter resembles that of its analog counterpart. Care must be exercised, however, because the excessive computation implied by a very small T , may lead to inefficient and expensive solutions. At the other extreme, if T is too large, the solution obtained may be completely unacceptable.

There are several techniques available for approximating an analog filter transfer function by a digital filter transfer function that satisfies the problem requirements. A particular technique is usually selected on the basis of the desired impulse response, step response, and frequency response characteristics.

5.4-1 Impulse Invariance Method

The design criterion for this method requires the impulse response of the digital filter $G(z)$ to be the same as the impulse response of the analog prototype filter $G(s)$, at the sampling instants kT , for $k=0, 1, 2, \dots$. If $x(t)$ is the input and $y(t)$ the output of the analog filter, we have

$$Y(s) = G(s)X(s) = G(s) \quad (5.4-1)$$

because $X(s)=1$ for a unit-impulse input. Then, the transfer function of the desired digital filter $G(z)$ is given by

$$G(z) = T\mathcal{Z}[G(s)] \quad (5.4-2)$$

where the quantity in the brackets, $G(s)$, represents the given analog filter transfer function. The impulse invariant method is illustrated next by an example.

Example 5.4-1

Design a low-pass digital filter from a second-order Butterworth analog filter with a cutoff frequency of 50 rad/sec. Assume a sampling frequency of 60 Hz and use the impulse invariance method.

The second-order analog reference filter is given by

$$G(s) = \frac{\omega_c^2}{s^2 + \sqrt{2}\omega_c s + \omega_c^2} = \frac{50^2}{s^2 + \sqrt{2}50s + 50^2} = \frac{2500}{s^2 + 70.7s + 2500}$$

Since the poles of the system are complex, we can write the above transfer function in the form

$$G(s) = \frac{2500}{(s+35.35)^2+35.35^2}$$

from which the digital filter transfer function becomes (refer to Table 3.5-1)

$$G(z) = \frac{T*70.7*0.308z}{z^2 - 0.922z + 0.308} = \frac{0.363z}{z^2 - 0.922z + 0.308}$$

5.4-2 Step Invariance or Zero-Order-Hold (ZOH) Method

The design criterion using the step invariance method requires that the step response of the digital filter matches, as closely as possible, the step response of the corresponding analog filter at the sampling instants. The step response of the analog filter is given by

$$y(t) = \mathcal{F}^{-1}\left[\frac{G(s)}{s}\right] \quad (5.4-3)$$

which when $\mathcal{Z}$ transformed becomes

$$Y(z) = \mathcal{Z}\{\mathcal{F}^{-1}\left[\frac{G(s)}{s}\right]\} \quad (5.4-4)$$

On the other hand, the output transform of the digital filter due to a unit-step input is given by

$$Y(z) = G(z) \frac{z}{z-1} \quad (5.4-5)$$

Equating Eq. (5.4-4) with Eq. (5.4-5) and solving for the digital filter transfer function $G(z)$, we obtain the desired result

$$G(z) = (1-z^{-1})\mathcal{Z}\{\mathcal{F}^{-1}\left[\frac{G(s)}{s}\right]\} = (1-z^{-1})\mathcal{Z}\left[\frac{G(s)}{s}\right] \quad (5.4-6)$$

The last equation is the familiar pulse transfer function relation for the continuous transfer function $G(s)$. Thus, in order to design a digital filter from an analog filter prototype we simply assume that the analog filter is preceded by a ZOH (or DAC), and calculate the $\mathcal{Z}$ transform of the combination.

This approach was followed numerous times in Chapter 3. In Chapter 4, we showed that the pulse transfer function can be obtained when simulating a continuous system by state space techniques. The step invariance or Zero-Order-Hold method is illustrated next by an example.

Example 5.4-2

Design the digital filter of Example 5.4-1 using the ZOH method.

The digital filter transfer function is calculated from

$$G(z) = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{2500}{s^2+70.7s+2500}\right] = (1-z^{-1})\mathcal{Z}\left[\frac{2500}{s(s^2+70.7s+2500)}\right]$$

or
$$G(z) = \frac{0.2306z + 0.1549}{z^2 - 0.9222z + 0.3077}$$

Observe that the gain of the digital filter matches the gain of its analog counterpart, in this case unity. Thus, no steady-state gain adjustments are required when the ZOH method of design is used.

5.4-3 Pole-Zero Matching Method

The pole-zero matching method consists basically of mapping the poles and zeros of $G(s)$ to poles and zeros of $G(z)$ using the transformation

$$z = e^{sT} \quad (5.4-7)$$

and then adjusting the gain of the digital filter to match that of the analog filter. The design technique can be summarized as follows:

1. Map all poles of $G(s)$ to the z plane according to Eq. (5.4-7).
2. Map all finite zeros of $G(s)$ to the z plane according to Eq. (5.4-7).
3. Map all zeros of $G(s)$ at infinity to the point $z = -1$. The implication of this rule is to add enough zeros at $z = -1$ so that the number of poles equals the number of zeros.
4. Adjust the gain of the digital filter so that it matches the gain of the analog filter at some critical frequency, usually $s=0$, or $z=1$.

Note that if a delay is desirable in $G(z)$, one zero of $G(s)$ at infinity is mapped into a zero of $G(z)$ at infinity. The pole-zero matching method is illustrated next by an example.

Example 5.4-3

Design the digital filter of Example 5.4-1 using the pole-zero matching method.

The complex pair of poles of the analog filter have already been mapped in the previous examples. The filter has no finite zeros, but it has two zeros at infinity. Therefore, the digital filter form is

$$G(z) = \frac{K(z+1)^2}{z^2 - 0.9222z + 0.308}$$

Observe that the gain of the analog filter transfer function is unity. This implies that

$$\left. \frac{K(z+1)^2}{z^2 - 0.9222z + 0.308} \right|_{z=1} = 1$$

Solving for K , we obtain $K=0.0965$. The digital filter transfer function then becomes

$$G(z) = \frac{0.0965(z+1)^2}{z^2-0.922z+0.308} = \frac{0.0965z^2+0.193z+0.0965}{z^2-0.922z+0.308}$$

5.4-4 Bilinear Transformation Method

In designing digital filters, the main objective so far has been to approximate the analog filter transfer function by a digital filter transfer function with similar characteristics. Alternatively, the analog filter can be represented by its differential equation model which can then be approximated by a difference equation, using any of the numerical integration techniques presented in Section 3.10. The resultant difference equation is then programmed on a digital computer in order to carry out the digital filter function.

One of the more popular methods for designing digital filters is the **bilinear transformation method**, named so because it uses the bilinear transformation relation introduced in Section 3.8-1. The bilinear transformation or mapping is defined by

$$z = \frac{1+s}{1-s} \quad (5.4-8)$$

Solving for s

$$s = \frac{z-1}{z+1} \quad (5.4-9)$$

Let us assume that the analog filter frequency is denoted by ω_c , and the digital frequency by ω_d . Then, substituting $s=j\omega_d$ in Eq. (5.4-9), we obtain

$$j\omega_c = \frac{e^{j\omega_d T}-1}{e^{j\omega_d T}+1} = \frac{e^{j\omega_d T/2}(e^{j\omega_d T/2}-e^{-j\omega_d T/2})}{e^{j\omega_d T/2}(e^{j\omega_d T/2}+e^{-j\omega_d T/2})} = \frac{j2\sin(\frac{\omega_d T}{2})}{2\cos(\frac{\omega_d T}{2})} = j\tan(\frac{\omega_d T}{2}) \quad (5.4-10)$$

Therefore, the relationship between the analog and digital frequencies is given by

$$\omega_c = \frac{\sin(\frac{\omega_d T}{2})}{\cos(\frac{\omega_d T}{2})} = \tan(\frac{\omega_d T}{2}) = \tan(\pi f_d T) \quad (5.4-11)$$

where f_d represents the digital frequency in Hz. Note that the above relationship is nonlinear. For instance, a zero analog frequency is mapped into a zero digital frequency, but the point $\omega_c=\infty$ is mapped into the digital frequency $\omega_d/2$, the folding frequency.

The bilinear transformation provides a one-to-one mapping of the $j\omega$ -axis in the s -plane onto the unit circle in the z -plane, but it compresses the infinite range of analog frequencies to a finite range of digital frequencies specified by the **Nyquist or primary-strip band**: $-\omega_d/2 < \omega < \omega_d/2$. Thus, the frequency response of the analog filter over an infinite range is transformed into the range from 0 to $\omega_d/2$ for the digital filter.

The frequency compression phenomenon associated with the bilinear transformation is known as **frequency warping** and must be remedied when designing digital filters. This can be done by prewarping the critical frequencies, usually the cutoff frequencies of the analog filter, so that the critical frequencies of the digital

filter end up where they belong. The bilinear transformation design procedure is outlined next by examples.

Example 5.4-4

Design the digital filter of Example 5.4-1 using the bilinear transformation method. The analog filter transfer function is given by

$$G(s) = \frac{\omega_c^2}{s^2 + 1.414\omega_c s + \omega_c^2}$$

where ω_c denotes the continuous cutoff frequency selected for prewarping. Using the digital filter specifications, the prewarped analog frequency is

$$\omega_c = \tan\left(\frac{\omega_d T}{2}\right) = \tan\left(\frac{50}{2 \times 60}\right) = 0.4426$$

Substituting the prewarped analog frequency into the filter transfer function, we obtain

$$G(s) = \frac{0.1959}{s^2 + 0.6258s + 0.1959}$$

The desired digital filter transfer function is obtained using the relation

$$G(z) = G(s) \Big|_{s=\frac{z-1}{z+1}} = \frac{0.1959}{\frac{(z-1)^2}{(z+1)^2} + 0.6258\frac{(z-1)}{(z+1)} + 0.1959} = \frac{0.1075z^2 + 0.215z + 0.1075}{z^2 - 0.8828z + 0.3129}$$

To verify how well the digital filters designed by the four previous methods meet the problem specifications, we can study their frequency response characteristics. The magnitudes or amplitudes vs. frequency for the various design methods are summarized in Table 5.4-1, and the phase angles vs. frequency are shown in Table 5.4-2.

Comparing the frequency data in the tables, we see that for the example considered the impulse invariant method has not given as good an approximation for the digital filter as the other methods. The bilinear transformation method seems to have done a relatively good job.

The bilinear transformation method can also be applied to design high-pass, band-pass, and band-reject or notch filters, as shown by the following example.

Table 5.4-1 *Magnitudes vs. frequency for the continuous and digital filters used in Examples 5.4-1 to 5.4-4.*

OMEGA	CONTIN	IMPULSE	STEP	POLE-ZERO	BILINEAR
5.00	0.9999	0.9409	0.9997	0.9988	0.9997
6.29	0.9999	0.9412	0.9995	0.9981	0.9997
7.92	0.9997	0.9416	0.9990	0.9969	0.9995
9.98	0.9992	0.9419	0.9982	0.9947	0.9992
12.56	0.9980	0.9421	0.9964	0.9909	0.9982
15.81	0.9951	0.9415	0.9925	0.9838	0.9958
19.91	0.9877	0.9378	0.9837	0.9700	0.9897
25.06	0.9699	0.9261	0.9636	0.9425	0.9745
31.55	0.9292	0.8951	0.9194	0.8875	0.9377
39.72	0.8458	0.8263	0.8316	0.7858	0.8562
50.00	0.7072	0.7062	0.6880	0.6279	0.7069
62.95	0.5337	0.5515	0.5099	0.4393	0.5047
79.25	0.3699	0.4038	0.3424	0.2678	0.3086
99.76	0.2436	0.2905	0.2126	0.1402	0.1607
125.59	0.1565	0.2153	0.1209	0.0573	0.0653
158.11	0.0995	0.1734	0.0579	0.0115	0.0131
199.05	0.0630	0.1640	0.0374	0.0014	0.0015
250.59	0.0398	0.2139	0.1190	0.0558	0.0635
315.48	0.0251	0.5682	0.5290	0.4592	0.5269
397.16	0.0159	0.9373	0.9827	0.9686	0.9890
500.00	0.0100	0.2206	0.1276	0.0630	0.0718

Table 5.4-2 *Phase Response Characteristics for the continuous and digital filters used in Examples 5.4-1 to 5.4-4.*

OMEGA	CONTIN	IMPULSE	STEP	POLE-ZERO	BILINEAR
5.00	-8.13	-8.59	-10.52	-8.59	-7.66
6.29	-10.25	-10.83	-13.27	-10.83	-9.66
7.92	-12.95	-13.68	-16.74	-13.68	-12.20
9.98	-16.37	-17.29	-21.15	-17.29	-15.43
12.56	-20.76	-21.92	-26.77	-21.92	-19.58
15.81	-26.42	-27.88	-33.98	-27.88	-24.95
19.91	-33.78	-35.62	-43.30	-35.62	-31.98
25.06	-43.42	-45.75	-55.39	-45.75	-41.31
31.55	-56.00	-58.94	-71.04	-58.94	-53.77
39.72	-71.81	-75.55	-90.71	-75.55	-70.12
50.00	-90.00	-94.76	113.71	-94.76	-90.02
62.95	-108.19	-114.26	-137.85	-114.26	-111.01
79.25	-124.00	-131.76	-160.93	-131.76	-129.98
99.76	-136.58	-146.54	-182.02	-146.54	-145.72
125.59	-146.22	-159.16	-200.36	-159.16	-158.85
158.11	-153.58	-170.78	-209.06	-170.78	-170.69
199.05	-159.24	-183.15	-163.94	-183.15	-183.18
250.59	-163.63	-200.54	-159.24	-200.54	-200.85
315.48	-167.05	-247.73	-224.67	-247.73	-251.16
397.16	-169.75	-36.28	-44.09	-36.28	-32.59
500.00	-171.87	-158.12	-198.94	-158.12	-157.77

Example 5.4-5

Design a second-order digital notch filter to eliminate a noise of 60 Hz (377 rad/sec). Use the bilinear transformation method, assuming a second-order Butterworth analog filter denominator. Design a digital filter for each of the following sampling frequencies:

- (a) Sampling frequency of 200 Hz (i.e., $T=0.005$ sec)
- (b) Sampling frequency of 1000 Hz (i.e., $T=0.001$ sec)

Plot frequency amplitude characteristics of the continuous as well as of the two digital filters. In addition, simulate the difference equation of the second digital filter and plot the response for the input $x(t) = \sin 2\pi 10t + 0.2\sin 2\pi 60t$, that is, for an input that consists of a signal of 10 Hz mixed with a noise of 60 Hz.

Solution: From Eq. (5.3-22), the normalized form of the second-order Butterworth band-reject analog filter is of the form

$$G(s) = \frac{s^2 + \omega_c^2}{s^2 + 1.414\omega_c s + \omega_c^2}$$

where the bandwidth B has been replaced by $1.414\omega_c$ in order to match the denominator of our filter with that of the second-order Butterworth filter form. The analog notch filter is obtained from the above transfer function by simply replacing ω_c by $2\pi 60=377$, the critical frequency. This results in the analog notch filter transfer function

$$G(s) = \frac{s^2 + 142129}{s^2 + 533s + 142129}$$

Case (a): To design the digital notch filter for the 200 Hz sampling rate case, we proceed as in the last example. That is, the prewarped analog frequency is

$$\omega_c = \tan\left(\frac{\omega_d T}{2}\right) = \tan\left(\frac{377}{2 \times 200}\right) = 1.37645$$

Substituting this value in the normalized filter transfer function yields

$$G(s) = \frac{s^2 + 1.8946}{s^2 + 1.9463s + 1.8946}$$

The digital filter transfer function then becomes

$$G(z) = G(s) \Big|_{s=\frac{z-1}{z+1}} = \frac{0.598z^2 + 0.3696z + 0.598}{z^2 + 0.3696z + 0.1959}$$

Case (b): For the 1000 Hz sampling rate case, the prewarped analog frequency is

$$\omega_c = \tan\left(\frac{\omega_d T}{2}\right) = \tan\left(\frac{377}{2 \times 1000}\right) = 0.19076$$

The digital filter transfer function then is

$$G(z) = G(s) \bigg|_{s=\frac{z-1}{z+1}} = \frac{0.7935z^2 - 1.4755z + 0.7935}{z^2 - 1.4755z + 0.587}$$

The frequency magnitude responses of the continuous notch filter and the two digital notch filters are shown in Fig. 5.4-1. Note the improvement of the approximation when the sampling rate is increased from 200 Hz to 1000 Hz. That is, the 1000 Hz digital filter resembles the analog filter closely and, in addition, it provides a safer guard band than the 200 Hz digital filter.

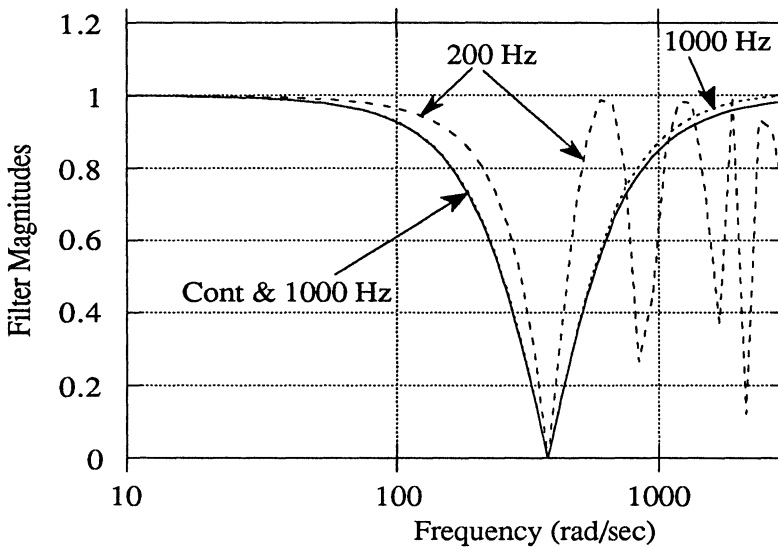


Fig. 5.4-1 Frequency magnitude responses for the 60 Hz continuous and digital notch filters designed in Example 5.4-5.

For the simulation of the digital filter, we can apply the concepts developed in Chapter 3. Simulation results for the 1000 Hz sampling rate case are shown in Fig. 5.4-2. Experimental results obtained from a real-time implementation of the digital filter are shown in Fig. 5.4-3.

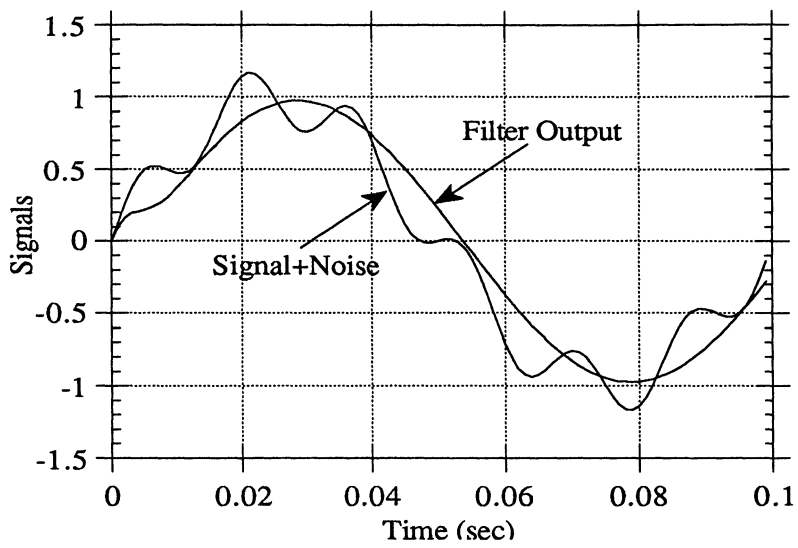


Fig. 5.4-2 A time-domain simulation of the 60 Hz digital notch filter designed in Example 5.4-5. Sampling rate is 1000 Hz.

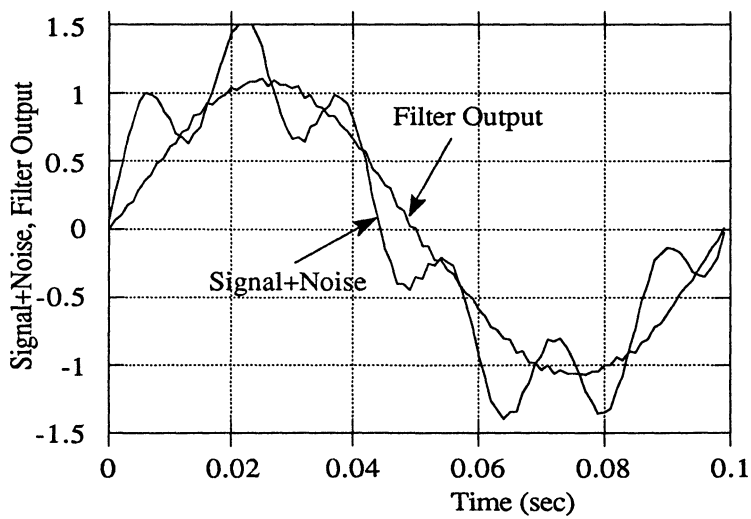


Fig. 5.4-3 Experimental results showing performance characteristics of the 60 Hz digital notch filter used in Example 5.4-5. The filter input is a 10 Hz signal mixed with 60 Hz noise. The output is obtained from the DAC.

5.5 DESIGN OF FINITE IMPULSE RESPONSE (FIR) FILTERS

In a Finite Impulse Response (FIR) or nonrecursive filter type. With this filter structure, the output depends only on present and past values of the input and, thus, the filter achieves its performance characteristics by placement of its zeros, since all of its poles are located at the origin, $z = 0$.

A linear, FIR digital filter can be represented by the equation

$$y(k) = \sum_{i=0}^{N-1} a_i x(k-i) \quad (5.5-1)$$

where $y(k)$ represents the filter output at time $t=kT$, and $x(k-i)$ the inputs at times $(k-i)T$, for $i=0, 1, 2, \dots, N-1$. Compared to IIR filters, FIR filters generally require more terms to produce as sharp of a cutoff, but exhibit much better phase characteristics. FIR filters can also be used to perform operations such as smoothing, prediction (extrapolation), and differentiation, either separately or simultaneously.

FIR filters operate on a number of input samples, considered to be a data window that moves down the time axis at the sampling rate. For an N -point data window, it takes $(N-1)$ sampling times for all of the points to respond to a change in the input. Equivalently, the operation specified by a FIR digital filter has a transient response which lasts $(N-1)$ sampling periods.

For a FIR filter to work as intended, it is important that the input signal can be approximated reasonably well by a relatively low degree polynomial, otherwise distortion may result which is not related to noise.

For a polynomial input $p(t)$ of degree q , the desired form for smoothing and prediction filters is

$$p(k+\alpha) = \sum_{i=0}^{N-1} a_i p(k-i) \quad (5.5-2)$$

Observe that for ' α ' negative, the output lags the input; for ' α ' positive, the output leads or predicts the input. Expanding $p(k+\alpha)$ and $p(k-i)$ in Taylor series about the point kT and collecting terms results in a set of necessary and sufficient conditions that must be satisfied. For polynomials of degree $q \leq (N-1)$, the conditions are

$$\sum_{k=0}^{N-1} a_k = 1, \quad i=0 \quad (5.5-3)$$

$$\sum_{k=0}^{N-1} k a_k = -\alpha, \quad i=1$$

$$\sum_{k=0}^{N-1} k^2 a_k = \alpha^2, \quad i=2$$

$\vdots$

$$\sum_{k=0}^{N-1} k^q a_k = (-1)^q \alpha^q, \quad i=q$$

There are various methods available for designing digital FIR filters. The design criterion usually requires a filter that performs a desired operation while, at the same time, minimizes the effect of noise. The methods of Lagrangian multipliers and least squares are presented next.

5.5-1 Lagrangian Multipliers Method

To determine the a_i coefficients of Eq. (5.5-2) subject to the constraints of Eq. (5.5-3), and at the same time to maximize the rejection of noise that may be present in the input signal, the Lagrangian multipliers method may be used. According to this method, in order to minimize the function

$$f(a_k) = \sum_{k=0}^{N-1} a_k^2 \quad (5.5-4)$$

subject to the constraints specified by Eq. (5.5-3), we minimize the new function

$$g(\lambda_i, a_k) = \sum_{k=0}^{N-1} a_k^2 + \lambda_1 \left(\sum_{k=0}^{N-1} a_k - 1 \right) + \dots + \lambda_q \left\{ \sum_{k=0}^{N-1} k^{q-1} a_k + (-1)^q \alpha^{q-1} \right\} \quad (5.5-5)$$

with respect to a_k . The λ_i in the auxiliary function of Eq. (5.5-5) are known as the Lagrangian multipliers. Taking the derivatives of $g(\lambda_i, a_k)$ with respect to a_k and setting them equal to zero results in the set of equations

$$a_k = \lambda_0 + \lambda_1 k + \lambda_2 k^2 + \dots + \lambda_q k^q \quad (5.5-6)$$

where $k = 0, 1, 2, \dots, N-1$.

For a q th-degree polynomial approximation over an N -point data window, the conditions in Eq. (5.5-3) are combined with Eq. (5.5-6). This results in a set of $N+q+1$ simultaneous equations in $N+q+1$ unknowns, which can be solved using matrix methods. The solution of these equations give the Lagrangian multipliers λ_i , $i=0, 1, 2, \dots, q$, and the filter coefficients a_k , $k=0, 1, 2, \dots, (N-1)$.

The parameter ' α ' shown in the constraint equations provides flexibility in the design. If just conditions for strict linearity between the input signal and filter output are required, then the ' α ' in the constraint equations can be set equal to zero as long as $N \geq q+1$. Furthermore, any larger-size data window, $N > q+1$, would help in the smoothing of the input signal. But if the input signal can be represented by a polynomial of degree $q < N-1$, the variable ' α ' can be manipulated to arrive at many different filter equations for a given-size data window and polynomial degree. The variable ' α ' is sometimes referred to as the prediction variable, because it can be used to obtain filter coefficients for a q th-order filter that will either lead or lag a q th-degree polynomial input signal by αT samples, after the filter response has settled.

Example 5.5-1

Use the method Lagrangian multipliers to design a FIR filter assuming a 4-point data window and a zero-degree (constant) polynomial input. That is, assume that $N=4$, $q=0$, and let $\alpha=0$.

From Eq. (5.5-3), the constraints are

$$a_0 + a_1 + a_2 + a_3 = 1$$

and, from Eq. (5.5-6), the other equations can be obtained from

$$a_k = \lambda_0 \quad ; k=0, 1, 2, 3$$

Solving the set of equations, we obtain $\lambda_0=0.25$. This results in the filter coefficients: $a_0=a_1=a_2=a_3=0.25$. Thus, the digital filter becomes

$$y(k) = \frac{x(k)+x(k-1)+x(k-2)+x(k-3)}{4}$$

which is a 4-point averager. Note, the above equations can also be expressed in matrix form as

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \lambda_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Example 5.5-2

It has been shown by Monroe that for a first-degree polynomial approximation over an N -point data window, the value of ' α ' for maximum noise reduction is given by

$$\alpha = -\frac{N-1}{2}$$

Assuming $N=4$, and $q=1$, we obtain $\alpha = -1.5$. The constraint equations then become

$$a_0 + a_1 + a_2 + a_3 = 1, \text{ and } a_1 + 2a_2 + 3a_3 = 1.5$$

Combining these equations with Eq. (5.5-6) results in the following set of simultaneous equations

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & 0 & 1 & 2 \\ 0 & 0 & 0 & -1 & 1 & 3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \lambda_0 \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1.5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Solving the above equations gives the desired filter coefficients as $a_0=a_1=a_2=a_3=0.25$.

5.5-2 Least-Squares Method

Assume that the input to a FIR filter is a polynomial signal corrupted by noise. In addition, assume that over a finite-size data window of N points, the input signal can be approximated by a q th-degree polynomial, where $q+1 < N$.

A widely used approach for noisy situations is the principle of least-squares approximation. It states that the best estimate of a number is that number which minimizes the sum of the squares of the deviations of the data from their estimate. In terms of the nonrecursive structure of Eq. (5.5-1), a least-squares solution implies finding the filter coefficients, $a_0, a_1, \dots, a_q$, which minimize the expression

$$\varepsilon = \sum_{k=0}^{N-1} [y(k) - x(k)]^2 \quad (5.5-7)$$

In the above equation, $y(k)$ represents the assumed polynomial evaluated at the sampling instants, and $x(k)$ denotes the signal-plus-noise sequence of input samples. The least squares method is illustrated next by an example.

Example 5.5-3

Determine a FIR digital filter assuming that the input signal can be approximated well by a first-degree polynomial over a 4-point window. Use the method of least-squares approximation.

Let the input polynomial be represented in the form

$$y(t) = a + bt$$

Then, the error function to be minimized is given by the

$$\varepsilon = \sum_{k=0}^3 [a + bkT - x(k)]^2$$

To minimize ε , we take the partial derivatives with respect to a and b and set them equal to zero. This results in the equations

$$a \sum_{k=0}^3 1 + bT \sum_{k=0}^3 k = \sum_{k=0}^3 x(k)$$

$$\text{and} \quad a \sum_{k=0}^3 k + bT \sum_{k=0}^3 k^2 = \sum_{k=0}^3 kx(k)$$

Evaluating the summations gives the equations

$$4a + 6Tb = x(0) + x(1) + x(2) + x(3)$$

$$\text{and} \quad 6a + 14Tb = x(1) + 2x(2) + 3x(3)$$

Solving these equations for a and b, we obtain

$$a = \frac{7x(0) + 4x(1) + x(2) - 2x(3)}{10}$$

and

$$b = \frac{-3x(0) - x(1) + x(2) + 3x(3)}{10T}$$

Next, we substitute the expressions for a and b in the polynomial equation

$$y(k) = a + bkT$$

and evaluate it for $k = 3$. This gives

$$y(3) = \frac{-2x(0) + x(1) + 4x(2) + 7x(3)}{10}$$

The last equation can be written in the general form

$$y(k) = \frac{7x(k) + 4x(k-1) + x(k-2) - 2x(k-3)}{10}$$

which represents the desired FIR digital filter.

The above filter has all its poles located at $z=0$ and it attains its performance characteristics by the placement of its zeros. The main function of the filter is to improve the signal-to-noise ratio of some input signal corrupted by noise. This operation is generally referred to as smoothing and its success depends on whether the input signal resembles a first-degree polynomial or not. FIR filters can be analyzed and implemented like IIR filters.

5.5-3 FIR Filters for Differentiation, Smoothing, and Prediction

FIR filters can perform the operations of prediction, smoothing, and differentiation. For instance, a filter can be designed to take the first derivative of a q th-degree polynomial input by using the Lagrangian multipliers method. The necessary constraints for this case are

$$\sum_{k=0}^{N-1} a_k = 0 \quad (5.5-8)$$

$$\begin{aligned} \sum_{k=0}^{N-1} k a_k &= -\frac{1}{T} \\ &\vdots \\ &\vdots \end{aligned}$$

and

$$\sum_{k=0}^{N-1} k^q a_k = \frac{(-1)^q q!}{T}$$

The design procedure is illustrated by the following example.

Example 5.5-4

Using the method of Lagrangian multipliers, design a FIR filter to perform smoothing and differentiation. Assume that the input signal can be approximated by a first-degree polynomial, that is $q=1$, over a 4-point data window. Also, let $\alpha=0$.

From Eq. (5.5-8), the constraints are

$$a_0 + a_1 + a_2 + a_3 = 0, \quad \text{and} \quad a_1 + 2a_2 + 3a_3 = -\frac{1}{T}$$

The Lagrangian multipliers and filter coefficients can then be obtained from Eq. (5.5-6) and the above equations. The resulting digital filter for differentiation and smoothing is

$$y(k) = b \frac{3x(k) + x(k-1) - x(k-2) - 3x(k-3)}{10T}$$

where T is the sampling time.

5.6 DIGITAL FILTER IMPLEMENTATION CONSIDERATIONS

In designing digital filters by various techniques, we assumed that the resulting transfer functions can be characterized by their coefficients which are represented accurately using as many decimal places as needed. To determine whether a digital filter meets frequency domain design specifications or not, we can use frequency analysis programs which provide adequate computational accuracy because they use floating-point arithmetic.

Like any other digital system, a digital filter can be realized using the direct, cascade, parallel, and coupled decomposition schemes discussed in Section 4.3-3. To carry out a digital filter function in real-time, however, we must program the difference equation (s) on a digital computer.

In many real-time applications, fixed-point arithmetic is used because it results in faster and more economical implementations. A fixed-point implementation implies that the filter coefficients, input sequence, and internal data must be represented by a finite number of bits, or a finite wordlength. If the representation is not exact, there will be quantization errors which must be properly handled by the filter programmer for the application to be successful.

There are basically three types of quantization errors. The first type of error is caused by quantizing the input sequence using an Analog-to-Digital Converter (ADC), that is, by using a finite number of bits to represent the input samples $x(k)$. The second type of error occurs when the result of an addition or multiplication operation is rounded in order to fit in a prescribed wordlength. To avoid the first and second type of errors, the wordlength of the ADC and the wordlength of the digital computer used for fixed-point operations must be sufficiently large and coordinated. Finally, the third type of error results when the filter coefficients are represented by a finite number of bits.

If the encoding of the coefficients is not accurate enough, the resulting filter may exhibit characteristics which differ substantially from those of the original design. For instance, the poles and zeros of the implemented filter may end up in locations which are not close enough to the intended ones. This movement of the poles and zeros may in turn cause the frequency response characteristics (i.e., the magnitude, phase, and critical frequencies) to shift appreciably, so that the filter performance becomes unsatisfactory. It is, therefore, important that the frequency characteristics of the implemented filter be recalculated to see if they still meet the specifications.

One way of determining the effects that finite wordlength has on the frequency characteristics of a digital filter is to obtain the transfer function of the approximating filter and analyze its magnitude and phase characteristics. A less practical but more rigorous approach is to use sensitivity analysis. For instance, we can assess the effect of changing a filter coefficient on the frequency response of the filter by differentiating the magnitude function $|G(j\omega)|$ with respect to the coefficient, and use the value of the derivative as an indication of the sensitivity of the frequency characteristics to variations in the value of the particular coefficient.

Some of the problems associated with parameter quantization, and their effect on the performance of the various filter realization forms, can be analyzed and evaluated by the procedure illustrated in the following example.

Example 5.6-1

The transfer function of a third-order, low-pass, Butterworth filter with a cutoff frequency of 50 rad/sec is

$$G(s) = \frac{125000}{s^3 + 100s^2 + 5000s + 125000}$$

The corresponding digital filter designed by the ZOH method for a sampling frequency of 60 Hz is found to be

$$G(z) = \frac{0.06253z^2 + 0.16192z + 0.02717}{z^3 - 1.42409z^2 + 0.86452z - 0.18881} = \frac{0.06253(z+0.18063)(z+2.40912)}{(z-0.43451)[(z-0.49479)^2 + 0.43557^2]}$$

Investigate the coefficient quantization effects of the various filter realization forms or structures. Determine the effect that coefficient resolution of (a) 10 bits, and (b) 6 bits has on the location of the filter poles and zeros.

The filter transfer function has been expressed both in unfactored form for direct or coupled realizations, and in factored form for cascade or parallel realizations.

(a) **10-bit coefficient resolution case.** If rounding is assumed, the filter coefficients are stored as integers determined by multiplying their decimal value by the scale factor 1024 ($2^{10}=1024$). For example, the numerator coefficient 0.06253 is stored as $\text{ROUND}(0.06253 \times 1024) = 64$, and so forth. Thus, for the direct realization form, the above filter transfer function is approximated by

$$G(z) = \frac{64z^2 + 166z + 28}{1024z^3 - 1458z^2 + 885z - 193}$$

The corresponding zeros and poles of the direct filter form are

$$\text{zeros: } -0.1814, -2.4124, \text{ and poles: } 0.4331, 0.4954 \pm j0.4356$$

For the factored forms, parallel/cascade, the transfer functions is

$$G(z) = \frac{64(1024z+185)(1024z+2467)}{(1024z-445)[(1024z-507)^2+446^2]}$$

The resulting zeros and poles are

zeros: -0.1807, -2.4092, and poles: 0.4346, $0.4951 \pm j0.4355$

For the coupled structure, it can be shown that the zeros and poles are

zeros: -0.1797, -2.4082, and poles: 0.4346, $0.4951 \pm j0.4355$

Comparing the approximating filter transfer functions to the original ones, we observe that the poles and zeros of the cascade/parallel form filter have not shifted as much from the original filter locations as the poles and zeros of the direct form filter. In other words, for this case, the cascade/parallel realization is a more accurate representation of the original digital filter. Equivalently, the filter coefficients of the cascade/parallel form realization are less sensitive to coefficient resolution constraints than the coefficients of the direct form.

(b) **6-bit coefficient resolution case.** For a 6-bit resolution, the scale factor is 64. Thus, the coefficients are obtained by multiplying their value by 64 and rounding the result. This leads to the direct form filter transfer function

$$\frac{4z^2 + 10z + 2}{64z^3 - 91z^2 + 55z - 12}$$

with zeros and poles given by

zeros: -0.2192, -2.2808, and poles: 0.4372, $0.4923 \pm j0.4318$

The cascade/parallel form filter is given by

$$G(z) = \frac{4(64z+12)(64z+154)}{(64z-28)[(64z-32)^2+28^2]}$$

with zeros and poles located at

zeros: -0.1875, -2.4062, and poles: 0.4375, $0.5 \pm j0.4375$

It can be shown that the zeros and poles for the coupled structure are

zeros: -0.1912, -2.3088, and poles: 0.4375, $0.5 \pm j0.4375$

The filter of our example is of relatively low order and its poles are located a safe distance from the unit circle. If the poles of the filter lie close to the unit circle, which is normally the case for high-order, sharp cutoff filters, and if the wordlength for representing filter coefficients is not sufficiently large, the poles of the implemented filter may fall outside the unit circle, resulting in an unstable realization. In general, if the wordlength is small and/or the filter is of relatively high order, the direct realization structure should be avoided.

Lack of sufficient accuracy for coefficient encoding or for internal calculations may lead to persistent oscillations referred to as a **limit cycle**. One type of the limit cycle phenomenon is characterized by a periodic output, even though there is no sustained input causing it, as is illustrated by the last example. Another type of limit cycle is caused by arithmetic overflow. At the very least, overflow should be monitored and the calculation results should be limited to prevent it.

Example 5.6-1

Consider a discrete system represented by the transfer function

$$G(z) = \frac{Y(z)}{X(z)} = \frac{z}{z+0.8}$$

Determine the output $y(k)$ when an input $x(k)$ is applied momentarily. Specifically, let the input be represented by the impulse function $x(k)=10\delta(k)$. Plot the system output for the following two cases:

- (a) When the values are quantized using floating-point precision, and
- (b) When the values are quantized by rounding

From the transfer function, the difference equation is

$$y(k) = x(k) - 0.8y(k-1), \quad k=0, 1, 2, \dots,$$

which can be easily simulated for the input sequence $x(k)=\{10, 0, 0, \dots, 0\}$. The results are shown plotted in Fig. 5.6-1. For case (a), the output dies out, which is expected for the impulse response of a stable system. For case (b), however, the output becomes periodic and oscillates between the values of ± 1.6 after the 7th sample.

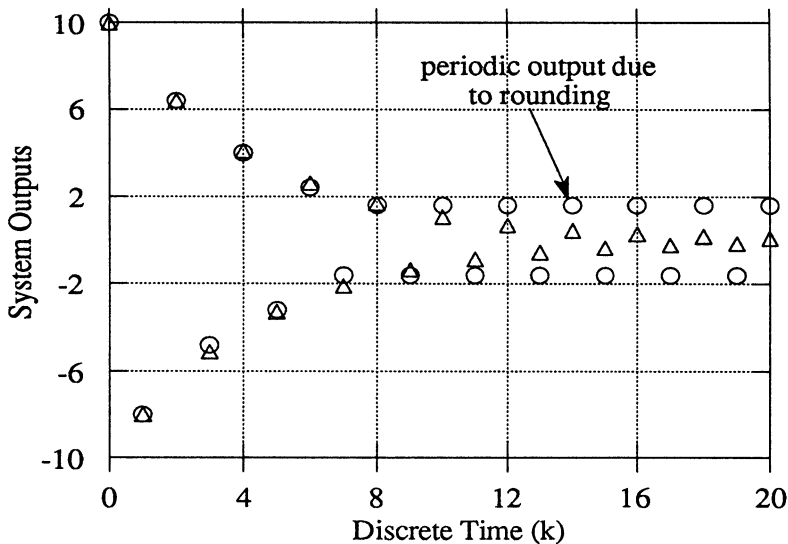


Fig. 5.6-1 System outputs with 'O' and without 'Δ' limit cycle.

PROBLEMS

- 5-1.** Design, that is, determine the transfer function for a first-order low-pass analog filter that has a cutoff/corner frequency of 20 Hz. Then, perform the following:
- (a) determine magnitude and phase versus frequency responses
 - (b) determine and plot its response to the input $x(t) = \sin(10\pi t)$
 - (c) determine and plot its response to the input $x(t) = \sin(80\pi t)$
- 5-2.** Design, that is, find the transfer function of first-order low-pass digital filter that has a corner frequency of 20 Hz. Assume a sampling frequency of 200 Hz. Design the filter by the following methods.
- (1) Impulse invariance
 - (2) step invariance
 - (3) bilinear transformation.
- Simulate the corresponding difference equations and tabulate the results for the inputs (a) $\sin(10\pi t)$, and (b) $\sin(80\pi t)$. How does the performance of this digital filter compare with that of the analog filter in Problem 5-1?
- 5-3.** Design a 60 Hz digital notch filter using (1) the ZOH (step invariance) method, and (2) the Bilinear transformation method. Assume a sampling frequency of 500 Hz. The transfer function of the analog prototype notch filter is given by

$$G(s) = \frac{s^2 + \omega_c^2}{s^2 + \sqrt{2}\omega_c s + \omega_c^2}$$

where ω_c is the notch frequency of the analog filter. Verify your design results by

- (a) time-domain simulation, that is, by programming the corresponding difference equation with an appropriate sinusoidal input.
- (b) frequency domain analysis, preferably by using computer methods. Compare performance characteristics between the analog prototype and the digital filters. Plot the magnitudes vs. frequency (rad/sec) on the same page for all 3 filters.

- 5-4.** Repeat problem 5-4, except use the analog prototype given by

$$G(s) = \frac{s^2 + \omega_c^2}{s^2 + \omega_c s + \omega_c^2}$$

Plot Bode plots for both filters and evaluate their performance.

- 5-5.** A digital filter is described by the transfer function

$$G(z) = \frac{Y(z)}{X(z)} = \frac{z^2}{z^2 - 0.9z + 0.2}$$

Find the corresponding difference equation representations assuming (a) direct decomposition, and (b) cascade decomposition. If the filter coefficients are quantized using a 3-bit coefficient resolution, determine the effect of this quantization on the pole locations of the filter, for both the direct and cascade realizations or structures. Which realization is better and why?

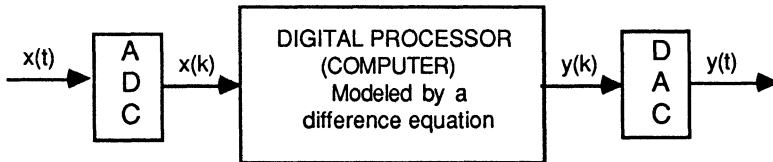
- 5-6. A discrete system is represented by the transfer function

$$G(z) = \frac{Y(z)}{X(z)} = \frac{z}{z+0.95}$$

Determine the output $y(k)$ when an input $x(k)$ is applied momentarily. Specifically, let the input be represented by the impulse function $x(k)=10\delta(k)$. Plot the system output for the following two cases:

- When the values are quantized using floating-point precision, and
 - When the values are quantized by rounding
- Evaluate your findings.

507. Consider the following computer system, with ADC and DAC interfaces.



In the diagram, $x(t)$ and $x(k)$ denote the continuous and discrete system inputs, respectively; $y(k)$ denotes the output of the difference equation and $y(t)$ the corresponding continuous signal/function.

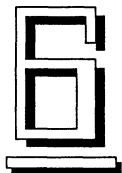
- For this part, disregard the ADC and DAC interfaces and assume that the input is described by the following sequence of numbers, representing amplitudes equally-spaced in time.

$$x(k) = \{0, 1, 2, 3, 4, 5, 5, 5, 5, 0, 0, 0, 0\}$$

Determine, in both tabular as well as graphical (sketch) form, the output $y(k)$ for each of the following cases:

- $y(k) = y(k-1) + x(k)$
- $y(k) = y(k-1) + x(k-1)$
- $y(k) = x(k) - x(k-1)$
- $y(k) = \frac{x(k) + x(k-1)}{2}$

- Assume now that the input sequence, $x(k)$, is obtained by sampling the corresponding continuous function $x(t)$ using an analog-to-digital converter (ADC), as shown in the diagram. For instance, assume that the above values represent voltage levels applied to the input of an 11-bit, 2's complement, ± 5 volt ADC. If the ADC output values are stored in memory, determine their corresponding decimal values. What would be the corresponding continuous values, $y(t)$, assuming a ± 5 volt, 8-bit, 2's complement DAC?



COMPUTER CONTROLLED SYSTEMS:

Computer Algorithms for Deadbeat Response

6.1 INTRODUCTION

The degree of success of a computer control application depends mainly on the effectiveness of the algorithm. Algorithms for digital computer control are generally designed from plant specifications and desired performance characteristics. If the plant is too complex for mathematical analysis, then appropriate mathematical models may be obtained using response data obtained experimentally.

The main objective of this chapter is to develop a systematic procedure for designing digital control systems which respond "ideally" to reference inputs, that is, which can track reference inputs as quickly as possible and with zero steady-state error. Such a "**time-optimal**" response is generally referred to as **deadbeat response**, and the control law as **deadbeat controller** or **algorithm**. More specifically, a deadbeat algorithm is designed to drive a plant output from an arbitrary initial state to a desired final state in the minimum number of sampling times and in such a way that after the output matches the input for the first time, the output becomes identical to the input at all sampling instants, exhibiting little or no ripple between them.

The design of deadbeat algorithms depends on knowing the model of the plant, and it is normally carried out for a specific type of input, such as a step or a ramp function. Deadbeat algorithms require no tuning and, therefore, offer the designer freedom to concentrate on other aspects of the control problem. In the sections which follow, the various control schemes considered involve the design of deadbeat algorithms for open-loop control, closed-loop control, and control when external disturbances enter the control loop. The different mathematical tools employed, namely, Laplace transforms, z transforms, and state variables, are meant to illustrate the alternative methods for solving the digital control problem and, thus, to help the reader understand it better.

6.2 OPEN-LOOP CONTROL: Deadbeat Algorithms

A system is often designed so that its step response is fast enough to meet problem requirements and, in addition, if oscillations are unavoidable, to keep their size and number to a minimum.

The step response of a control system, for instance, is governed by its exponential time constants and its form will be the same for a step input of any size. This means that it should be possible to force the system to respond to an input with greater speed by driving it with a proportionally larger step, followed by a series of delayed steps, the size, number, and duration of which depend on the order of the system and problem requirements. In other words, one can improve considerably the step-response characteristics of a system, namely, its rise time, delay time, settling time, and overshoot, by appropriately manipulating the form of its input.

A method is developed for controlling systems by a computer in an **open-loop** or **batch** manner. For example, the method can be used to improve the characteristics of a feedback control system which has already been designed to perform well when set-point and/or load changes enter its control loop(s).

The objective is to design and implement open-loop computer control algorithms that result in deadbeat response. This means to determine appropriate control actions or manipulations that will drive a system from an arbitrary initial state to a desired final state in minimum time and in such a way that the system output contains little or no oscillations after it matches the reference input for the first time. Even for underdamped systems, the open-loop controller eliminates any oscillations, thus resulting in a ripple-free response.

The proposed control scheme may be implemented as shown in Fig. 6.3-1. The size of the step input is denoted by R_0 and a digital computer is used to output the manipulation command to the plant via a DAC every T sec. The number of manipulation inputs that must be allotted so that the plant may respond in deadbeat manner, equals the order of the plant. This number does not include the last manipulation value which for a unity-gain input, for instance, is equal to the size of the reference input that the plant output must follow. Parts (a), (b), and (c) of Fig. 6.3-2 represent typical manipulation signals used for the control of first-, second-, and third-order plants.

The Laplace transform method is considered first in order to develop appropriate mathematical relationships from which the manipulation values m_0, m_1 , etc. can be calculated.

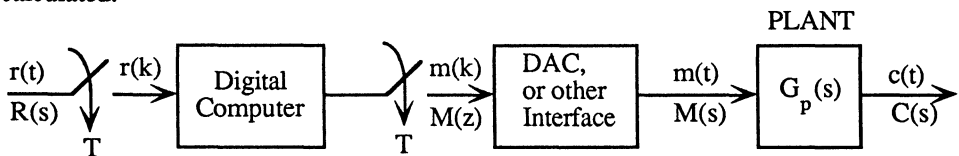


Fig. 6.3-1 Open-loop computer control system.

6.3 THE LAPLACE TRANSFORM METHOD

The Laplace transform method is an effective and efficient tool for solving many problems, including the open-loop control problem for deadbeat response. The method is applied next to analyze the general cases involving first-order, and second-order processes or plants.

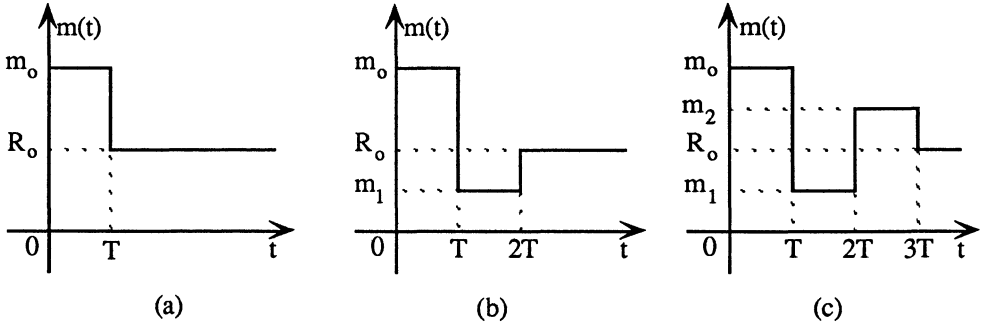


Fig. 6.3-2 Manipulation variable $m(t)$ for (a) first-order, (b) second-order, and (c) third-order processes or plants.

6.3-1 First-Order Plant

Assume that a plant can be adequately described by the linear, unity-gain, first-order transfer function given by

$$G_p(s) = \frac{C(s)}{M(s)} = \frac{a}{s+a} \quad (6.3-1)$$

Assuming the form depicted in Fig. 6.3-2a, the manipulation variable $m(t)$ can be expressed analytically as

$$m(t) = m_o[u(t) - u(t-T)] + R_o u(t-T) = m_o u(t) + (R_o - m_o)u(t-T) \quad (6.3-2)$$

where $u(t)$ denotes the unit-step function. For convenience, let

$$k_o = m_o, \text{ and } k_1 = R_o - m_o \quad (6.3-3)$$

Then, Eq. (6.3-2) becomes

$$m(t) = k_o u(t) + k_1 u(t - T) \quad (6.3-4)$$

Using Laplace transforms

$$M(s) = k_o \frac{1}{s} + k_1 \frac{e^{-sT}}{s} \quad (6.3-5)$$

From Eq. (6.3-1)

$$\begin{aligned} C(s) &= G_p(s)M(s) = \frac{a}{s(s+a)} [k_o + k_1 e^{-sT}] \\ &= k_o \left[\frac{1}{s} - \frac{1}{s+a} \right] + k_1 \left[\frac{e^{-sT}}{s} - \frac{e^{-sT}}{s+a} \right] \end{aligned} \quad (6.3-6)$$

Taking the inverse Laplace transform results in the response function

$$c(t) = k_0[1 - e^{-at}]u(t) + k_1[1 - e^{-a(t-T)}]u(t-T) \quad (6.3-7)$$

For the n th-order plant case, the conditions for deadbeat response to step inputs can be stated as follows:

1. The plant response must match the reference input at time $t=nT$

2. The derivative of the response must be set equal to zero for all sampling times greater than or equal to nT

The second condition guarantees that the plant output will not change or ripple after it matches the input for the first time.

From Eq. (6.3-7), the derivative of the plant response is obtained as

$$c'(t) = k_0 a e^{-at} u(t) + k_1 a e^{-a(t-T)} u(t-T) \quad (6.3-8)$$

Assuming a step input of size R_o , and applying the above deadbeat conditions to Eq. (6.3-7) and Eq. (6.3-8) gives

$$c(T) = k_0(1 - e^{-aT}) + k_1(1 - 1) = R_o \quad (6.3-9)$$

and

$$c'(T) = k_0 a e^{-aT} + k_1 a = 0 \quad (6.3-10)$$

Eq. (6.3-9) implies that

$$k_0 = \frac{R_o}{1 - e^{-aT}} \quad (6.3-11)$$

Also, from Eq. (6.3-10),

$$k_1 = -k_0 e^{-aT} \quad (6.3-12)$$

Substituting Eq. (6.3-11) and Eq. (6.3-12) into Eq. (6.3-7) gives the desired results

$$m_0 = \frac{R_o}{1 - e^{-aT}} \quad (6.3-13)$$

and

$$k_1 = \frac{-R_o e^{-aT}}{1 - e^{-aT}} \quad (6.3-14)$$

Eq. (6.3-13) and Eq. (6.3-14) may then be substituted into Eq. (6.3-7) to obtain the manipulated step response for a first-order plant. The form of this response is given by

$$c(t) = \frac{R_o}{1 - e^{-aT}} [(1 - e^{-st})u(t) - e^{-aT}(1 - e^{-a(t-T)})u(t-T)] \quad (6.3-15)$$

Thus, for open-loop control and deadbeat response of a first-order plant, the first manipulation pulse has amplitude given by Eq. (6.3-13). Consecutive pulses have amplitudes equal to R_o , and all pulses are of width or duration T sec.

Example 6.3-1

Consider the plant described by its transfer function

$$G_p(s) = \frac{1}{s+1}$$

It is required to manipulate the input of the plant so that its response reaches steady state in 0.5 sec.

The time constant of the plant is 1 sec which implies that the normal step response will be within 2% of its steady-state value in about 4 time constants or 4 sec. From Eq. (6.3-13), substituting the parameter values of $a = 1$ and $T = 0.5$ sec, we obtain the desired manipulation value

$$m_o = \frac{R_o}{1-e^{-0.5}} = 2.54149R_o$$

Thus, for a step response of size R_o , the manipulation function is described by

$$m(t) = \begin{cases} 2.54149R_o & \text{for } 0 \leq t < T \\ R_o & \text{for } t \geq T \end{cases}$$

Digital simulation results showing both the normal and the manipulated unit-step responses of the first-order plant are shown in Fig. 6.3-3a and Fig. 6.3-3b, respectively. Similar results obtained experimentally are shown in Fig. 6.3-4a and Fig. 6.3-4b.

Even though it is possible to achieve extremely short response times by making the sampling period arbitrarily small, excessive manipulation signals may be impractical to implement and/or may result in amplifier saturation or cause damage to a physical system.

6.3-2 Second-Order Plant: Overdamped Case

Consider the plant described by its transfer function

$$G_p(s) = \frac{C(s)}{M(s)} = \frac{ab}{(s+a)(s+b)} \quad (6.3-16)$$

It is required to manipulate the input of the plant so that its response reaches steady state in deadbeat fashion.

From Fig. 6.3-2b, the manipulation variable $m(t)$ can be written as

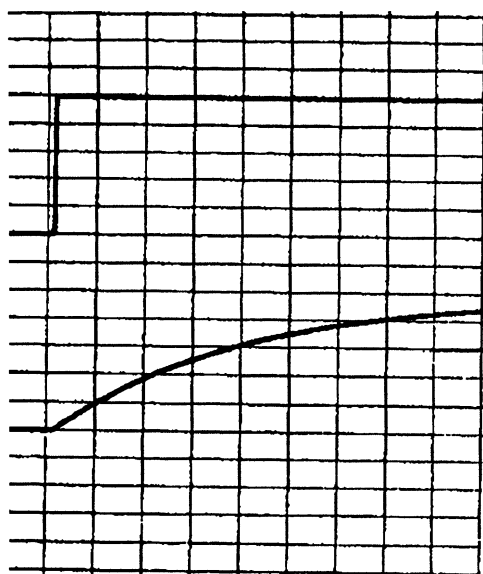
$$m(t) = m_o u(t) + (m_1 - m_o) u(t-T) + (R_o - m_1) u(t-2T) \quad (6.3-17)$$

Again, for convenience, let

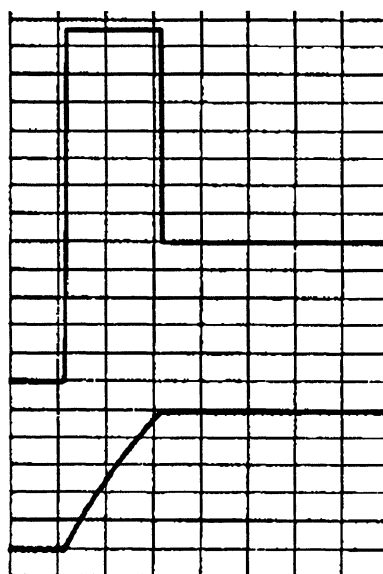
$$k_o = m_o, \quad k_1 = m_1 - m_o, \quad \text{and} \quad k_2 = R_o - m_1 \quad (6.3-18)$$

TIME	OUTPUT		TIME	OUTPUT
0.0	0.0		0.0	0.0
0.5	0.39347		0.25	0.56218
1.0	0.63212		0.5	0.99999
1.5	0.77687		0.75	1.0
2.0	0.86467	(a)	1.0	1.0
2.5	0.91791		1.25	1.0
3.0	0.95021		1.5	1.0
3.5	0.96980		2.0	1.0
4.0	0.98168			
4.5	0.98889			
5.0	0.99326			
5.5	0.99591			
6.0	0.99752			
6.5	0.99850			
7.0	0.99909			

Fig. 6.3-3. Digital simulation results of normal (a) and manipulated (b) step responses for the plant $1/(s+1)$; $T=0.5$ sec.



(a)



(b)

Fig. 6.3-4. Experimental results of normal (a) and manipulated (b) step responses for the plant $G_p(s) = \frac{1}{(s+1)}$; $T=0.5$ sec.

Then, the manipulation signal can be expressed in the form

$$m(t) = k_0 u(t) + k_1 u(t-T) + k_2 u(t-2T) \quad (6.3-19)$$

Using Laplace transforms,

$$M(s) = k_0 \frac{1}{s} + k_1 \frac{e^{-sT}}{s} + k_2 \frac{e^{-2sT}}{s} \quad (6.3-20)$$

From Eq. (6.3-16), the output transform can be written as

$$C(s) = \frac{ab}{s(s+a)(s+b)} [k_0 + k_1 e^{-sT} + k_2 e^{-2sT}] \quad (6.3-21)$$

Partial fraction expansion gives

$$\begin{aligned} C(s) = & k_0 \left(\frac{1}{a-b} \right) \left[(a-b) \frac{1}{s} + b \frac{1}{s+a} - a \frac{1}{s+b} \right] + k_1 \left(\frac{1}{a-b} \right) \left[(a-b) \frac{e^{-sT}}{s} + b \frac{e^{-sT}}{s+a} - a \frac{e^{-sT}}{s+b} \right] \\ & + k_2 \left(\frac{1}{a-b} \right) \left[(a-b) \frac{e^{-2sT}}{s} + b \frac{e^{-2sT}}{s+a} - a \frac{e^{-2sT}}{s+b} \right] \end{aligned} \quad (6.3-22)$$

Multiplying both sides of Eq. (6.3-22) by $(a-b)$, and taking the inverse Laplace transform, results in the response function

$$\begin{aligned} c(t)(a-b) = & k_0 [(a-b) + be^{-at} - ae^{-bt}] u(t) + k_1 [(a-b) + be^{-a(t-T)} - ae^{-b(t-T)}] u(t-T) \\ & + k_2 [(a-b) + be^{-a(t-2T)} - ae^{-b(t-2T)}] u(t-2T) \end{aligned} \quad (6.3-23)$$

Differentiating the expression in Eq. (6.3-23) we get

$$\begin{aligned} c'(t)(a-b) = & k_0 [-abe^{-at} + abe^{-bt}] u(t) + k_1 [-abe^{-a(t-T)} + abe^{-b(t-T)}] u(t-T) \\ & + k_2 [-abe^{-a(t-2T)} + abe^{-b(t-2T)}] u(t-2T) \end{aligned} \quad (6.3-24)$$

The conditions for deadbeat response of a second-order plant imply that

$$c(2T) = R_0, \text{ and } c'(2T) = 0 \quad (6.3-25)$$

Applying these conditions to Eq. (6.3-23) and Eq. (6.3-24), respectively, results in the system of simultaneous algebraic equations

$$(a-b + be^{-2aT} - ae^{-2bT})k_0 + (a-b + be^{-aT} - ae^{-bT})k_1 = R_0(a-b) \quad (6.3-26)$$

and

$$(e^{-2bT} - e^{-2aT})k_0 + (e^{-bT} - e^{-aT})k_1 = 0 \quad (6.3-27)$$

Solving the above equations for k_0 and k_1 , and then substituting the values in Eq. (6.3-18), gives

$$m_0 = \frac{R_o(e^{-bT}-e^{-aT})}{e^{-bT}-e^{-aT}+e^{-2aT}-e^{-2bT}+e^{-aT}e^{-2bT}-e^{-2aT}e^{-bT}} \quad (6.3-28)$$

and

$$m_1 = \frac{R_o(e^{-bT}-e^{-aT}-e^{-2bT}+e^{-2aT})}{e^{-bT}-e^{-aT}+e^{-2aT}-e^{-2bT}+e^{-aT}e^{-2bT}-e^{-2aT}e^{-bT}} \quad (6.3-29)$$

To check the correctness of Eq. (6.3-28) and Eq. (6.3-29), assume that $b \rightarrow \infty$ in Eq. (6.3-16). That is, assume that Eq. (6.3-16) reduces to the first-order plant case. Then, from Eq. (6.3-28)

$$m_0 = \frac{R_o(-e^{-aT})}{-e^{-aT}+e^{-2aT}} = \frac{R_o}{1-e^{-aT}} \quad (6.3-30)$$

and from Eq. (6.3-29)

$$m_1 = \frac{R_o(-e^{-aT}+e^{-2aT})}{-e^{-aT}+e^{-2aT}} = R_o \quad (6.3-31)$$

As expected, the above results agree with those obtained in Section 6.2-1.

Example 6.3-2

As an example of open-loop deadbeat control for a second-order, overdamped case, consider the plant described by its transfer function

$$G_p(s) = \frac{2}{(s+1)(s+2)}$$

It is required to manipulate the input of the plant so that its response reaches steady state in 1 sec.

The time constants of this plant are 0.5 sec and 1 sec, respectively. This means that the normal step response of the plant will take approximately 4 sec to settle. To manipulate the same step response so that it will settle sooner in 1.0 sec, the relationships of Eq. (6.3-28) and Eq. (6.3-29) must be evaluated and used to excite the plant. Since this is a second-order plant, it will take two sampling times to settle, and thus $T = 0.5$ sec. Substituting $a=1$ and $b=2$ in Eq. (6.3-28) and Eq. (6.3-29), we obtain

$$m_0 = 4.02058R_o ; \quad m_1 = 0.102887R_o$$

Thus, for a step of size R_o , the manipulation signal becomes

$$m(t) = \begin{cases} 4.0206R_o & \text{for } 0 \leq t < T \\ 0.10289R_o & \text{for } T \leq t < 2T \\ R_o & \text{for } t \geq 2T \end{cases}$$

Simulation results showing both the normal and the manipulated unit-step responses of the plant are shown in Fig. 6.3-5a and 6.3-5b, respectively. Experimental results for the manipulated step response are shown in Fig. 6.3-6.

TIME	OUTPUT		TIME	OUTPUT
0.0	0.0		0.0	0.0
0.5	0.15482		0.25	0.19672
1.0	0.39958		0.5	0.62246
1.5	0.60353		0.75	0.92763
2.0	0.74765	(a)	1.0	0.99999
2.5	0.84257		1.25	1.0
3.0	0.90291		1.5	1.0
3.5	0.94052		2.0	1.0
4.0	0.96370			
4.5	0.97791			
5.0	0.98657			
5.5	0.99184			
6.0	0.99505			
6.5	0.99699			
7.0	0.99818			

Fig. 6.3-5 Digital simulation results of (a) normal and (b) manipulated step responses for the plant used in Example 6.3-2.

6.3-3 Second-Order Plant: *Underdamped Case*

Open-loop computer control has practical implications, particularly in situations where the dynamic characteristics of a batch plant are represented by an underdamped mathematical mode. In such cases, the deadbeat method may be applied, for instance, to adjust the input to the plant when starting a cycle.

To develop the relationships from which appropriate manipulation values can be calculated, an approach similar to the previous one may be followed.

Let a second-order, underdamped plant be described by the transfer function

$$G_p(s) = \frac{C(s)}{M(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (6.3-32)$$

where the parameters ζ and ω_n represent the damping ratio, and the (undamped) natural frequency, respectively. From Eq. (6.3-20)

$$M(s) = k_0 \frac{1}{s} + k_1 \frac{e^{-sT}}{s} + k_2 \frac{e^{-2sT}}{s} \quad (6.3-33)$$

Then, from Eq. (6.3-32),

$$C(s) = \frac{\omega_n^2}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} [k_0 + k_1 e^{-sT} + k_2 e^{-2sT}] \quad (6.3-34)$$

The step response of the underdamped second-order plant is given by

$$\begin{aligned} c(t) &= 1 - e^{-\zeta\omega_n t} \left[\cos\omega_n \sqrt{1-\zeta^2} t + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin\omega_n \sqrt{1-\zeta^2} t \right] \\ &= 1 + \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin(\omega_n \sqrt{1-\zeta^2} t - \phi) \end{aligned} \quad (6.3-35)$$

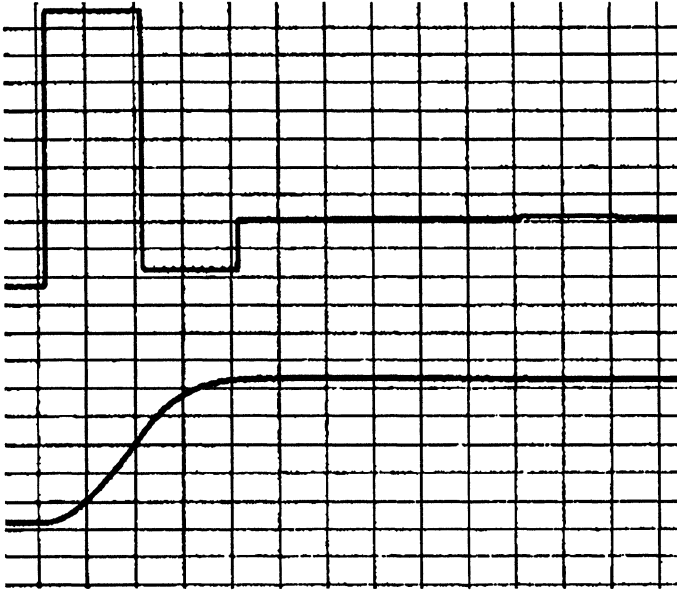


Fig. 6.3-6 Experimental results showing the manipulated step response for the plant used in Example 6.3-2.

where
$$\phi = \tan^{-1}\left(\frac{\sqrt{1-\zeta^2}}{-\zeta}\right) \quad (6.3-36)$$

or
$$\sin\phi = \sqrt{1-\zeta^2}, \quad \cos\phi = -\zeta \quad (6.3-37)$$

The derivative of the step response is given by

$$\begin{aligned} c'(t) &= \frac{-\zeta\omega_n e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin(\omega_n \sqrt{1-\zeta^2} t - \phi) + \frac{-e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \omega_n \sqrt{1-\zeta^2} \cos(\omega_n \sqrt{1-\zeta^2} t - \phi) \\ &= \frac{\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin\omega_n \sqrt{1-\zeta^2} t \end{aligned} \quad (6.3-38)$$

Applying the deadbeat conditions to Eq. (6.3-35) and Eq. (6.3-38) means

$$c(2T) = R_0 \quad (6.3-39)$$

and
$$c'(2T) = 0 \quad (6.3-40)$$

This results in the following system of equations in k_0 and k_1

$$\left[1 + \frac{e^{-2\zeta\omega_n T}}{\sqrt{1-\zeta^2}} \sin(2\omega_n \sqrt{1-\zeta^2} T - \phi)\right]k_0 + \left[1 + \frac{e^{-\zeta\omega_n T}}{\sqrt{1-\zeta^2}} \sin(\omega_n \sqrt{1-\zeta^2} T - \phi)\right]k_1 = R_0 \quad (6.3-41)$$

and

$$[e^{-\zeta\omega_n T} \sin 2\omega_n \sqrt{1-\zeta^2} T]k_0 + [\sin \omega_n \sqrt{1-\zeta^2} T]k_1 = 0 \quad (6.3-42)$$

Solving the above equations, and employing appropriate trigonometric identities, leads to the manipulation values

$$m_0 = \frac{R_0}{1 - 2e^{-\zeta\omega_n T} \cos \omega_n \sqrt{1-\zeta^2} T + e^{-2\zeta\omega_n T}} \quad (6.3-43)$$

and

$$m_1 = \frac{R_0(1 - 2e^{-\zeta\omega_n T} \cos \omega_n \sqrt{1-\zeta^2} T)}{1 - 2e^{-\zeta\omega_n T} \cos \omega_n \sqrt{1-\zeta^2} T + e^{-2\zeta\omega_n T}} \quad (6.3-44)$$

Example 6.3-3

As a specific example of open-loop deadbeat control for a second-order, underdamped case, consider the transfer function model given by

$$G_p(s) = \frac{4}{s^2 + s + 4}$$

Determine the manipulation values for deadbeat response in 1 sec.

For this plant, $\zeta = 0.25$, and $\omega_n = 2$ rad/sec. The time constant of this second-order, underdamped system is

$$\tau = \frac{1}{\zeta\omega_n} = 2 \text{ sec}$$

which implies a settling time of 4τ , or about 8 sec. To manipulate the input of this plant so that it will settle in 1 sec, the sampling time must be $T=0.5$ sec. Then, from Eq. (6.3-43) and Eq. (6.3-44), the desired manipulation values are

$$m_0 = \frac{R_0}{0.72377} = 1.38167R_0$$

and

$$m_1 = \frac{0.11724R_0}{0.72377} = 0.16198R_0$$

Relevant simulation and experimental results are shown in Fig. 6.3-7 and Fig. 6.3-8.

TIME	OUTPUT		TIME	OUTPUT
0.0	0.0		0.0	0.0
0.5	0.39295		0.25	0.15594
1.0	1.07064		0.5	0.54292
1.5	1.43056		0.75	0.88834
2.0	1.33723 (a)		1.0	1.00001 (b)
2.5	1.03655		1.25	1.00001
3.0	0.82772		1.5	1.0
3.5	0.82575		2.0	1.0
4.0	0.95067			
4.5	1.06214			
5.0	1.08478			
5.5	1.03715			
6.0	0.98137			
6.5	0.96103			
7.0	0.97689			
7.5	1.00324			
8.0	1.01688			

Fig. 6.3-7 Digital simulation results of (a) normal, and (b) manipulated step responses for the plant: $\frac{4}{s^2+s+4}$; $T=0.5$ sec.

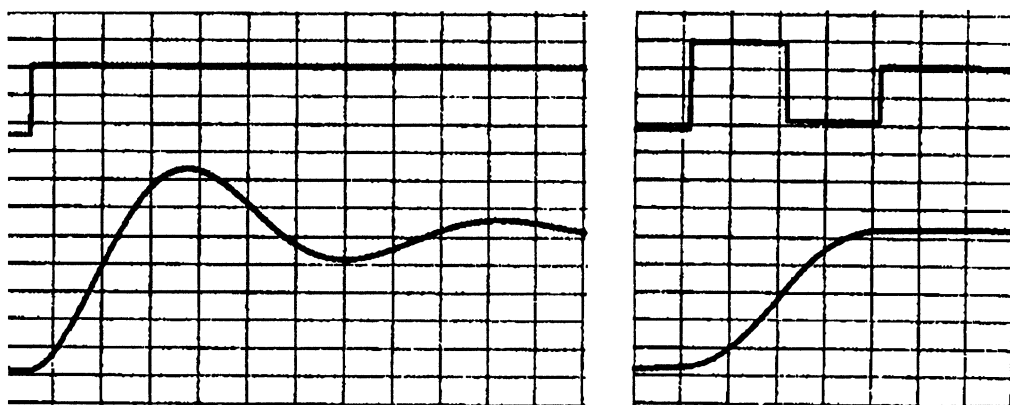


Fig. 6.3-8 Experimental results for normal (left) and manipulated (right) step responses for the plant: $G_p(s) = \frac{4}{s^2+s+4}$; $T=0.5$ sec.

6.3-4 "Positive Cast" or "Posicast" Control

A special case of the deadbeat control described above is what O.J. Smith has called "Positive Cast" or "Posicast" control. For Posicast control of a second-order, underdamped plant, the sampling period is set equal to one half the time at which the maximum overshoot occurs in its step response. From Section 2.9-3, the time for maximum overshoot is given by Eq. (2.9-40). The value of the sampling time then becomes

$$T = \frac{\pi}{2\omega_n \sqrt{1-\zeta^2}} \tag{6.3-45}$$

Substituting this value of T in Eq. (6.3-43) and Eq. (6.3-44) results in the Posicast control manipulation values

$$m_0 = m_1 = \frac{R_o}{1 + e^{-\pi\zeta/\sqrt{1-\zeta^2}}} \tag{6.3-46}$$

For the second-order plant used in Example 6.3-3, this implies

$$m_0 = m_1 = 0.69236 R_o \tag{6.3-47}$$

Simulation as well as experimental results for Posicast control of the plant used in Example 6.3-3 are shown in Fig. 6.3-9a and Fig. 6.3-9b, respectively.

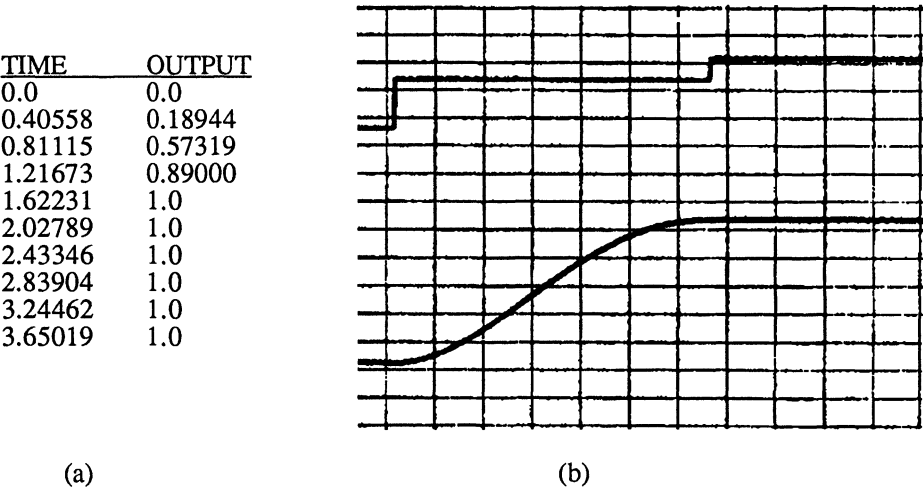


Fig. 6.3-9 Digital simulation (a) and experimental results (b) for Posicast control for the plant $G_p(s) = \frac{4}{s^2+s+4}$; $T = \frac{\pi}{2\omega_n \sqrt{1-\zeta^2}} = 0.81155$ sec.

6.3-5 Second-Order Plant: Critically-Damped Case

In the second-order, critically damped case, the transfer function of the plant may be represented by

$$G_p(s) = \frac{a^2}{(s+a)^2} \quad (6.3-48)$$

Then, following the approach developed previously,

$$C(s) = \frac{a^2}{s(s+a)^2} [k_0 + k_1 e^{-sT} + k_2 e^{-2sT}]$$

$$C(s) = k_0 \left[\frac{1}{s} - \frac{1}{s+a} - \frac{a}{(s+a)^2} \right] + k_1 \left[\frac{e^{-sT}}{s} - \frac{e^{-sT}}{s+a} - \frac{ae^{-sT}}{(s+a)^2} \right] + k_2 \left[\frac{e^{-2sT}}{s} - \frac{e^{-2sT}}{s+a} - \frac{ae^{-2sT}}{(s+a)^2} \right] \quad (6.3-49)$$

Taking the inverse Laplace transform of Eq. (6.3-59) gives

$$\begin{aligned} c(t) = & k_0 [1 - e^{-at} - ate^{-at}] u(t) + k_1 [1 - e^{-a(t-T)} - a(t-T)e^{-a(t-T)}] u(t-T) \\ & + k_2 [1 - e^{-a(t-2T)} - a(t-2T)e^{-a(t-2T)}] u(t-2T) \end{aligned} \quad (6.3-50)$$

from which the derivative becomes

$$c'(t) = k_0 a^2 t e^{-at} u(t) + k_1 a^2 (t-T) e^{-a(t-T)} u(t-T) + k_2 a^2 (t-2T) e^{-a(t-2T)} u(t-2T) \quad (6.3-51)$$

Applying the deadbeat conditions, that is, setting

$$c(2T) = R_o, \text{ and } c'(2T) = 0 \quad (6.3-52)$$

results in the system of equations

$$(1 - e^{-2aT} - 2aTe^{-2aT})k_0 + (1 - e^{-aT} - aTe^{-aT})k_1 = R_o$$

$$\text{and} \quad 2e^{-aT}k_0 + k_1 = 0 \quad (6.3-53)$$

Solving the previous equations for k_0 and k_1 , and recalling that $k_0 = m_0$ and $m_1 = m_0 + k_1$, results in the relationships

$$m_0 = \frac{R_o}{(1 - e^{-aT})^2} \quad (6.3-54)$$

$$\text{and} \quad m_1 = \frac{R_o(1 - 2e^{-aT})}{(1 - e^{-aT})^2} \quad (6.3-55)$$

Thus, if a second-order, critically-damped plant is manipulated by the above values, it will exhibit deadbeat response characteristics for step inputs.

Example 6.3-4

Consider the second-order, critically-damped plant described by the transfer function

$$G_p(s) = \frac{1}{(s+1)^2}$$

and assume, again, that $T = 0.5$ sec. Then, from Eq. (6.3-54) and Eq. (6.3-55),

$$m_o = \frac{R_o}{0.15482} = 6.45919R_o$$

and

$$m_1 = -1.3762R_o$$

Simulation and experimental results for this case are shown in Fig. 6.3-10.

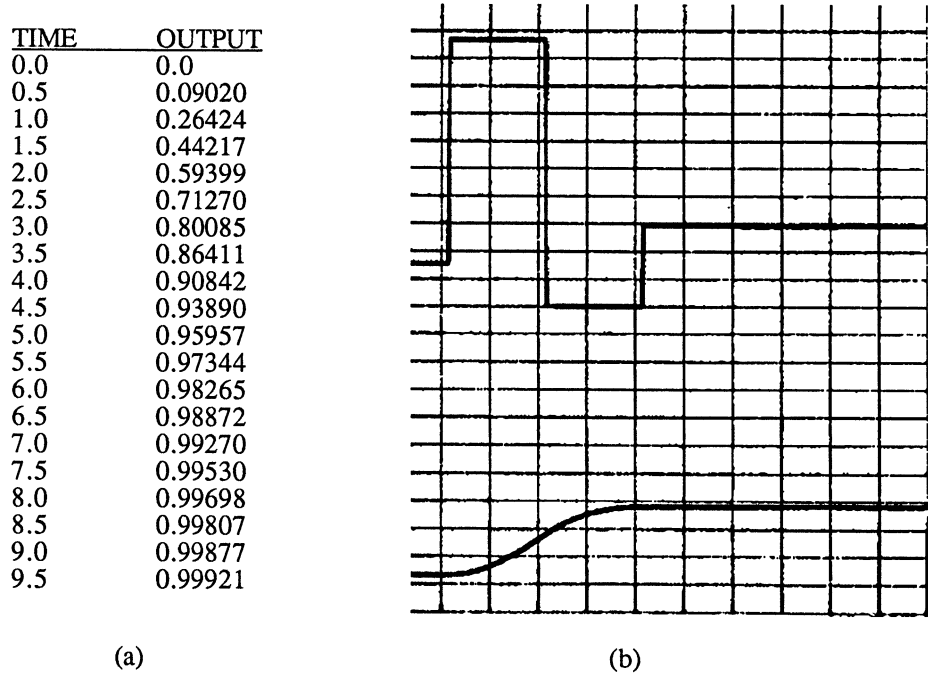


Fig. 6.3-10 Open-loop control of the plant $G_p(s) = \frac{1}{(s+1)^2}$; $T=0.5$ sec. Shown are simulation results for the normal step response (a) and experimental results for the manipulated step response (b).

6.4 THE $\mathcal{Z}$ TRANSFORM METHOD

The open-loop controller design method presented earlier can also be developed using $\mathcal{Z}$ transform techniques. The corresponding system block diagram is shown in Fig. 6.4-1. In the block diagram, $G_c(z)$ represents the transfer function of the digital controller (algorithm), and $G_h(s)$ the transfer function of the Zero-Order-Hold (ZOH) element whose transfer function was determined in Chapter 3 and is given by

$$G_h(s) = \frac{1 - e^{-sT}}{s} \quad (6.4-1)$$

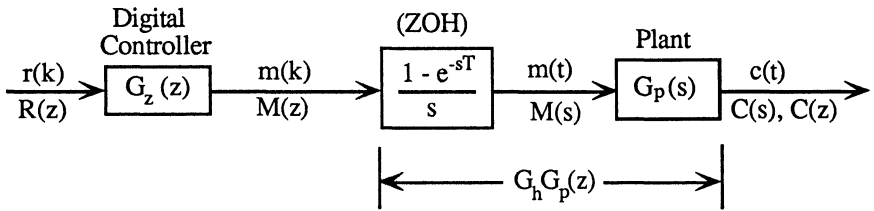


Fig. 6.4-1 Open-loop control system.

From the block diagram, the relevant transfer function expressions are

$$G_c(z) = \frac{M(z)}{R(z)} \quad (6.4-2)$$

and

$$G_h G_p(z) = \frac{C(z)}{M(z)} \quad (6.4-3)$$

where $G_h G_p(z)$ is the pulse transfer function of the system. Solving Eq.(6.4-3) for $M(z)$, and substituting in Eq. (6.4-2) gives the desired controller transfer function $G_c(z)$ as

$$G_c(z) = \frac{C(z)/R(z)}{G_h G_p(z)} \quad (6.4-4)$$

As a case study, consider a first-order plant described by its transfer function

$$G_p(s) = \frac{a}{s+a} \quad (6.4-5)$$

The pulse transfer function then is

$$G_h G_p(z) = \mathcal{Z}\left[\frac{1 - e^{-sT}}{s} \cdot \frac{a}{s+a}\right] = (1 - z^{-1}) \mathcal{Z}\left[\frac{a}{s(s+a)}\right]$$

or

$$G_h G_p(z) = (1 - z^{-1}) \mathcal{Z}\left[\frac{1}{s} - \frac{1}{s+a}\right] = \frac{1 - e^{-aT}}{z - e^{-aT}} \quad (6.4-6)$$

The corresponding difference equation model of the plant can be determined from the pulse transfer function

$$G_h G_p(z) = \frac{C(z)}{M(z)} = \frac{1-e^{-aT}}{z-e^{-aT}} = \frac{(1-e^{-aT})z^{-1}}{(1-e^{-aT}z^{-1})} \quad (6.4-7)$$

by cross-multiplication. This gives

$$(1-e^{-aT}z^{-1})C(z) = (1-e^{-aT})z^{-1}M(z) \quad (6.4-8)$$

The difference equation then becomes

$$c(k) = e^{-aT}c(k-1) + (1-e^{-aT})m(k-1) \quad (6.4-9)$$

The deadbeat conditions for processes/plants represented by difference rather than differential equations are

$$1. \quad c(nT) = c(n) = R_o, \text{ and} \quad (6.4-10)$$

$$2. \quad c'(kT) = c'(k) = c(k+1) - c(k) = 0, \text{ for } k \geq n \quad (6.4-11)$$

where n represents the order of the process/plant.

In the case of a first-order plant, for instance, the deadbeat conditions become

$$c(1) = R_o, \text{ and } c(2)-c(1) = 0 \quad (6.4-12)$$

Assuming the plant at rest, that is, with $c(0)=0$ in Eq. (6.4-9), we obtain

$$c(1) = R_o = (1-e^{-aT})m(0) \quad (6.4-13)$$

$$\text{and} \quad c(2) = e^{-aT}R_o + (1-e^{-aT})m(1) \quad (6.4-14)$$

The derivative condition results in the equation

$$c(2)-c(1) = e^{-aT}R_o + (1-e^{-aT})m(1) - R_o = 0 \quad (6.4-15)$$

from which

$$m(1) = \frac{R_o(1-e^{-aT})}{(1-e^{-aT})} = R_o \quad (6.4-16)$$

From Eq. (6.4-13)

$$m(0) = \frac{R_o}{1-e^{-aT}} \quad (6.4-17)$$

which agrees with Eq. (6.3-13). Note the notation $m(0)=m_0$, $m(1)=m_1$, etc.

The manipulation signal can be written in transform form as

$$M(z) = m(0) + R_o \frac{z^{-1}}{1-z^{-1}} = \frac{m(0) + [R_o - m(0)]z^{-1}}{(1-z^{-1})} \quad (6.4-18)$$

Assuming a step function input of arbitrary size R_o

$$R(z) = R_o \frac{1}{1-z^{-1}} \quad (6.4-19)$$

the digital controller transfer function becomes

$$G_c(z) = \frac{M(z)}{R(z)} = \frac{1}{R_o} \{ m(0) + [R_o - m(0)]z^{-1} \} = \frac{1 - e^{-aT}z^{-1}}{1 - e^{-aT}} \quad (6.4-20)$$

The control algorithm can be expressed in difference equation form as

$$m(k) = \frac{1}{(1 - e^{-aT})} [r(k) - e^{-aT}r(k-1)] \quad (6.4-21)$$

The difference equations given by Eq. (6.4-21) and Eq. (6.4-9) can be used to simulate the control scheme of Fig. 6.4-1 on a digital computer.

To verify the result, an alternative approach would be to multiply Eq. (6.4-20) with Eq. (6.4-6). This leads to the overall transfer function

$$\frac{C(z)}{R(z)} = z^{-1} \quad (6.4-22)$$

from which the system output becomes

$$c(k+1) = r(k) \quad (6.4-23)$$

Eq. (6.4-23) establishes that, for a first-order plant, the plant output follows the reference input one sampling time later.

Example 6.4-1

For a specific example, use the $\mathcal{Z}$ transform method to determine the manipulation values necessary for deadbeat response of the second-order, overdamped plant used in Example 6.3-2. Assume $T=0.5$ sec. The plant transfer function is given by

$$G_p(s) = \frac{2}{(s+1)(s+2)}$$

The pulse transfer function then is

$$G_h G_p(z) = \mathcal{Z}\left[\frac{(1-e^{-sT})2}{s(s+1)(s+2)}\right] = (1-z^{-1})\mathcal{Z}\left[\frac{1}{s} - \frac{2}{s+1} + \frac{1}{s+2}\right] = (z-1)\left[\frac{1}{z-1} - \frac{2}{z-e^{-T}} + \frac{1}{z-e^{-2T}}\right]$$

$$= \frac{0.15482z+0.0939}{(z-0.60653)(z-0.36788)} = \frac{0.15482z^{-1}+0.0939z^{-2}}{1-0.97441z^{-1}+0.22313z^{-2}}$$

The corresponding difference equation model of the plant is

$$c(k) = 0.97441c(k-1) - 0.22313c(k-2) + 0.15482m(k-1) + 0.0939m(k-2)$$

from which, assuming the plant at rest,

$$c(1) = 0.15482m(0)$$

$$c(2) = 0.97441c(1) + 0.15482m(1) + 0.0939m(0)$$

and
$$c(3) = 0.97441c(2) - 0.22313c(1) + 0.15482m(2) + 0.0939m(1)$$

Making appropriate substitutions, and applying the deadbeat conditions

$$c(2) = R_o, \text{ and } c(3) - c(2) = 0$$

along with $m(2) = R_o$, we obtain the system of equations

$$0.24476m(0) + 0.15482m(1) = R_o$$

and
$$0.03455m(0) - 0.0939m(1) = 0.12923R_o$$

Solving the equations for $m(0)$ and $m(1)$ gives

$$m(0) = 4.02058R_o$$

and
$$m(1) = 0.10289R_o$$

These results were also obtained in Example 6.3-2 using Laplace transforms.

6.5 THE STATE VARIABLE METHOD

This section presents a state variable approach which is systematic and general. The state variable approach can be used to design control algorithms for n th-order plants with multiple inputs and/or outputs, and it is suitable for computerization.

In the open-loop computer control system of Fig. 6.3-1, let the plant dynamics be described by the state and output equations

$$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{m}(t) \quad (6.5-1)$$

and
$$\mathbf{c}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{m}(t) \quad (6.5-2)$$

for the continuous case, and

$$\mathbf{x}(k+1) = \mathbf{W}\mathbf{x}(k) + \mathbf{V}\mathbf{m}(k) \quad (6.5-3)$$

and

$$\mathbf{c}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{m}(k) \quad (6.5-4)$$

for the discrete case.

For the single-input-single-output (SISO) case, $\mathbf{A}(t)$ and $\mathbf{W}(k)$ represent the $(n \times n)$ system matrices, $\mathbf{B}(t)$ and $\mathbf{V}(k)$ the $(n \times 1)$ driving or input vectors, $\mathbf{C}(t)$ and $\mathbf{C}(k)$ the $(1 \times n)$ output vectors, and $\mathbf{D}(t)$ and $\mathbf{D}(k)$ the (1×1) transmission vectors, which are usually zero.

Example 6.5-1 A First-Order Plant

As a first illustration of the state variable approach, consider the transfer function used in Example 6.3-1; namely, let

$$G_p(s) = \frac{1}{s+1}$$

It is required to manipulate the input so that the response reaches steady state in 0.5 sec.

The continuous state and output equations can be written as

$$\dot{x}_1(t) = -x_1(t) + m(t)$$

and

$$c(t) = x_1(t)$$

Assuming a sampling time of $T = 0.5$ sec, we obtain the discrete dynamic equations

$$x_1(k+1) = 0.60653x_1(k) + 0.39347m(k)$$

and

$$c(k) = x_1(k)$$

It can further be assumed, without loss of generality, that the system is initially at rest, or that

$$c(0) = x_1(0) = 0$$

Then letting $k = 0$ in Eq. (6.5-8) gives

$$c(1) = 0.39347m(0)$$

and with $k = 1$ in the same equation we get

$$c(2) = 0.60653c(1) + 0.39347m(1)$$

The deadbeat conditions dictate that

$$c(1) = R_o, \quad \text{and} \quad c'(1) = c(2) - c(1) = 0$$

Using these conditions, we obtain the manipulation values

$$m(0) = \frac{R_o}{0.39347} = 2.54149R_o$$

and

$$m(1) = \frac{R_o(1-0.60653)}{0.39347} = R_o$$

Example 6.5-2 A Second-Order Plant

For a second example using the state variable approach, consider the plant used in Example 6.3-2, that is, assume that

$$G_p(s) = \frac{2}{(s+1)(s+2)}$$

It is required to manipulate the input so that the response reaches steady state in 1 sec.

Applying direct decomposition to this transfer function results in the following set of state and output equations

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 2 \end{bmatrix} m(t)$$

and

$$c(t) = [1 \ 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

For a sampling time of $T = 0.5$ sec, the corresponding discrete dynamic equations become

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.84518 & 0.23865 \\ -0.47730 & 0.12923 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.15482 \\ 0.47730 \end{bmatrix} m(t)$$

and

$$c(k) = [1 \ 0] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

Assuming the system is at rest, and letting $k = 0$ in the state equations gives

$$\begin{bmatrix} x_1(1) \\ x_2(1) \end{bmatrix} = \begin{bmatrix} 0.15482 \\ 0.47730 \end{bmatrix} m(0)$$

Then, letting $k = 1$, and substituting the expressions for $x_1(1)$ and $x_2(1)$, we obtain

$$\begin{aligned} \begin{bmatrix} x_1(2) \\ x_2(2) \end{bmatrix} &= \begin{bmatrix} 0.84518 & 0.23865 \\ -0.47730 & 0.12923 \end{bmatrix} \begin{bmatrix} 0.15482 \\ 0.47730 \end{bmatrix} m(0) + \begin{bmatrix} 0.15482 \\ 0.47730 \end{bmatrix} m(1) \\ &= \begin{bmatrix} 0.24476 \\ -0.01221 \end{bmatrix} m(0) + \begin{bmatrix} 0.15482 \\ 0.47730 \end{bmatrix} m(1) \end{aligned}$$

The deadbeat conditions for this example imply that

$$c(2) = x_1(2) = R_o, \text{ and } c'(2) = x'_1(2) = x_2(2) = 0$$

When these conditions are applied to the present plant, we get

$$\begin{bmatrix} 0.24476 & 0.15482 \\ -0.01221 & 0.47730 \end{bmatrix} \begin{bmatrix} m(0) \\ m(1) \end{bmatrix} = \begin{bmatrix} R_o \\ 0 \end{bmatrix}$$

Solving the equations results in the manipulation values

$$m(0) = 4.02058R_o, \quad \text{and} \quad m(1) = 0.10288R_o$$

The above values agree with those obtained in Example 6.3-2.

6.5-1 The *n*th-Order Plant Case

To design deadbeat algorithms for open-loop control of an *n*th-order plant, consider again the block diagram of Fig. 6.3-1 and the state space description of a plant as denoted by Eq. (6.5-1) - Eq. (6.5-4). The discrete state transition equation of the plant can be written as

$$\mathbf{x}(k) = \mathbf{W}^k \mathbf{x}(0) + \sum_{i=1}^{k-1} \mathbf{W}^{k-i-1} \mathbf{V} m(i) \quad (6.5-5)$$

and the corresponding output equation as

$$\mathbf{c}(k) = \mathbf{C} \mathbf{W}^k \mathbf{x}(0) + \sum_{i=0}^{k-1} \mathbf{C} \mathbf{W}^{k-i-1} \mathbf{V} m(i) \quad (6.5-6)$$

where

$$\mathbf{W} = e^{\mathbf{A}T} \quad (6.5-7)$$

and

$$\mathbf{V} = \left[\int_0^T e^{\mathbf{A}^T t} dt \right] \mathbf{B} \quad (6.5-8)$$

Assuming zero initial conditions, $\mathbf{x}(0)=0$, Eq. (6.5-5) and Eq. (6.5-6) become

$$\mathbf{x}(k) = \sum_{i=0}^{k-1} \mathbf{W}^{k-i-1} \mathbf{V} m(i) \quad (6.5-9)$$

and

$$\mathbf{c}(k) = \sum_{i=0}^{k-1} \mathbf{C} \mathbf{W}^{k-i-1} \mathbf{V} m(i) \quad (6.5-10)$$

Following a vector step input of arbitrary size, denoted by R_o , we apply the deadbeat conditions. From the first deadbeat condition, the system output $\mathbf{c}(k)$ must match the step input of size R_o in *n* sampling periods, where *n* represents the order of the plant. Thus,

$$R_o = \sum_{i=0}^{n-1} C W^{n-i-1} V m(i) \quad (6.5-11)$$

or, in matrix form,

$$[C W^{n-1} V \ C W^{n-2} V \ \dots \ C V] \begin{bmatrix} m(0) \\ m(1) \\ \vdots \\ m(n-1) \end{bmatrix} = R_o \quad (6.5-12)$$

The second deadbeat condition dictates that the derivative of the output is set equal to zero at all sampling instants greater than or equal to n . That is,

$$c'(kT) = c[(k+1)T] - c(kT) = 0, \text{ for } k \geq n \quad (6.5-13)$$

The second condition guarantees that the plant output will not change at the sampling instants, after it matches the reference input for the first time. From Eq. (6.5-1), we obtain

$$x'(nT) = A x(nT) + B m(nT) \quad (6.5-14)$$

and from Eq. (6.5-5), with the plant at rest, we get

$$x(nT) = \sum_{i=0}^{n-1} W^{n-i-1} V m(i) \quad (6.5-15)$$

Setting $x'(nT)=0$ in Eq. (6.5-14), and substituting $x(nT)$ from Eq. (6.5-15) results in

$$A \left[\sum_{i=0}^{n-1} W^{n-i-1} V m(i) \right] + B m(n) = 0 \quad (6.5-16)$$

or

$$\sum_{i=0}^{n-1} A W^{n-i-1} V m(i) + B m(n) = 0 \quad (6.5-17)$$

The equation in matrix form is

$$[A W^{n-1} V \ A W^{n-2} V \ \dots \ A V \ B] \begin{bmatrix} m(0) \\ m(1) \\ \vdots \\ m(n) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (6.5-18)$$

Combining Eq. (6.5-12) and Eq. (6.5-18) gives

$$\begin{bmatrix} C W^{n-1} V & C W^{n-2} V & \dots & C V & 0 \\ A W^{n-1} V & A W^{n-2} V & \dots & A V & B \end{bmatrix} \begin{bmatrix} m(0) \\ m(1) \\ \vdots \\ m(n) \end{bmatrix} = \begin{bmatrix} R_o \\ 0 \end{bmatrix} \quad (6.5-19)$$

Eq. (6.5-19) represents the system of equations that must be solved to determine the manipulation sequence $m(i)$, for $i = 0, 1, 2, \dots, n$. Let the solution vector be denoted by

$$\mathbf{m}(i) = \mathbf{H}(i) \mathbf{R}_o \quad ; i = 0, 1, 2, \dots, n \quad (6.5-20)$$

Taking the $\mathcal{Z}$ transform of this sequence results in

$$\mathbf{M}(z) = \sum_{k=0}^{\infty} \mathbf{H}(k) \mathbf{R}_o z^{-k} \quad (6.5-21)$$

Since the plant input is constant after n sampling periods

$$\mathbf{m}(k) = \mathbf{H}(n) \quad \text{for } k \geq n \quad (6.5-22)$$

and

$$\mathbf{M}(z) = \left[\sum_{k=0}^{n-1} z^{-k} \mathbf{H}(k) + \mathbf{H}(n) \sum_{k=n}^{\infty} z^{-k} \right] \mathbf{R}_o = \left[\sum_{k=0}^{n-1} z^{-k} \mathbf{H}(k) + \mathbf{H}(n) \frac{z^{-n}}{1-z^{-1}} \right] \mathbf{R}_o \quad (6.5-23)$$

which holds for any arbitrary $\mathbf{R}_o$. Note that in Eq. (6.5-23) $\mathbf{H}$ is a matrix and, for a multiple input case, $\mathbf{M}$ is also a matrix.

Example 6.5-3 A First-Order Plant

Determine the deadbeat controller for the first-order plant used in Example 6.5-1. Assume that $T=0.5$ sec. Since the plant transfer function is

$$G_p(s) = \frac{1}{s+1}$$

Eq. (6.5-19) gives

$$\begin{bmatrix} \mathbf{C}\mathbf{V} & \mathbf{0} \\ \mathbf{A}\mathbf{V} & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{m}(0) \\ \mathbf{m}(1) \end{bmatrix} = \begin{bmatrix} \mathbf{R}_o \\ \mathbf{0} \end{bmatrix}$$

For the above plant, $\mathbf{A} = -1$, $\mathbf{B} = 1$, $\mathbf{C} = 1$, $\mathbf{W} = 0.60653$, and $\mathbf{V} = 0.39347$. Thus, upon substitution, we get

$$\begin{bmatrix} 0.39347 & 0 \\ -0.39347 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{m}(0) \\ \mathbf{m}(1) \end{bmatrix} = \begin{bmatrix} \mathbf{R}_o \\ \mathbf{0} \end{bmatrix}$$

Solving the equation gives

$$\begin{bmatrix} \mathbf{m}(0) \\ \mathbf{m}(1) \end{bmatrix} = \begin{bmatrix} 2.54149 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_o \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} 2.54149 \\ 1 \end{bmatrix} \mathbf{R}_o$$

Thus, the manipulation values are

$$\mathbf{m}(0) = 2.54149 \mathbf{R}_o, \text{ and } \mathbf{m}(1) = \mathbf{R}_o$$

The manipulation variable can be expressed in the form

$$\begin{aligned} M(z) &= [H(0)+H(1)\frac{z^{-1}}{(1-z^{-1})}]R_o = (2.54149+\frac{z^{-1}}{(1-z^{-1})})R_o \\ &= \frac{(2.54149-1.54149z^{-1})}{(1-z^{-1})}R_o \end{aligned}$$

Assuming a step-function input of arbitrary size R_o , implies the input transform

$$R(z) = \frac{R_o}{(1-z^{-1})}$$

The open-loop digital controller transfer function then becomes

$$G_c(z) = \frac{M(z)}{R(z)} = 2.54149 - 1.54149z^{-1}$$

The corresponding difference equation form of the controller is

$$m(k) = 2.54149r(k) - 1.54149r(k-1)$$

Example 6.5-4 A Second-Order Plant

Design a deadbeat controller for the following second-order plant, assuming $T=0.5$ sec.

$$G_p(s) = \frac{4}{s^2+s+4}$$

For a second-order plant, Eq. (6.5-19) becomes

$$\begin{bmatrix} CWV & CV & 0 \\ AWV & AV & B \end{bmatrix} \begin{bmatrix} m(0) \\ m(1) \\ m(2) \end{bmatrix} = \begin{bmatrix} R_o \\ 0 \\ 0 \end{bmatrix}$$

For this example, we have

$$\begin{bmatrix} 0.67770 & 0.39295 & 0 \\ -0.15538 & 1.32538 & 0 \\ -2.55542 & 2.89716 & 4 \end{bmatrix} \begin{bmatrix} m(0) \\ m(1) \\ m(2) \end{bmatrix} = \begin{bmatrix} R_o \\ 0 \\ 0 \end{bmatrix}$$

from which the solution vector is

$$\begin{bmatrix} m(0) \\ m(1) \\ m(2) \end{bmatrix} \begin{bmatrix} 1.38166 & -0.40963 & 0 \\ 0.16198 & 0.70648 & 0 \\ 1 & 0.25 & 0.25 \end{bmatrix} \begin{bmatrix} R_o \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.38166 \\ 0.16198 \\ 1 \end{bmatrix} R_o$$

The manipulation values are $m(0) = 1.38166R_o$, $m(1) = 0.16198R_o$, and $m(2) = R_o$.

which agree with the results obtained in Example 6.3-3.

From Eq. (6.5-23), we can express the manipulation variable in transform form as

$$\begin{aligned} M(z) &= [H(0)+H(1)z^{-1}+H(2)\frac{z^{-2}}{1-z^{-1}}]R_o = (1.38166+0.16198z^{-1}+\frac{z^{-2}}{1-z^{-1}})R_o \\ &= \frac{1.38166-1.21968z^{-1}+0.83802z^{-2}}{1-z^{-1}}R_o \end{aligned}$$

For a step-function input of size R_o

$$R(z) = \frac{R_o z}{z-1}$$

The transfer function of the open-loop digital controller is

$$G_c(z) = \frac{M(z)}{R(z)} = 1.38166-1.21968z^{-1}+0.83902z^{-2}$$

In difference equation form, the control algorithm is

$$m(k) = 1.38166r(k) - 1.21968r(k-1) + 0.83902r(k-2)$$

Example 6.5-4 A Third-Order Plant

As another example, determine the open-loop deadbeat controller for a third-order plant. Assume a sampling time $T=0.5$ sec. The dynamic equations for the plant are

$$\begin{bmatrix} x_1'(t) \\ x_2'(t) \\ x_3'(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -0.133 & 4.693 \\ -30 & -20 & -30 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 10 \end{bmatrix} m(k)$$

and $c(t) = x_1(t)$

Using a sampling time of $T = 0.5$ sec results in the corresponding set of discrete dynamic equations

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \end{bmatrix} = \begin{bmatrix} 0.67491 & 0.21442 & 0.03345 \\ -1.00364 & -0.0227 & 0.00264 \\ -0.04534 & -0.22513 & -0.03952 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} 0.10836 \\ 0.33455 \\ 0.01511 \end{bmatrix} m(k)$$

and $c(k) = x_1(k)$

Evaluating Eq. (6.5-19) gives

$$\begin{bmatrix} 0.07047 & 0.14536 & 0.10836 & 0 \\ -0.14348 & -0.11631 & 0.33455 & 0 \\ 0.12603 & -0.36384 & 0.02643 & 0 \\ 0.07178 & 0.38974 & -10.3952 & 10 \end{bmatrix} \begin{bmatrix} m(0) \\ m(1) \\ m(2) \\ m(3) \end{bmatrix} = \begin{bmatrix} R_o \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

The solution vector becomes

$$\begin{bmatrix} 5.32359 & -1.94147 & 2.7477 & 0 \\ 2.0619 & -0.52916 & -1.75543 & 0 \\ 3.0 & 1.9725 & 0.5681 & 0 \\ 3.0 & 2.08502 & 0.63925 & 0.1 \end{bmatrix} \begin{bmatrix} R_o \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 5.32359 \\ 2.0619 \\ 3.0 \\ 3.0 \end{bmatrix} R_o$$

From Eq. (6.5-23), and for a step-function input of arbitrary size denoted by R_o , the open-loop controller transfer function is

$$G_c(z) = H(0) + [H(1) - H(0)]z^{-1} + [H(2) - H(1)]z^{-2} + [H(3) - H(2)]z^{-3}$$

For the above case, this gives

$$M(z) = 5.32359 - 3.26169z^{-1} + 0.99381z^{-2}$$

The difference-equation form of the corresponding control algorithm then becomes

$$m(k) = 5.32359r(k) - 3.26169r(k-1) + 0.99381r(k-2)$$

Note that at steady-state, that is, as $z \rightarrow 1$, the sum of the coefficients is 3, rather than the usual 1. The reason for this is that the gain of our plant is not unity, but $1/3$. However, the control algorithm compensates for it so that a unit-step input results in a unity-step response.

Digital simulation results for the normal and the manipulated step responses of the above plant are shown in Fig. 6.5-1.

6.6 CLOSED-LOOP CONTROL: Deadbeat Algorithms

A closed-loop or feedback control system is one in which the actuating error signal is fed to the controller in order to reduce the error and drive the output of the controlled plant to a desired value. An important advantage of the closed-loop control system is that the use of feedback makes the response relatively insensitive to external disturbances, and internal variations in parameters.

A block diagram of a closed-loop control system is shown in Fig. 6.6-1. As can be deduced from the diagram, controlling a plant by a computer in a closed-loop manner implies that the computer is used as an integral part of a feedback control system. The computer samples the response of the system at certain instants of time, and compares its findings with prescribed values stored internally. Any deviation from the desired behavior initiates corrective action, as dictated by the control algorithm stored in the computer.

The main objective of this section is to design algorithms that will drive a plant from an arbitrary initial state to a desired final state in the minimum number of sampling times. Furthermore, after the output matches the input for the first time, the input and output signals become equal at all subsequent sampling instants with little or no ripple between them. In other words, we wish to design algorithms for closed-loop control that result in deadbeat response for the controlled plant or process.

TIME	OUTPUT	TIME	OUTPUT
0.0	0.0	0.0	0.0
0.25	0.03152	0.25	0.16778
0.5	0.10836	0.5	0.57688
0.75	0.18892	0.75	0.90293
1.0	0.25374	1.0	0.99735
1.25	0.29804	1.25	1.0
1.5	0.32421	1.5	1.0
1.75	0.33716	1.75	1.0
2.0	0.34177	2.0	1.0
2.25	0.34189		
2.5	0.34011		
2.75	0.33791		
3.0	0.33600		
3.25	0.33463		
3.5	0.33377		
3.75	0.33332		
4.0	0.33313		
4.25	0.33310		
4.5	0.33313		
4.75	0.33319		

Fig. 6.5-1 Digital simulation results showing (a) normal, and (b) manipulated step responses for the third-order system used in Example 6.5-4.

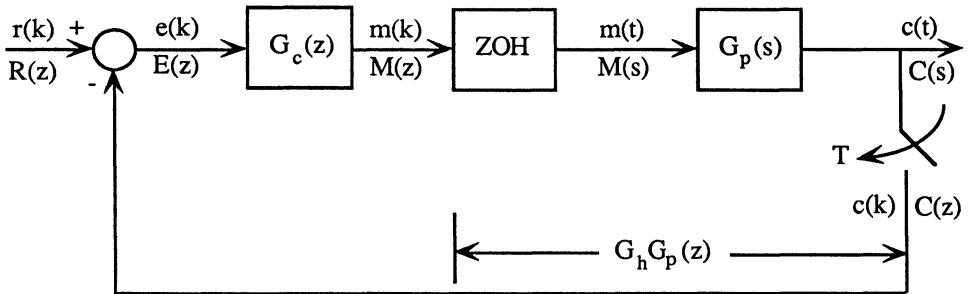


Fig. 6.6-1 Closed-loop control system.

6.6-1 The Single (Plain) Integrator Plant

As an example of closed-loop control for deadbeat response, consider a plant that behaves physically as a plain integrator. This means

$$G_p(s) = \frac{K}{s} \quad (6.6-1)$$

where K is an arbitrary constant that represents the open-loop plant gain.

Using $\mathcal{Z}$ transforms, the pulse transfer function is obtained as

$$G_h G_p(z) = (1-z^{-1})\mathcal{Z}\left(\frac{K}{s^2}\right) = \frac{KT}{z-1} = \frac{KTz^{-1}}{1-z^{-1}} \quad (6.6-2)$$

From Fig. 6.6-1,

$$\frac{C(z)}{M(z)} = \frac{KTz^{-1}}{1-z^{-1}} \quad (6.6-3)$$

The corresponding difference equation model for the plant is

$$c(k) = c(k-1) + KTm(k-1) \quad (6.6-4)$$

Applying the first deadbeat condition to the output function, and assuming the plant at rest, we obtain

$$c(1) = KTm(0) = R_o \quad (6.6-5)$$

from which

$$m(0) = \frac{R_o}{KT} \quad (6.6-6)$$

From the second deadbeat condition

$$c'(1) = c(2) - c(1) = c(1) + KTm(1) - c(1) = 0 \quad (6.6-7)$$

we get

$$m(1) = 0 \quad (6.6-8)$$

Thus, the controller output transform becomes

$$M(z) = m(0) = \frac{R_o}{KT} \quad (6.6-9)$$

To find the controller input transform, note that from Fig. 6.6-1 the error signal is given by

$$e(k) = r(k) - c(k) \quad (6.6-10)$$

Thus,

$$e(0) = r(0) - c(0) = r(0) = R_o \quad (6.6-11)$$

and

$$e(1) = r(1) - c(1) = R_o - R_o = 0 \quad (6.6-12)$$

Thus, the controller input transform $E(z)$ is

$$E(z) = \frac{R_o}{1-z^{-1}} - \frac{R_o z^{-1}}{1-z^{-1}} = R_o \quad (6.6-13)$$

and the closed-loop deadbeat controller transfer function for the plain integrator plant becomes

$$G_c(z) = \frac{M(z)}{E(z)} = \frac{1}{KT} \quad (6.6-14)$$

Example 6.6-1

Determine a closed-loop deadbeat controller for a plain integrator plant with gain $K=1$. Assume $T=0.5$ sec.

From Eq. (6.6-14), the closed-loop deadbeat controller is

$$G_c(z)=2$$

and the corresponding difference equation becomes

$$m(k)=2e(k)$$

Simulation and experimental results are shown in Fig. 6.6-2.

6.6-2 First-Order Plant Case

As another case, consider the problem of having to design a deadbeat algorithm for controlling a first-order plant described by the differential equation

$$\frac{dc(t)}{dt} + ac(t) = Km(t) \quad (6.6-15)$$

The transfer function of the plant is obtained from the above differential equation as

$$G_p(s) = \frac{C(s)}{M(s)} = \frac{K}{s+a} \quad (6.6-16)$$

The pulse transfer function is

$$G_h G_p(z) = (1-z^{-1}) \mathcal{Z} \left[\frac{K}{s(s+a)} \right] = \left(\frac{K}{a} \right) \frac{(1-e^{-aT})z^{-1}}{(1-e^{-aT}z^{-1})} \quad (6.6-17)$$

and the corresponding difference equation is

$$c(k) = e^{-aT}c(k-1) + \left(\frac{K}{a} \right) (1-e^{-aT})m(k-1) \quad (6.6-18)$$

Assuming the system at rest, the deadbeat conditions result in the equation

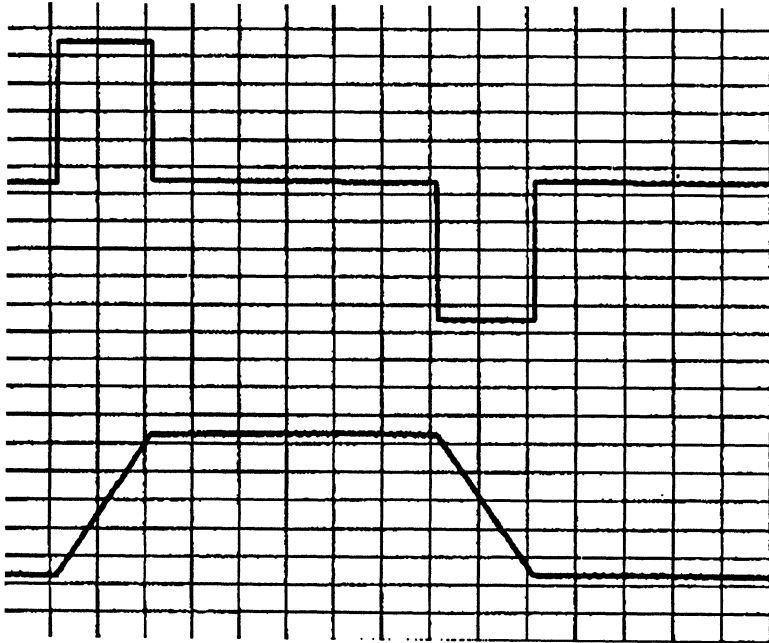
$$c(1) = \left(\frac{K}{a} \right) (1-e^{-aT})m(0) = R_o \quad (6.6-19)$$

from which

$$m(0) = \left(\frac{a}{K} \right) \frac{R_o}{(1-e^{-aT})} \quad (6.6-20)$$

SAMPLE #	INPUT	OUTPUT	ERROR	MANIPULATION
0	1	0	1	2
1	1	1	0	0
2	1	1	0	0
3	1	1	0	0
4	1	1	0	0

(a)



(b)

Fig. 6.6-2 (a) Digital simulation and (b) experimental results for deadbeat control of the plant: $G_p(s) = \frac{1}{s}$; $T=0.5$ sec.

From the derivative condition

$$c'(1) = c(2)-c(1) = e^{-aT}R_o + (\frac{K}{a})(1-e^{-aT})m(1)-R_o = 0 \tag{6.6-21}$$

from which
$$m(1) = (\frac{a}{K})R_o \tag{6.6-22}$$

Thus, the controller output transform is

$$M(z) = m(0)+\frac{m(1)z^{-1}}{(1-z^{-1})} = (\frac{a}{K})\frac{R_o(1-e^{-aT}z^{-1})}{(1-e^{-aT})(1-z^{-1})} \tag{6.6-23}$$

Like in the previous example, the controller input transform $E(z)$ is given by

$$E(z) = R_o \quad (6.6-24)$$

Thus, the digital controller becomes

$$G_c(z) = \frac{M(z)}{E(z)} = \left(\frac{a}{K}\right) \frac{(1-e^{-aT}z^{-1})}{(1-e^{-aT})(1-z^{-1})} \quad (6.6-25)$$

Example 6.6-2

Determine the deadbeat algorithm to control a first-order plant described by the transfer function

$$G_p(s) = \frac{1}{s+1}$$

Assume $T=0.5$ sec. From Eq. (6.6-25), above, the transfer function for the deadbeat controller is

$$G_c(z) = \frac{2.54149 - 1.54149z^{-1}}{1 - z^{-1}}$$

In difference equation form, the controller is

$$m(k) = m(k-1) + 2.54149e(k) - 1.54149e(k-1)$$

Results obtained by digital simulation are shown in Fig. 6.6-3. Experimental results are identical to those obtained using the open-loop control approach for the same plant, shown in Fig. 6.3-4.

SAMPLE #	INPUT	OUTPUT	ERROR	MANIPULATION
0	1	0	1	2.54149
1	1	1	0	1
2	1	1	0	1
3	1	1	0	1
4	1	1	0	1

Fig. 6.6-3 Simulation results showing the step response of the closed-loop control system of Example 6.6-2. The plant is $G_p(s) = \frac{1}{(s+1)}$, and $T=0.5$ sec.

6.6-3 Second-Order Plant Case

For a second-order plant, the output transform of the digital deadbeat controller can be expressed in the form

$$M(z) = m(0) + m(1)z^{-1} + \frac{R_o z^{-2}}{1-z^{-1}} = \frac{m(0) + [m(1) - m(0)]z^{-1} + [R_o - m(1)]z^{-2}}{1-z^{-1}} \quad (6.6-29)$$

where the manipulation values $m(0)$ and $m(1)$ were derived in Section 6.3 for the overdamped, underdamped, and critically-damped plant cases.

To determine the digital controller $G_c(z)$, we also need the input transform $E(z)$. From the error signal sequence

$$e(k) = r(k) - c(k) \quad (6.6-30)$$

we obtain

$$e(0) = r(0) - c(0) = R_o ; \text{ for } k=0 \quad (6.6-31)$$

$$e(1) = r(1) - c(1) = R_o - c(1) ; \text{ for } k=1 \quad (6.6-32)$$

$$\text{and} \quad e(k) = R_o - R_o = 0 ; \text{ for } k \geq 2 \quad (6.6-33)$$

The controller input transform then becomes

$$E(z) = R_o + [R_o - c(1)]z^{-1} \quad (6.6-34)$$

where $c(1)=c(T)$ represents the plant response evaluated at $t=T$. If the plant is overdamped, $c(1)$ can be determined from Eq. (6.3-23); if it is underdamped, $c(1)$ can be determined from Eq. (6.3-35); and in the case of a critically damped plant, $c(1)$ can be determined from Eq. (6.3-50). Thus, for the overdamped plant case

$$c(1) = \frac{m(0)(a-b+be^{-aT}-ae^{-bT})}{(a-b)} \quad (6.6-35)$$

for the underdamped case

$$c(1) = m(0)(1 - e^{-\zeta\omega_n T} \cos\omega_n \sqrt{1-\zeta^2} T - \frac{\zeta e^{-\zeta\omega_n T}}{\sqrt{1-\zeta^2}} \sin\omega_n \sqrt{1-\zeta^2} T) \quad (6.6-36)$$

and for the critically-damped case

$$c(1) = m(0)(1 - e^{-aT} - aTe^{-aT}) \quad (6.6-37)$$

Note that $m(0)$ must be evaluated for the appropriate plant case. The digital controller transfer function can now be written as

$$G_c(z) = \frac{M(z)}{E(z)} = \frac{m(0) + [m(1) - m(0)]z^{-1} + [R_o - m(1)]z^{-2}}{\{R_o + [R_o - c(1)]z^{-1}\}(1 - z^{-1})}$$

or

$$G_c(z) = \frac{m(0) + [m(1) - m(0)]z^{-1} + [R_o - m(1)]z^{-2}}{R_o - c(1)z^{-1} - [R_o - c(1)]z^{-2}} \quad (6.6-38)$$

Example 6.6-4

To illustrate the procedure, determine the deadbeat controller for the second-order overdamped plant used in Example 6.3-2. For this case,

$$m(0) = 4.02058R_o, \text{ and } m(1) = 0.10289R_o$$

From Eq. (6.6-35)

$$c(1) = \frac{4.02058R_o(1-2+2e^{-0.5}-e^{-1})}{1-2} = 0.62246R_o$$

Then, from Eq. (6.6-32), the desired closed-loop deadbeat controller transfer function becomes

$$G_c(z) = \frac{4.02058-3.91769z^{-1}+0.89711z^{-2}}{1-0.62246z^{-1}-0.37754z^{-2}}$$

The corresponding difference equation form of the controller is

$$m(k) = 0.62246m(k-1)+0.37754m(k-2)+4.02058e(k)-3.91769e(k-1)+0.89711e(k-2)$$

Example 6.6-5

As another example, consider the second-order underdamped plant used in Example 6.2-3. For this plant case,

$$m(0) = 1.38167R_o, \text{ and } m(1) = 0.16198R_o$$

From Eq. (6.6-36)

$$c(1) = 0.54292R_o$$

Thus, from Eq. (6.6-38), the closed-loop deadbeat controller becomes

$$G_c(z) = \frac{1.38166-1.21968z^{-1}+0.83802z^{-2}}{1-0.54292z^{-1}-0.45708z^{-2}}$$

The difference equation form of the control algorithm is

$$m(k) = 0.54292m(k-1)+0.45708m(k-2)+1.38166e(k)-1.21968e(k-1)+0.83802e(k-2)$$

Example 6.6-6

Determine the deadbeat controller for the second-order critically damped plant used in Example 6.3-4. For this case,

$$m(0) = 6.45919R_o, \text{ and } m(1) = -1.3762R_o$$

Then, from Eq. (6.6-37),

$$c(1) = 0.58264R_o$$

and, from Eq. (6.6-38), the closed-loop deadbeat controller is

$$G_c(z) = \frac{6.45919 - 7.83539z^{-1} + 2.3762z^{-2}}{1 - 0.58265z^{-1} - 0.41736z^{-2}}$$

In difference equation form, the control algorithm is

$$m(k) = 0.58265m(k-1) + 0.41736m(k-2) + 6.45919e(k) - 7.83539e(k-1) + 2.3762e(k-2)$$

6.6-4 The nth-Order Case

A state variable method for designing deadbeat algorithms for closed-loop control is introduced in this section. As in the open-loop control case, the state variable method is practical, general, and can be programmed on a digital computer.

The objective is to determine a general expression for the digital controller which is defined by

$$G_c(z) = \frac{M(z)}{E(z)} \quad (6.6-51)$$

where $M(z)$ and $E(z)$ denote the output and input transforms, respectively. The output transform has already been determined and is given by Eq. (6.5-23). Thus, we must find an appropriate expression for the input transform $E(z)$.

From Fig. 6.6-1, we have

$$e(k) = R_o - c(k) \quad (6.6-52)$$

From Eq. (6.5-10), the system output is

$$c(k) = \sum_{i=0}^{k-1} C W^{k-i-1} V m(i) \quad (6.6-53)$$

and the error signal sequence becomes

$$e(k) = R_o - \sum_{i=0}^{k-1} C W^{k-i-1} V m(i) \quad (6.6-54)$$

Substituting Eq. (6.5-20) into the above expression results in

$$e(k) = [I - \sum_{i=0}^{k-1} C W^{k-i-1} V H(i)] R_o \quad (6.6-55)$$

where I represents the unit matrix of appropriate dimensions. Taking the $\mathcal{Z}$ transform on both sides of the equation gives

$$E(z) = \sum_{k=0}^{\infty} e(k) z^{-k} = \sum_{k=0}^{\infty} z^{-k} [I - \sum_{i=0}^{k-1} C W^{k-i-1} V H(i)] R_o \quad (6.6-56)$$

since the coefficients $e(k)=0$ for $k>n$. Thus, the digital controller $G_c(z)$ can be expressed as

$$G_c(z) = \left\{ \sum_{k=0}^{n-1} z^{-k} \mathbf{H}(k) + \frac{\mathbf{H}(n)z^{-n}}{1-z^{-1}} \right\} \left\{ \sum_{k=0}^{n-1} z^{-k} \left[\mathbf{I} - \sum_{i=0}^{k-1} \mathbf{C} \mathbf{W}^{k-i-1} \mathbf{V} \mathbf{H}(i) \right] \right\}^{-1} \quad (6.6-57)$$

which holds for arbitrary $\mathbf{R}_0$. Note that $\mathbf{W}$ is a matrix and for the multiple-input-multiple-output (MIMO) case, $G_c(z)$ is also a matrix of transfer functions in z . For the single-input-single-output (SISO) case, however, $G_c(z)$ represents the digital controller transfer function shown in Fig. 6.6-1.

Example 6.6-6

Determine the deadbeat controller for a first-order plant described by its transfer function

$$G_p(s) = \frac{1}{s+1}$$

Assume that $T = 0.5$ sec. From Eq. (6.6-56), since $n=1$, the input transform of $G_c(z)$ is

$$E(z) = [1 - 0]R_0 = R_0$$

The controller output transform $M(z)$ for this plant was calculated in Example 6.6-6 as

$$M(z) = (2.54149 + \frac{z^{-1}}{1-z^{-1}})R_0 = \frac{(2.54149 - 1.54149z^{-1})R_0}{1-z^{-1}}$$

Thus, the desired deadbeat controller becomes

$$G_c(z) = \frac{M(z)}{E(z)} = \frac{2.54149 - 1.54149z^{-1}}{1-z^{-1}}$$

In difference equation form, the control algorithm is

$$m(k) = m(k-1) + 2.54149e(k) - 1.54149e(k-1)$$

As expected, the above result agrees with that obtained in Example 6.6-2.

Example 6.6-8

Determine the deadbeat controller for the second-order underdamped plant whose transfer function is given by

$$G_p(s) = \frac{4}{s^2 + s + 4}$$

Assume a sampling time of $T = 0.5$ sec.

For a second-order plant, that is for $n = 2$, Eq. (6.6-56) gives the controller input transform

$$E(z) = \sum_{k=0}^1 z^{-k} [\mathbf{I} - \sum_{i=0}^{k-1} \mathbf{C}\mathbf{W}^{k-i-1} \mathbf{V}\mathbf{H}(i)] \mathbf{R}_0 = \mathbf{R}_0 + z^{-1} [\mathbf{I} - \mathbf{C}\mathbf{V}\mathbf{H}(0)] \mathbf{R}_0$$

The plant transfer function given by the controller output transform $M(z)$ was obtained in Example 6.5-4 and can be written in the form

$$M(z) = \frac{\mathbf{R}_0(1.38166 - 1.21968z^{-1} + 0.83802z^{-2})}{1 - z^{-1}}$$

From Eq. (6.6-63), the controller input transform is

$$E(z) = \mathbf{R}_0[1 + (1 - 0.54292)z^{-1}] = \mathbf{R}_0(1 + 0.45708z^{-1})$$

Thus, the digital controller transfer function is given by

$$G_c(z) = \frac{1.38166 - 1.21968z^{-1} + 0.83802z^{-2}}{(1 + 0.45708z^{-1})(1 - z^{-1})} = \frac{1.38166 - 1.21968z^{-1} + 0.83802z^{-2}}{1 - 0.54292z^{-1} - 0.45708z^{-2}}$$

The corresponding difference equation form of the controller is

$$m(k) = 0.54292m(k-1) + 0.45708m(k-2) + 1.38166e(k) - 1.21868e(k-1) + 0.83802e(k-2)$$

6.7 DEADBEAT ALGORITHMS FOR RAMP INPUTS

The control algorithms designed this far respond to step function inputs in deadbeat fashion. These algorithms drive the output of an n th-order plant to steady state in nT seconds, where n is the order of the plant, and T is the sampling time in seconds. In addition, the steady-state error, that is, the difference between the reference input and controlled output will be zero for times $t \geq nT$.

The plain integrator plant K/s , for instance, will track step inputs with zero steady-state error when controlled by the algorithm $1/KT$, given by Eq. (6.6-14). However, if the input is a ramp function of slope k_1 , that is, if

$$r(kT) = k_1 kT \quad (6.7-1)$$

or, in transform form, if

$$R(z) = \frac{k_1 T z^{-1}}{(1 - z^{-1})^2} \quad (6.7-2)$$

then, from the final value theorem, we obtain the steady-state error

$$e(\infty) = \lim_{z \rightarrow 1} (1 - z^{-1})E(z) = \lim_{z \rightarrow 1} (1 - z^{-1})^2 R(z) = \lim_{z \rightarrow 1} \frac{(1 - z^{-1})^2 k_1 T z^{-1}}{(1 - z^{-1})^2} = k_1 T \quad (6.7-3)$$

If it is required that a control system track ramp inputs in deadbeat fashion, then an appropriate algorithm can be designed to accomplish this. Such an algorithm can drive the output of an n th-order plant to steady state in $(n+1)T$, rather than in nT seconds. Thus, the deadbeat conditions for ramp inputs are

$$1. \quad c(n+1) = (n + 1)k_1 \quad (6.7-4)$$

and $2. \quad c'(k+1) = c(k+2) - c(k+1) = k_1 \quad ; \text{ for } k \geq n \quad (6.7-5)$

where n represents the order of the plant and k_1 the slope of the ramp. The design procedure is illustrated next by examples.

6.7-1 The Plain Integrator Plant

Consider the problem of designing a deadbeat controller for ramp inputs when the plant is described by the transfer function

$$G_p(s) = \frac{K}{s} \quad (6.7-6)$$

where K is an arbitrary constant. The plant pulse transfer function is given by

$$\frac{C(z)}{M(z)} = G_h G_p(z) = \frac{KTz^{-1}}{1-z^{-1}} \quad (6.7-7)$$

and the corresponding difference equation model is

$$c(k) = c(k-1) + KTm(k-1) \quad (6.7-8)$$

Applying the deadbeat conditions leads to the equations

$$c(2) = 2k_1 \quad (6.7-9)$$

and $c'(2) = c(3) - c(2) = c(2) + KTm(2) - c(2) = k_1 \quad (6.7-10)$

From Eq. (6.7-10), the manipulation value $m(2)$ is given by

$$m(2) = \frac{k_1}{KT} \quad (6.7-11)$$

The derivative condition implies that

$$m(k+1) = m(k) \quad , \text{ for } k \geq 2 \quad (6.7-12)$$

Assuming the plant is at rest, $c(0)=0$, and since for a ramp function input $r(0)=0$, it follows that $c(1)=0$ and, from Eq. (6.7-8), $m(0)=0$. The manipulation $m(1)$ can be obtained by substituting $n=2$ into Eq. (6.7-8). This gives

$$m(1) = \frac{2k_1}{KT} \quad (6.7-13)$$

The controller output transform $M(z)$ is

$$M(z) = m(1)z^{-1} + \frac{m(2)z^{-2}}{1-z^{-1}} = \frac{z^{-1}\{m(1)+[m(2)-m(1)]z^{-1}\}}{1-z^{-1}} \quad (6.7-14)$$

Using $m(1)$ and $m(2)$ we get

$$M(z) = \left(\frac{k_1}{KT}\right) \left(\frac{z^{-1}}{1-z^{-1}}\right) (2-z^{-1}) \quad (6.7-15)$$

Since $e(0)=0$ and $e(k)=0$ for $k \geq 2$, the controller input transform $E(z)$ is

$$E(z) = e(0) + e(1)z^{-1} = e(1)z^{-1} = [r(1)-c(1)]z^{-1} = k_1z^{-1} \quad (6.7-16)$$

The deadbeat controller for ramp inputs then becomes

$$G_c(z) = \frac{M(z)}{E(z)} = \frac{1}{KT} \frac{2-z^{-1}}{1-z^{-1}} = \frac{1}{KT} \left(1 + \frac{1}{1-z^{-1}}\right) \quad (6.7-17)$$

This is a Proportional-plus-integral (PI) control algorithm. The corresponding difference equation is

$$m(k) = m(k-1) + \frac{[2e(k) - e(k-1)]}{KT} \quad (6.7-18)$$

The closed-loop transfer function of the control system is given by

$$\frac{C(z)}{R(z)} = \frac{G_c(z)G_hG_p(z)}{1+G_c(z)G_hG_p(z)} = 2z^{-1} - z^{-2} \quad (6.7-19)$$

and in difference equation form

$$c(k) = 2r(k-1) - r(k-2) \quad (6.7-20)$$

Thus, for a ramp input of slope 1 the control system behaves as shown in Table 6.7-1.

If the input to the control system is a unit step, instead of a unit ramp, then from Eq. (6.7-20),

$$c(1) = 2r(0) = 2 \quad (6.7-21)$$

which signifies excessive overshoot that may not be acceptable. The behavior of this control system to step inputs is shown in Table 6.7-2.

Table 6.7-1. Ramp input

k	r(k)	e(k)	m(k)	c(k)
0	0	0	0	0
1	1	1	2/KT	0
2	2	0	1/KT	2
3	3	0	1/KT	3
4	4	0	1/KT	4
5	5	0	1/KT	5

Table 6.7-2. Step input

k	r(k)	e(k)	m(k)	c(k)
0	1	1	2/KT	0
1	1	-1	-1/KT	2
2	1	0	0	1
3	1	0	0	1
4	1	0	0	1
5	1	0	0	1

6.7-2 First-Order Plant Case

Consider the problem of designing a deadbeat controller for ramp inputs when the plant is given by

$$G_p(s) = \frac{K}{s+a} \quad (6.7-22)$$

where K and ' a ' are arbitrary constants. The corresponding pulse transfer function and difference equation models are given by Eq. (6.6-17) and Eq. (6.6-18), respectively.

Letting $n=2$ in Eq. (6.6-18), and applying the first deadbeat condition for ramp inputs, we obtain

$$c(2) = \frac{K}{a} (1-e^{-aT})m(1) = 2k_1 \quad (6.7-23)$$

from which

$$m(1) = \frac{2k_1 a}{K(1-e^{-aT})} \quad (6.7-24)$$

For a first-order plant, the derivative deadbeat condition for ramp inputs is

$$c'(k+1) = c(k+2) - c(k+1) = k_1 ; \text{ for } k \geq 1 \quad (6.7-25)$$

Substituting $c(k)$ from Eq. (6.6-18) into Eq. (6.7-25) results in the relationship

$$e^{-aT}[c(k+1)-c(k)] + \frac{K}{a} (1-e^{-aT})[m(k+1)-m(k)] = k_1 ; \text{ for } k \geq 1 \quad (6.7-26)$$

Eq. (6.7-26) can be used to calculate subsequent manipulation values. Thus,

$$m(2) = \left(\frac{k_1 a}{K}\right) \left(\frac{3-2e^{-aT}}{1-e^{-aT}}\right) \quad (6.7-27)$$

$$m(3) = \left(\frac{k_1 a}{K}\right) \left(\frac{4-3e^{-aT}}{1-e^{-aT}}\right) \quad (6.7-28)$$

and so forth. The controller output transform then becomes

$$M(z) = m(1)z^{-1} + m(2)z^{-2} + m(3)z^{-3} + m(4)z^{-4} + \dots$$

$$\begin{aligned} &= \frac{k_1 a}{K(1-e^{-aT})} [2z^{-1} + (3-2e^{-aT})z^{-2} + (4-3e^{-aT})z^{-3} + (5-4e^{-aT})z^{-4} + \dots] \\ &= \frac{k_1 a}{K(1-e^{-aT})} [(2z^{-1} + 3z^{-2} + 4z^{-3} + \dots) - e^{-aT}z^{-2}(2 + 3z^{-1} + 4z^{-2} + \dots)] \\ &= \frac{k_1 a}{K(1-e^{-aT})} \left[\frac{2-z^{-1}}{(1-z^{-1})^2} - \frac{e^{-aT}z^{-1}(2-z^{-1})}{(1-z^{-1})^2} \right] = \frac{k_1 a}{K(1-e^{-aT})} \frac{(2-z^{-1})(1-e^{-aT}z^{-1})}{(1-z^{-1})^2} \end{aligned} \quad (6.7-29)$$

The controller input transform $E(z)$ is given by Eq. (6.7-16). Therefore, for the first-order plant, the deadbeat controller for ramp inputs is

$$G_c(z) = \frac{a}{K(1-e^{-aT})} \frac{(2-z^{-1})(1-e^{-aT}z^{-1})}{(1-z^{-1})^2} \quad (6.7-30)$$

From the steady-state error criteria, it can be seen that the deadbeat controller of Eq. (6.7-30) has converted the control system into a Type 2 system. The result was the same in the plain integrator case considered earlier. However, in that case the controller contributed only one of the integrators, because the plant itself contributed the other. The open-loop transfer function for both control systems is given by

$$G_c(z)G_hG_p(z) = \frac{2z^{-1}-z^{-2}}{(1-z^{-1})^2} \quad (6.7-31)$$

Example 6.7-1

As an example of a plain integrator plant required to track ramp inputs in a deadbeat manner, consider the transfer function

$$G_p(s) = \frac{10}{s}$$

and assume a sampling time of $T=0.1$ sec. Then, from Eq. (6.7-17), the controller is

$$G_c(z) = \frac{2-z^{-1}}{1-z^{-1}}$$

Example 6.7-2

As another example, determine the deadbeat controller for ramp inputs if

$$G_p(s) = \frac{10}{s+5}$$

Assume that $T=0.1$ sec. From eq. (6.7-30), the appropriate deadbeat controller for ramp inputs is

$$G_c(z) = \frac{2.54149-2.81225z^{-1}+0.77075z^{-2}}{(1-z^{-1})^2}$$

Example 6.7-3

Consider a second-order plant whose transfer function is

$$G_p(s) = \frac{1600}{s^2+56.5685s+1600}$$

It is required that the control system track ramp inputs in a deadbeat fashion using a sampling time of $T=0.01$ sec. The respective pulse transfer function and difference equation models are

$$G_h G_p(z) = \frac{0.06597z^{-1} + 0.05461z^{-2}}{1 - 1.44739z^{-1} + 0.56797z^{-2}}$$

and $c(k) = 1.44739c(k-1) - 0.56797c(k-2) + 0.06597m(k-1) + 0.05461m(k-2)$

The appropriate deadbeat conditions for a ramp input of slope k_1 are

$$c(k+1) = r(k+1) = (k+1)k_1 ; \text{ for } k=2$$

and $c'(k+1) = c(k+2) - c(k+1) = k_1 ; \text{ for } k \geq 2$

Applying the above deadbeat conditions, and solving the resulting equations for $m(1)$, $m(2)$, and $m(3)$ we obtain the manipulation values

$$m(1) = 20.3426, m(2) = -0.80684, \text{ and } m(3) = 7.0358$$

The controller output transform is given by

$$\begin{aligned} M(z) &= m(1)z^{-1} + m(2)z^{-2} + m(3)z^{-3} \frac{1}{1-z^{-1}} + z^{-4} \frac{1}{(1-z^{-1})^2} \\ &= (k_1 z^{-1}) \frac{20.3426 - 41.492z^{-1} + 28.992z^{-2} - 6.8426z^{-3}}{(1-z^{-1})^2} \end{aligned}$$

and the controller input transform is

$$\begin{aligned} E(z) &= e(1)z^{-1} + e(2)z^{-2} = k_1 z^{-1} + k_1 [r(2) - c(2)]z^{-2} \\ &= k_1 z^{-1} [1 + (2 - 1.342)z^{-1}] = k_1 z^{-1} (1 + 0.658z^{-1}) \end{aligned}$$

Thus, the required deadbeat controller for ramp inputs is

$$G_c(z) = \frac{20.3426 - 41.492z^{-1} + 28.992z^{-2} - 6.8426z^{-3}}{(1-z^{-1})^2 (1 + 0.658z^{-1})}$$

It can be shown that the corresponding difference equation model of the control system is

$$c(k) = 1.342r(k-1) + 0.316r(k-2) - 0.658r(k-3)$$

which can be used to verify that the system tracks ramp inputs with zero steady-state error. The transient values of the response are

$$c(0) = 0, c(1) = 0, \text{ and } c(2) = 1.342k_1$$

Note that if the input is a step of size R_0 , the transient values of the step response are

$$c(0) = 0, c(1) = 1.342R_0, \text{ and } c(2) = 1.658R_0$$

which imply rather excessive overshoot. However, the steady-state error is zero for both ramp and step input cases.

Example 6.7-4

Consider the plant transfer function

$$G_p(s) = \frac{10}{s(s+5)}$$

which is a second-order system with one plain integrator. Find the deadbeat controller assuming $T = 0.1$ sec. The pulse transfer function is given by

$$G_h G_p(z) = \frac{0.04261z^{-1} + 0.03608z^{-2}}{1 - 1.60653z^{-1} + 0.60653z^{-2}}$$

and the corresponding difference equation is

$$c(k) = 1.60653c(k-1) - 0.60653c(k-2) + 0.04261m(k-1) + 0.03608m(k-2)$$

The first deadbeat condition for a second-order plant leads to the equation

$$0.10453m(1) + 0.04261m(2) = 3k_1$$

where k_1 is the slope of the ramp input. Applying the second deadbeat condition

$$c'(3) = c(4) - c(3) = k_1$$

we obtain the equation

$$-0.02584m(1) + 0.03608m(2) + 0.04261m(3) = -0.81959k_1$$

Applying the second deadbeat condition again,

$$c'(4) = c(5) - c(4) = k_1$$

we obtain the equation

$$0.04261m(4) + 0.03608m(3) = 0.39347k_1$$

from which

$$m(3) = m(4) = 5k_1$$

By substituting and solving the resulting equation, we obtain the manipulation values

$$m(1) = 31.245, \text{ and } m(2) = -6.2436$$

The controller output transform is

$$\begin{aligned} M(z) &= k_1 z^{-1} [m(1) + m(2)z^{-1} + m(3) \frac{z^{-2}}{1-z^{-1}}] \\ &= \frac{k_1 z^{-1} \{m(1) + [m(2) - m(1)]z^{-1} + [m(3) - m(2)]z^{-2}\}}{1-z^{-1}} \end{aligned}$$

and the controller input transform is

$$E(z) = k_1 z^{-1} \{1 + [r(2) - c(2)]z^{-1}\} = k_1 z^{-1} [1 + 0.66865z^{-1}]$$

Therefore, the deadbeat controller for ramp inputs becomes

$$G_c(z) = \frac{31.245 - 37.4886z^{-1} + 11.2436z^{-2}}{(1 - z^{-1})(1 + 0.66865z^{-1})}$$

6.8 ALGORITHMS FOR CONTROLLING DISTURBANCES

The design methods developed so far deal with algorithms that control reference input or setpoint changes. However, such control laws may or may not perform well when, in addition to the reference input, a **disturbance (noise)** $d(t)$ enters the control loop as shown in Fig. 6.8-1.

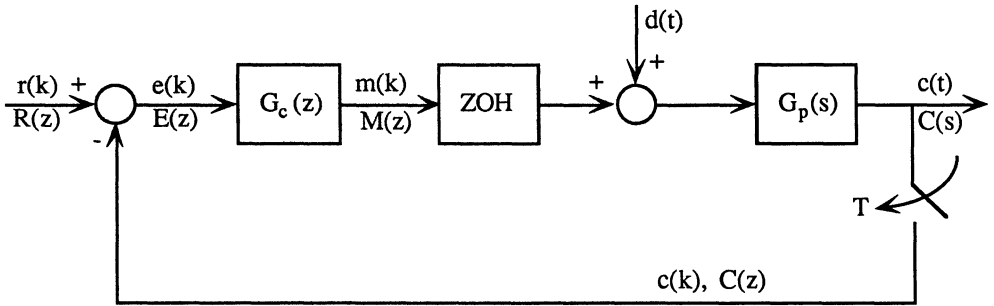


Fig. 6.8-1 Control loop with disturbance present.

The control method developed in this section assumes that the disturbance occurs as a step and that it enters the control loop at the sampling instants. Unlike the reference input case, the point in time relative to the sampling instant when a disturbance enters the control loop cannot be controlled. Even though this may affect the performance of the system, it has been found to be rather unimportant for most cases.

The first objective of this section is to design algorithms for controlling disturbances in a deadbeat manner. In other words, to formulate controllers that will restore the "disturbed" output of a plant to its "pre-disturbed" state in the minimum number of sampling times. Furthermore, after the response matches its pre-disturbed value for the first time, it is held at this value for all sampling instants, exhibiting little or no ripple between them. The second interrelated objective is to develop a general control scheme that tracks reference inputs and/or rejects disturbances, both in deadbeat manner.

For an n th-order plant, the proposed disturbance control scheme implies that it will take $(n+1)$ sampling times to reject the effect of a disturbance in deadbeat manner, that is, to restore the response of the system in a time-optimal sense. For a first-order plant, the controller becomes aware of the disturbance one sampling time after it enters the control loop, and it takes the controller one additional sampling time to prescribe the appropriate

corrective action. For a second-order plant, the disturbance is detected in the first sampling interval, and it takes the controller two additional sampling times to restore the response, a total of three sampling times since the disturbance entered the control loop.

As stated earlier, control algorithms designed for setpoint changes, in many cases perform well for disturbance changes also. The method developed is for algorithms designed specifically to control disturbances in a deadbeat manner. The control system used for the algorithm design is represented by the block diagram shown in Fig. 6.8-2. This diagram is obtained from Fig. 6.8-1 by assuming that the reference input is zero. The disturbance controller is denoted by $G_d(z)$, and the disturbance input $d(k)$ is often viewed as a load change.

For an n th-order plant, the conditions for designing a deadbeat algorithm to remove external disturbances are given by

$$1. \quad c(n+1) = 0, \text{ and} \quad (6.8-1)$$

$$2. \quad c'(k+1) = c(k+2) - c(k+1) = 0 \quad ; \text{ for } k \geq n \quad (6.8-2)$$

The first condition implies that the system response is forced back to its undisturbed state, in this case zero, in $(n+1)$ sampling intervals. The second or derivative condition ensures that the system response does not ripple or change after it meets the first condition.

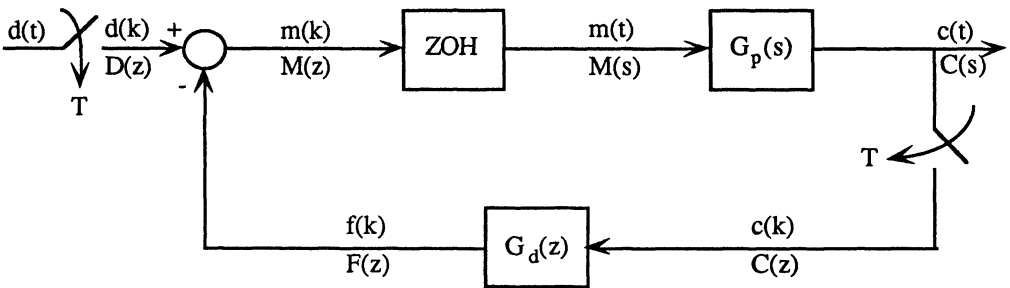


Fig. 6.8-2. Control loop subject to disturbance inputs only.

6.8-1 The z Transform Method

The z transform method for designing deadbeat controllers to remove external disturbances is illustrated for the following general cases.

The First-Order Plant. Let a first-order plant be represented by the transfer function

$$G_p(s) = \frac{K}{s+a} \quad (6.8-3)$$

From Eq. (6.6-17), the pulse transfer function is

$$G_h G_p(z) = \frac{C(z)}{M(z)} = \left(\frac{k}{a} \right) \frac{(1-e^{-aT})z^{-1}}{1-e^{-aT}z^{-1}} \quad (6.8-4)$$

and the corresponding difference equation is

$$c(k) = e^{-aT}c(k-1) + \left(\frac{K}{a}\right)(1-e^{-aT})m(k-1) \quad (6.8-5)$$

Assuming the plant at rest, the first deadbeat condition specified by Eq. (6.8-1) gives

$$\begin{aligned} c(2) &= e^{-aT}c(1) + \left(\frac{K}{a}\right)(1-e^{-aT})m(1) \\ &= e^{-aT}\left(\frac{K}{a}\right)(1-e^{-aT})m(0) + \left(\frac{K}{a}\right)(1-e^{-aT})m(1) = 0 \end{aligned} \quad (6.8-6)$$

or

$$e^{-aT}m(0) + m(1) = 0 \quad (6.8-7)$$

The derivative condition described by Eq. (6.8-2) implies that $m(2)=m(3)=\dots=0$. Next, for a step function disturbance input of size D_o

$$d(k) = D_o u(k) \quad (6.8-8)$$

which in transform form is

$$D(z) = \frac{D_o}{1-z^{-1}} \quad (6.8-9)$$

From Fig. 6.8-2, and with $f(0)=0$

$$m(0) = d(0) - f(0) = D_o - 0 = D_o \quad (6.8-10)$$

From Eq. (6.8-7), we obtain the manipulation value

$$m(1) = -e^{-aT}m(0) = -e^{-aT}D_o \quad (6.8-11)$$

It follows then that the manipulation transform $M(z)$ is given by

$$M(z) = m(0) + m(1)z^{-1} = D_o(1-e^{-aT}z^{-1}) \quad (6.8-12)$$

Thus, the noise controller output transform becomes

$$F(z) = D(z) - M(z) = D_o \left[\frac{1}{1-z^{-1}} - (1-e^{-aT}z^{-1}) \right] = D_o \frac{(1+e^{-aT})z^{-1} - e^{-aT}z^{-2}}{1-z^{-1}} \quad (6.8-13)$$

The controller input transform is given by

$$C(z) = c(0) + c(1)z^{-1} = c(1)z^{-1} = \left(\frac{K}{a}\right)(1-e^{-aT})m(0)z^{-1} = D_o \left(\frac{K}{a}\right)(1-e^{-aT})z^{-1} \quad (6.8-14)$$

and the desired noise/disturbance deadbeat controller becomes

$$G_d(z) = \left(\frac{a}{K}\right) \frac{(1+e^{-aT} - e^{-aT}z^{-1})}{(1-e^{-aT})(1-z^{-1})} \quad (6.8-15)$$

From Fig. 6.8-2, we have

$$\frac{C(z)}{D(z)} = \frac{G_h G_p(z)}{1 + G_h G_p(z) G_d(z)} = \left(\frac{K}{a} \right) (1 - e^{-aT})(z^{-1} - z^{-2}) \quad (6.8-16)$$

Hence, the control system difference equation model is

$$c(k) = \left(\frac{K}{a} \right) (1 - e^{-aT}) [d(k-1) - d(k-2)] \quad (6.8-17)$$

Eq. (6.8-17) verifies that for a step disturbance input the response goes to zero in two sampling times.

The Plain Integrator Plant. As another example, consider the plant transfer function

$$G_p(s) = \frac{K}{s} \quad (6.8-18)$$

for which the corresponding pulse transfer function is given by Eq. (6.5-2) as

$$G_h G_p(z) = \frac{KTz^{-1}}{1 - z^{-1}} \quad (6.8-19)$$

The corresponding difference equation is

$$c(k) = c(k-1) + KTm(k-1) \quad (6.8-20)$$

Assuming the plant is at rest, the first deadbeat condition of Eq. (6.8-1) implies

$$c(2) = c(1) + KTm(1) = KTm(0) + KTm(1) = 0 \quad (6.8-21)$$

from which we obtain

$$m(0) + m(1) = 0, \text{ or } m(1) = -m(0) \quad (6.8-22)$$

Since

$$m(k) = d(k) - f(k), \text{ and } f(0) = 0 \quad (6.8-23)$$

$$m(0) = d(0) = D_o \quad (6.8-24)$$

where D_o denotes the size of the disturbance step input. Thus, from Eq. (6.8-22),

$$m(1) = -D_o \quad (6.8-25)$$

and the controller output transform becomes

$$F(z) = f(0) + f(1)z^{-1} + f(2)z^{-2} \frac{1}{1 - z^{-1}} = f(1)z^{-1} + f(2)z^{-2} \frac{1}{1 - z^{-1}}$$

$$= (D_o + D_o)z^{-1} + D_o \frac{z^{-2}}{1-z^{-1}} = D_o z^{-1} \frac{(2-z^{-1})}{(1-z^{-1})} \quad (6.8-26)$$

The input transform is given by

$$C(z) = c(0) + c(1)z^{-1} = c(1)z^{-1} = KTD_o z^{-1} \quad (6.8-27)$$

and the noise controller becomes

$$G_d(z) = \frac{1}{KT} \frac{2-z^{-1}}{1-z^{-1}} = \frac{1}{KT} \left(1 + \frac{1}{1-z^{-1}}\right) \quad (6.8-28)$$

This is a proportional-plus-integral (PI) control algorithm. The corresponding difference equation form of the PI controller is

$$f(k) = f(k-1) + \frac{[2c(k) - c(k-1)]}{KT} \quad (6.8-29)$$

Example 6.8-1

Design an algorithm to reject disturbances in deadbeat manner for the first-order plant

$$G_p(s) = \frac{10}{s+5}$$

assuming a sampling time of $T = 0.1$ sec.

From Eq. (6.8-15), the appropriate disturbance deadbeat controller is given by the transfer function

$$G_d(z) = \frac{2.04149 - 0.77075z^{-1}}{1-z^{-1}}$$

The corresponding difference equation is

$$f(k) = f(k-1) + 2.04149c(k) - 0.77075c(k-1)$$

Example 6.8-2

Design a deadbeat disturbance controller for the plain integrator plant

$$G_p(s) = \frac{10}{s}$$

assuming $T=0.1$ sec.

From Eq. (6.8-28), the disturbance controller transfer function becomes

$$G_d(z) = \frac{2-z^{-1}}{1-z^{-1}}$$

and in difference equation form

$$f(k) = f(k-1) + 2c(k) - c(k-1)$$

6.8-2 The State Variable Method

The design of algorithms for rejecting disturbances in deadbeat manner can be generalized using state variables. For an n th-order, multiple-input and/or multiple-output case, algorithms can be developed systematically and the method can be programmed on a digital computer.

Consider again the control system block diagram shown in Fig. 6.8-2. The discrete state transition equation for the plant can be written as

$$\mathbf{x}(k) = \mathbf{W}^k \mathbf{x}(0) + \sum_{i=0}^{k-1} \mathbf{W}^{k-i-1} \mathbf{V} \mathbf{m}(i) \quad (6.8-30)$$

and the corresponding output equation as

$$\mathbf{c}(k) = \mathbf{C} \mathbf{W}^k \mathbf{x}(0) + \sum_{i=0}^{k-1} \mathbf{C} \mathbf{W}^{k-i-1} \mathbf{V} \mathbf{m}(i) \quad (6.8-31)$$

where $\mathbf{W}$ and $\mathbf{V}$ are the discrete system and input matrices, respectively.

Assuming zero initial conditions, that is, $\mathbf{x}(0)=0$, Eq. (6.8-30) and Eq. (6.8-31) can be written as

$$\mathbf{x}(k) = \sum_{i=0}^{k-1} \mathbf{W}^{k-i-1} \mathbf{V} \mathbf{m}(i) \quad (6.8-32)$$

and

$$\mathbf{c}(k) = \sum_{i=0}^{k-1} \mathbf{C} \mathbf{W}^{k-i-1} \mathbf{V} \mathbf{m}(i) \quad (6.8-33)$$

Following a vector step input of arbitrary size, denoted by $\mathbf{D}_0$, the first deadbeat condition implies that the disturbance controller $G_d(z)$ drive the disturbed output of the plant back to zero in minimum time. For the n th-order plant case, the minimum time is $(n+1)$ sampling periods, because the first sampling period is used to detect the presence of the disturbance. Thus,

$$\mathbf{c}(k) = \sum_{i=0}^n \mathbf{C} \mathbf{W}^{n-i} \mathbf{V} \mathbf{m}(i) = 0 \quad (6.8-34)$$

or, in matrix form,

$$[\mathbf{C} \mathbf{W}^{n-2} \mathbf{V} \dots \mathbf{C} \mathbf{V} \mathbf{0}] \begin{bmatrix} \mathbf{m}(1) \\ \mathbf{m}(2) \\ \vdots \\ \mathbf{m}(n+1) \end{bmatrix} = -[\mathbf{C} \mathbf{W}^n \mathbf{V}] \mathbf{m}(0) \quad (6.8-35)$$

The second deadbeat condition for $G_d(z)$ is to keep the response from changing or rippling after it becomes zero for the first time. This condition implies that the derivative of the output, $c'(k)$, be set equal to zero for $k \geq n+1$. From Eq. (6.5-1),

$$\mathbf{x}'(k) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{m}(k) \quad (6.8-36)$$

Setting $\mathbf{x}'(k)=0$, and substituting $\mathbf{x}(k)$ from Eq. (6.8-32) into Eq. (6.8-36), we get

$$\sum_{i=0}^n \mathbf{A}\mathbf{W}^{n-i}\mathbf{V}\mathbf{m}(i) + \mathbf{B}\mathbf{m}(n+1) = 0 \quad (6.8-37)$$

which in matrix form becomes

$$[\mathbf{A}\mathbf{W}^{n-2}\mathbf{V} \dots \mathbf{A}\mathbf{V} \ \mathbf{B}] \begin{bmatrix} m(1) \\ m(2) \\ \vdots \\ m(n+1) \end{bmatrix} = -[\mathbf{A}\mathbf{W}^n\mathbf{V}]m(0) \quad (6.8-38)$$

Combining Eq. (6.8-35) and Eq. (6.8-38) results in the system of equations

$$\begin{bmatrix} \mathbf{C}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{C}\mathbf{V} & \mathbf{0} \\ \mathbf{A}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{A}\mathbf{V} & \mathbf{B} \end{bmatrix} \begin{bmatrix} m(1) \\ m(2) \\ \vdots \\ m(n+1) \end{bmatrix} = - \begin{bmatrix} \mathbf{C}\mathbf{W}^n\mathbf{V} \\ \mathbf{A}\mathbf{W}^n\mathbf{V} \end{bmatrix} m(0) \quad (6.8-39)$$

From Fig. 6.8-2, $m(k)=d(k)-f(k)$. Then, Eq. (6.8-39) can be rewritten as

$$\begin{bmatrix} \mathbf{C}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{C}\mathbf{V} & \mathbf{0} \\ \mathbf{A}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{A}\mathbf{V} & \mathbf{B} \end{bmatrix} \begin{bmatrix} D_o - f(1) \\ D_o - f(2) \\ \vdots \\ D_o - f(n+1) \end{bmatrix} = - \begin{bmatrix} \mathbf{C}\mathbf{W}^n\mathbf{V} \\ \mathbf{A}\mathbf{W}^n\mathbf{V} \end{bmatrix} [D_o - f(0)] \quad (6.8-40)$$

Letting $f(0) = 0$ in Eq. (6.8-40), and rearranging, we obtain

$$\begin{bmatrix} f(1) \\ f(2) \\ \vdots \\ f(n+1) \end{bmatrix} = \begin{bmatrix} \mathbf{C}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{C}\mathbf{V} & \mathbf{0} \\ \mathbf{A}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{A}\mathbf{V} & \mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{C}\mathbf{W}^n\mathbf{V} \\ \mathbf{A}\mathbf{W}^n\mathbf{V} \end{bmatrix} D_o + \begin{bmatrix} D_o \\ \vdots \\ D_o \end{bmatrix} \quad (6.8-41)$$

Denoting

$$\mathbf{Q} = \begin{bmatrix} \mathbf{C}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{C}\mathbf{V} & \mathbf{0} \\ \mathbf{A}\mathbf{W}^{n-2}\mathbf{V} & \dots & \mathbf{A}\mathbf{V} & \mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{C}\mathbf{W}^n\mathbf{V} \\ \mathbf{A}\mathbf{W}^n\mathbf{V} \end{bmatrix} \quad (6.8-42)$$

in Eq. (6.8-41), and since

$$f(0) = 0 = -D_o + D_o \quad (6.8-43)$$

we obtain the system of equations

$$\begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ f(n+1) \end{bmatrix} = \mathbf{H}\mathbf{D}_o + \begin{bmatrix} \mathbf{D}_o \\ \mathbf{D}_o \\ \vdots \\ \mathbf{D}_o \end{bmatrix} \quad (6.8-44)$$

where $\mathbf{H} = \begin{bmatrix} -1 \\ \mathbf{Q} \end{bmatrix}$ (6.8-45)

Since the output of the noise controller is constant after $(n+1)$ sampling times, its output sequence becomes

$$\mathbf{f}(k) = \mathbf{D}_o + \mathbf{H}(k)\mathbf{D}_o = [1 + \mathbf{H}(k)]\mathbf{D}_o \quad (6.8-46)$$

When $\mathcal{Z}$ transformed, Eq. (6.8-46) becomes

$$\mathbf{F}(z) = \mathbf{D}_o \sum_{k=0}^{\infty} [1 + \mathbf{H}(k)]z^{-k} = \mathbf{D}_o \left\{ \sum_{k=0}^n [1 + \mathbf{H}(k)]z^{-k} + [1 + \mathbf{H}(n+1)] \frac{z^{-n-1}}{1-z^{-1}} \right\} \quad (6.8-47)$$

The input sequence of the noise controller is given by

$$\mathbf{c}(k) = \sum_{i=0}^{k-1} \mathbf{C}\mathbf{W}^{k-i-1} \mathbf{V}\mathbf{m}(i) \quad (6.8-48)$$

Using Eq. (6.8-44), we get

$$\mathbf{c}(k) = \sum_{i=0}^{k-1} \mathbf{C}\mathbf{W}^{k-i-1} \mathbf{V}[\mathbf{D}_o - \mathbf{f}(i)] = -\mathbf{D}_o \sum_{i=0}^{k-1} \mathbf{C}\mathbf{W}^{k-i-1} \mathbf{V}\mathbf{H}(i) \quad (6.8-49)$$

Taking the $\mathcal{Z}$ transform of both sides of Eq. (6.8-49) gives

$$\mathbf{C}(z) = -\mathbf{D}_o \sum_{k=0}^n z^{-k} \sum_{i=0}^{k-1} \mathbf{C}\mathbf{W}^{k-i-1} \mathbf{V}\mathbf{H}(i) \quad (6.8-50)$$

Thus the transfer function of the digital controller designed to remove disturbances in deadbeat fashion is

$$\mathbf{G}_d(z) = \left\{ \sum_{k=0}^n [1 + \mathbf{H}(k)]z^{-k} + [1 + \mathbf{H}(n+1)] \frac{z^{-n-1}}{1-z^{-1}} \right\} \left\{ - \sum_{k=0}^n z^{-k} \sum_{i=0}^{k-1} \mathbf{C}\mathbf{W}^{k-i-1} \mathbf{V}\mathbf{H}(i) \right\}^{-1} \quad (6.8-51)$$

For the n th-order plant case, Eq. (6.8-51) is a matrix relationship for arbitrary step disturbance inputs. $\mathbf{H}$ is a matrix, and $\mathbf{G}_d(z)$ is also a matrix of transfer functions for plants with multiple inputs and outputs. For the single-input-single-output case, however, $G_d(z)$ represents the noise controller shown in Fig. 6.8-2.

Example 6.8-3

Use the state variable method to design a disturbance deadbeat controller for the first-order plant

$$G_p(s) = \frac{10}{s+5}$$

assuming a sampling time $T=0.1$ sec.

It is easy to show that for this case the relevant matrices are

$$\mathbf{A} = -5, \mathbf{B} = 10, \mathbf{C} = 1, \mathbf{W} = 0.60653, \text{ and } \mathbf{V} = 0.78694$$

For a first-order plant, $n=1$, Eq. (6.8-42) gives

$$\mathbf{Q} = \begin{bmatrix} \mathbf{CV} & \mathbf{0} \\ \mathbf{A} & \mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{CWV} \\ \mathbf{AWV} \end{bmatrix}$$

which for the present example becomes

$$\begin{aligned} \mathbf{Q} &= \begin{bmatrix} 0.78694 & 0 \\ -3.9347 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 0.4773 \\ -2.3865 \end{bmatrix} \\ &= \begin{bmatrix} 1.2704 & 0 \\ 0.5 & 0.1 \end{bmatrix} \begin{bmatrix} 0.4773 \\ -2.3865 \end{bmatrix} = \begin{bmatrix} 0.6065 \\ 0 \end{bmatrix} \end{aligned}$$

From Eq. (6.8-45),

$$\mathbf{H} = \begin{bmatrix} -1 \\ 0.6065 \\ 0 \end{bmatrix}$$

For a first-order plant, the noise controller can be written in the form

$$\begin{aligned} G_d(z) &= \frac{[1+\mathbf{H}(0)]+[1+\mathbf{H}(1)]z^{-1}+\frac{[1+\mathbf{H}(2)]z^{-2}}{1-z^{-1}}}{-z^{-1}\mathbf{CVH}(0)} \\ &= \frac{[1+\mathbf{H}(0)]+[\mathbf{H}(1)-\mathbf{H}(0)]z^{-1}+[\mathbf{H}(2)-\mathbf{H}(1)]z^{-2}}{-z^{-1}(1-z^{-1})\mathbf{CVH}(0)} \end{aligned}$$

Substituting the given parameter values, results in the disturbance controller transfer function

$$G_d(z) = \frac{1.6065z^{-1}-0.6065z^{-2}}{0.78694(1-z^{-1})} = \frac{2.0415-0.7707z^{-1}}{1-z^{-1}}$$

In difference equation form, the disturbance control algorithm is

$$f(k) = f(k-1) + 2.0415c(k) - 0.7707c(k-1)$$

The results agree with those obtained in Example 6.8-1.

Example 6.8-4

Consider the plain integrator plant described by the transfer function

$$G_p(s) = \frac{1}{s}$$

Design the deadbeat disturbance controller assuming $T=0.5$ sec. For this case,

$$\mathbf{A} = 0, \mathbf{B} = 1, \mathbf{C} = 1, \mathbf{W} = 1, \text{ and } \mathbf{V} = 0.5$$

From Eq. (6.8-24)

$$\mathbf{Q} = \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

and, thus

$$\mathbf{H} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$$

From Example 6.8-3, the disturbance controller transfer function is

$$G_d(z) = \frac{2z^{-1} - z^{-2}}{-z^{-1}(1 - z^{-1})(0.5)(-1)} = \frac{4 - 2z^{-1}}{1 - z^{-1}}$$

and the corresponding difference equation is

$$f(k) = f(k-1) + 4c(k) - 2c(k-1)$$

6.9 A CONTROL SCHEME FOR REFERENCE AND/OR DISTURBANCE INPUTS

It was pointed out earlier that control algorithms designed with reference inputs in mind may perform satisfactorily when reference and/or disturbance inputs enter the control loop. For instance, consider the first-order plant described by the transfer function

$$G_p(s) = \frac{K}{s+a} \quad (6.9-1)$$

which was considered in section 6.6. From Fig. 6.8-1, and assuming $r(k)=0$, we determine the output-to-disturbance transfer function

$$\frac{C(z)}{D(z)} = \frac{G_h G_p(z)}{1 + G_h G_p(z) G_c(z)} \quad (6.9-2)$$

The pulse transfer function $G_h G_p(z)$ is given by Eq. (6.6-17), and the deadbeat controller $G_c(z)$ is given by Eq. (6.6-25). After substituting these expressions in Eq. (6.9-2), we obtain the desired result

$$\frac{C(z)}{D(z)} = \left(\frac{K}{a} \right) \frac{(1-e^{-aT})(1-z^{-1})z^{-1}}{1-e^{-aT}z^{-1}} \quad (6.9-3)$$

From the final-value theorem, one can show that the output vanishes for a step function input. That is,

$$c(\infty) = \lim_{z \rightarrow 1} (1-z^{-1})C(z) = \lim_{z \rightarrow 1} \left(\frac{K}{a} \right) \frac{(1-z^{-1})(1-e^{-aT})z^{-1}D_0}{1-e^{-aT}z^{-1}} = 0 \quad (6.9-4)$$

Thus, a deadbeat controller designed for the first-order plant given by Eq. (6.9-1) to track reference inputs will reject a step in disturbance successfully, also. The disturbance, however, will not be removed in deadbeat fashion. The actual system response to a step in disturbance can be determined from Eq. (6.9-3) and it is given by

$$c(k) = D_0 \left(\frac{K}{a} \right) (1-e^{-aT})e^{-aT(k-1)} ; \text{ for } k \geq 1 \quad (6.9-5)$$

where D_0 is the step size of the disturbance, and $c(0)=0$.

Example 6.9-1

For a specific example, consider the control system shown in Fig. 6.8-1, and assume a plant whose transfer function is given by

$$G_p(s) = \frac{1}{s+1}$$

Assuming $T=0.5$ sec, determine the deadbeat reference controller, and evaluate its response to a unit-step in disturbance. The deadbeat controller was derived in Example 6.6-2, and its performance for a unit-step in disturbance is shown in Fig. 6.9-1.

Example 6.9-2

Again, consider the control system shown in Fig. 6.8-1, and assume a plant whose transfer function is the plain integrator

$$G_p(s) = \frac{K}{s}$$

Assuming $T=0.5$ sec, determine the deadbeat reference controller, and evaluate its response to a unit-step in disturbance.

For the above plant, the closed-loop deadbeat controller for reference inputs is given by Eq. (6.6-14). Then, from Fig. 6.8-1, the output-to-disturbance transfer function becomes

$$\frac{C(z)}{D(z)} = \frac{G_h G_p(z)}{1 + G_h G_p(z) G_c(z)} = K T z^{-1}$$

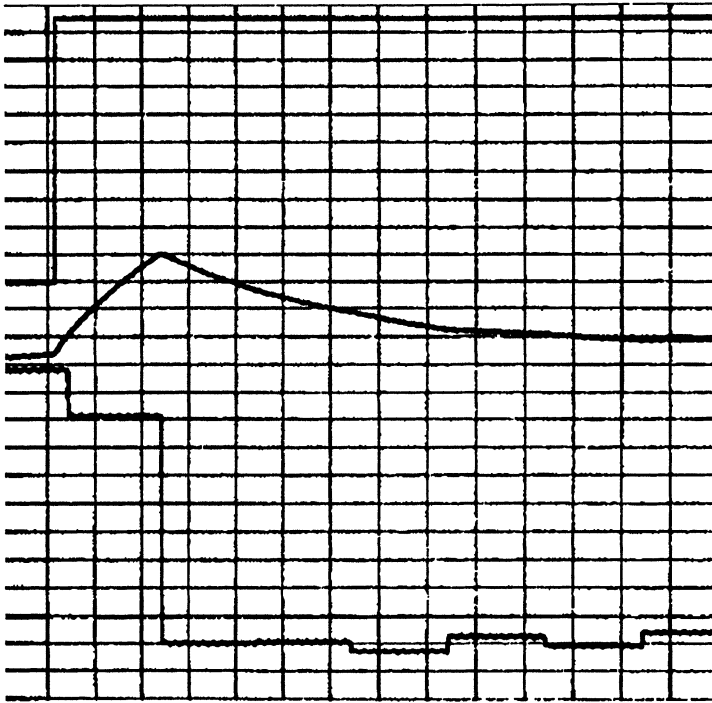


Fig. 6.9-1. Performance of deadbeat controller for the system of Fig. 6.8-1 when a disturbance of 1 volt enters the control loop. $G_p(s) = \frac{1}{(s+1)}$, $T=0.5$ sec.

Thus, for the integrator plant control system the output follows the step input in disturbance by another step scaled by KT and delayed by one sampling time. In general, deadbeat algorithms designed for closed-loop control will not remove the effect of an external disturbance if the plant contains a plain integrator. This type of performance in rejecting disturbances is not considered acceptable.

The control scheme proposed next responds to reference and/or disturbance inputs in deadbeat manner. Specifically, if a step in reference enters the control loop, the system tracks it with deadbeat response, and if a step in disturbance enters the control loop, the system rejects it in deadbeat manner. The control scheme is shown in Fig. 6.9-2, and it is basically obtained by superimposing the control systems of Fig. 6.8-1 and of Fig. 6.8-2. Note that $G_c(z)$ is the deadbeat controller for following reference inputs as shown in Fig. 6.8-1, and $G_d(z)$ is the deadbeat controller for rejecting disturbance inputs as shown in Fig. 6.8-2.

The operation of the control system of Fig. 6.9-2 may be viewed as follows. If there is no disturbance entering the control system, that is, if $d(k)=0$, then the input to the disturbance controller is zero, also. This assumes that the response of the actual plant matches that of the plant model, represented by its pulse transfer function $G_h G_p(z)$. If $d(k)$ is not zero, then the signal $x(k)$ is not zero, and the controller $G_d(z)$ is driven to correct the situation by removing the effect of noise.

Even though the control system of Fig. 6.9-2 performs ideally (by our deadbeat definition), its implementation can be improved considerably by simplifying the system block diagram. Using block-diagram reduction, the system of Fig. 6.9-2 can be represented by its equivalent diagram shown in Fig. 6.9-3. For a specific case, we can determine the relevant transfer functions $G_c(z)$, $G_d(z)$, and $G_h G_p(z)$, and substitute them to simplify the system diagram and, thus, operations. The overall procedure is illustrated next by general and specific examples.

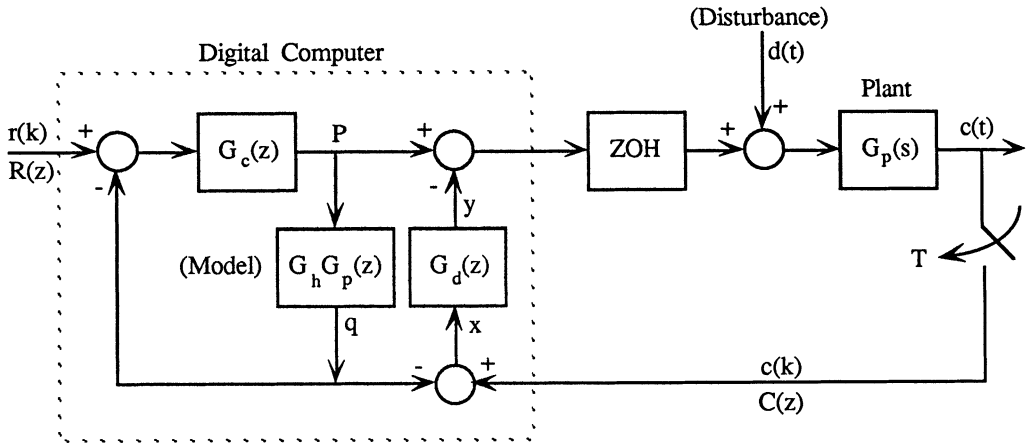


Fig. 6.9-2 Control scheme for reference and/or disturbance inputs.

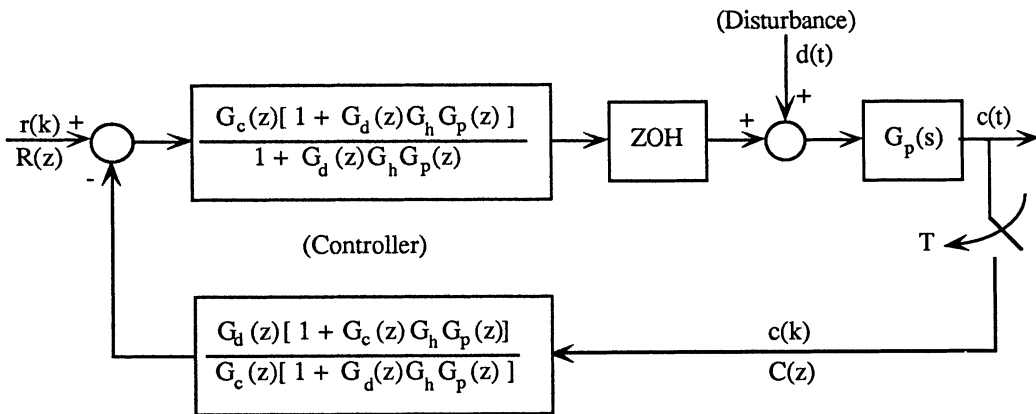


Fig. 6.9-3 Equivalent representation of the system of Fig. 6.9-2.

6.9-1 The Plain Integrator Plant: Step Inputs

To illustrate the previous concepts, consider the integrator plant described by

$$G_p(s) = \frac{K}{s} \quad (6.9-8)$$

and assume that it is required to determine the relevant controllers and block diagrams so that this plant is controlled in a deadbeat fashion for both reference and/or disturbance inputs.

The transfer functions have already been calculated. That is, $G_h G_p(z)$ is given by Eq. (6.6-3), $G_c(z)$ is given by Eq. (6.6-14), and $G_d(z)$ was determined in Example 6.8-2. Substituting these expressions in Fig. 6.9-3 and simplifying results in the equivalent system diagram shown in Fig. 6.9-4a. Rearranging the block diagram further, leads to the more "conventional" representation shown in Fig. 6.9-4b.

Note that in the system diagram of Fig. 6.9-4b there are two control actions, the proportional (P) in the feedback (F) path, and the integral (I) in the forward path. This system is a modified version of the PDF control scheme used in examples in Chapters 2 and 3. From Fig. 6.9-4b, the proportional and integral control gains are given by

$$K_p = K_i = \frac{1}{KT} \quad (6.9-9)$$

These gains guarantee that the system will follow step inputs in deadbeat fashion, and that it will reject step disturbances, also in deadbeat fashion.

It is relatively easy to determine that the output-to-reference difference equation model for the system is

$$c(k) = r(k-1) \quad (6.9-10)$$

and that the output-to-disturbance difference equation model is

$$c(k) = KT[d(k-1) - d(k-2)] \quad (6.9-11)$$

The above equations verify the deadbeat performance of the control system for both reference and/or disturbance inputs.

Example 6.9-2

Determine the deadbeat control system for the integrator plant transfer function

$$G_p(s) = \frac{10}{s}$$

assuming a sampling time $T=0.1$ sec.

The digital PDF controller gains required so that the system may track step inputs and/or reject step disturbances in deadbeat manner are

$$K_p = K_i = \frac{1}{KT} = 1$$

with the derivative gain $K_d=0$.

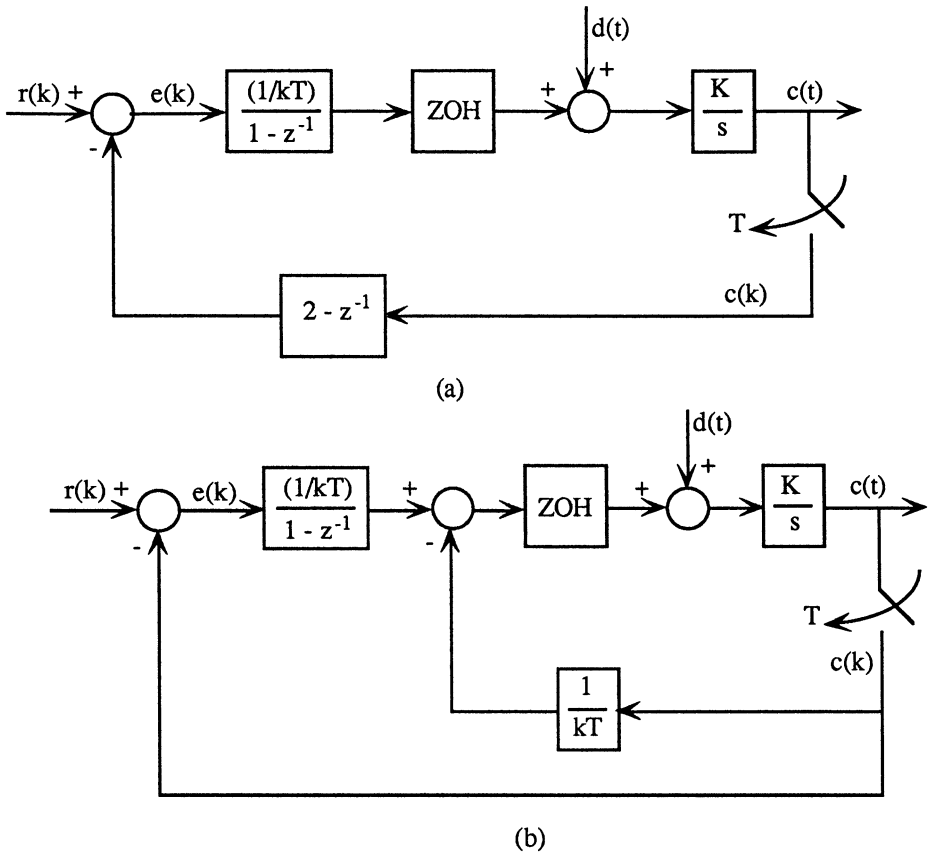


Fig. 6.9-4 Deadbeat control of integrator plant for reference and/or disturbance step inputs (a). Equivalent diagram showing "time-optimal" or deadbeat PDF control loop (b).

6.9-2 The First-Order Plant: Step Inputs

To determine the PDF control system for the first-order plant case, consider the transfer function model

$$G_p(s) = \frac{K}{s+a} \quad (6.9-12)$$

For this plant, the $G_c(z)$ deadbeat controller is given by Eq. (6.6-25), and the $G_d(z)$ controller by Eq. (6.8-15). From the block diagram of Fig. 6.9-3, we arrive at the simplified implementation shown in Fig. 6.9-5. Thus, for deadbeat response, the PI gains of the PDF controller are

$$K_i = \left(\frac{a}{K}\right) \left(\frac{1}{1-e^{-aT}}\right) \quad (6.9-13)$$

and

$$K_p = e^{-aT} K_i \quad (6.9-14)$$

with $K_d = 0$.

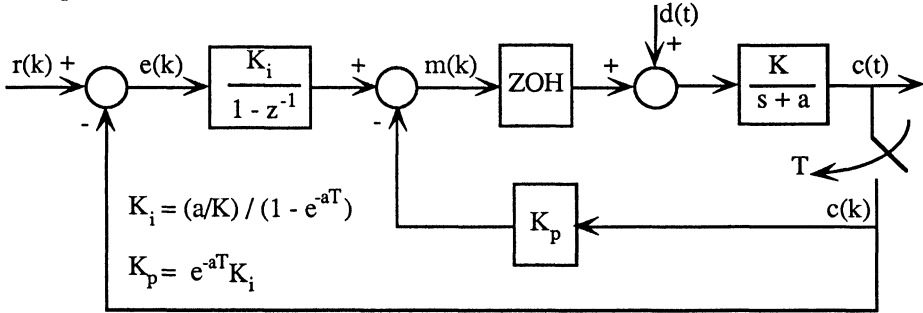


Fig. 6.9-5 Deadbeat control of first-order plant for reference and/or disturbance inputs (steps). "Optimally-tuned" PDF loop.

Example 6.9-3

Design the PDF control system, that is, determine the controller gains for the first-order plant

$$G_p(s) = \frac{10}{s+5}$$

It is required that the control system track step inputs and/or reject step disturbances in deadbeat manner. Use a sampling time of $T=0.1$ sec.

From Eq. (6.9-13) and Eq. (6.9-14), the required PDF controller gains are

$$K_i = \left(\frac{a}{K}\right) \left(\frac{1}{1-e^{-aT}}\right) = 1.27075$$

and

$$K_p = e^{-aT} K_i = 0.77075$$

Example 6.9-4

For another example of designing a deadbeat control system with reference and/or disturbance inputs, consider the second-order plant

$$G_p(s) = \frac{10}{s(s+5)}$$

and assume that $T=0.1$ sec. The corresponding pulse transfer function $G_h G_p(z)$ is given by Eq. (6.7-49), and the noise controller $G_d(z)$ is given by Eq. (6.8-74). The deadbeat controller for step inputs is determined as

$$G_c(z) = \frac{12.7075 - 7.70747z^{-1}}{1 + 0.45851z^{-1}}$$

Substituting the transfer function relationships in the system of Fig. 6.9-3, and rearranging, we obtain the control system block diagram shown in Fig. 6.9-6.

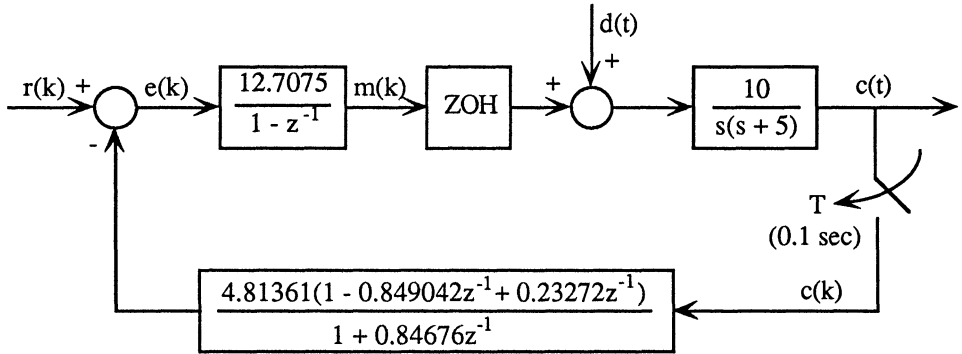


Fig. 6.9-6 Deadbeat control system for reference and/or disturbance inputs (steps). The plant transfer function is $\frac{10}{s(s + 5)}$ and $T = 0.1$ sec.

6.9-3 The Plain Integrator Plant: Ramp Inputs

Consider now the plain integrator plant of Eq. (6.7-6), and assume that the reference input is a ramp function of slope k_1 . The deadbeat controller for ramp inputs is given by Eq. (6.7-17), the controller for steps in disturbance is given by Eq. (6.8-8), and the pulse transfer function is given by Eq. (6.6-17). Substituting these expressions in the system of Fig. 6.9-3, and simplifying, leads to the system shown in Fig. 6.9-7a.

The block diagram of Fig. 6.9-7a can be further rearranged as shown in Fig. 6.9-7b, which represents a modified (PI) version of the conventional PID algorithm, introduced in analog form in Chapter 2. The relevant difference equation models are

$$c(k) = 2r(k-1) - r(k-2) \quad (6.9-15)$$

$$\text{and} \quad c(k) = KT[d(k-1) - d(k-2)] \quad (6.9-16)$$

The above equations verify the deadbeat performance of the control system.

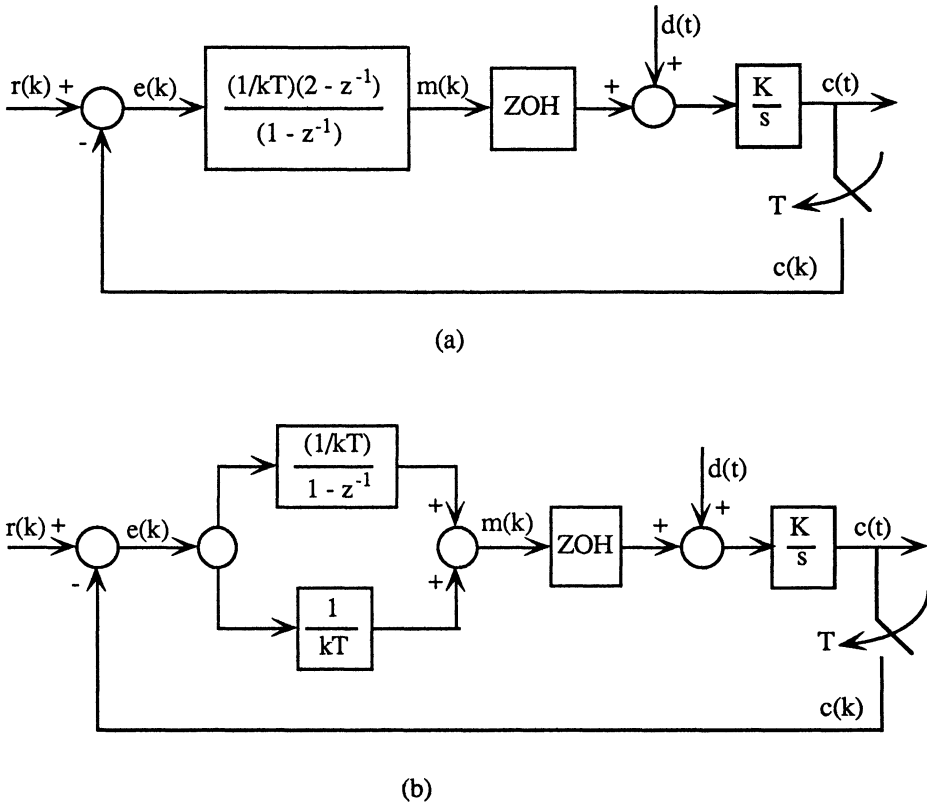


Fig. 6.9-7 (a) Deadbeat control of integrator plant for reference (ramp) and/or disturbance (step) inputs . (b) Equivalent diagram showing PID control loop with tuning values for deadbeat response.

Example 6.9-5

Consider the integrator plant of Example 6.9-2, but assume that it is required to track ramp, rather than step, inputs in deadbeat manner. For this problem, the control gains are given by Eq. (6.9-13), that is,

$$K_p = K_i = 1$$

and the control system is shown in Fig. 6.9-7b. Performance characteristics are shown simulated in Fig. 6.10-4.

Example 6.9-6

It is required to design a control system so that the first-order plant of Example 6.9-3 tracks ramp references and/or rejects step disturbances in deadbeat manner. Assume that $T=0.1$ sec.

It can be determined that the relevant transfer functions are

$$G_h G_p(z) = \frac{0.78694z^{-1}}{1 - 0.60653z^{-1}}$$

$$G_c(z) = \frac{2.54149 - 2.81225z^{-1} + 0.77075z^{-2}}{(1 - z^{-1})^2}$$

and

$$G_d(z) = \frac{2.04149 - 0.77075z^{-1}}{1 - z^{-1}}$$

Substituting the above expressions in the system of Fig. 6.9-3, and simplifying, we obtain the control system shown in Fig. 6.9-8.

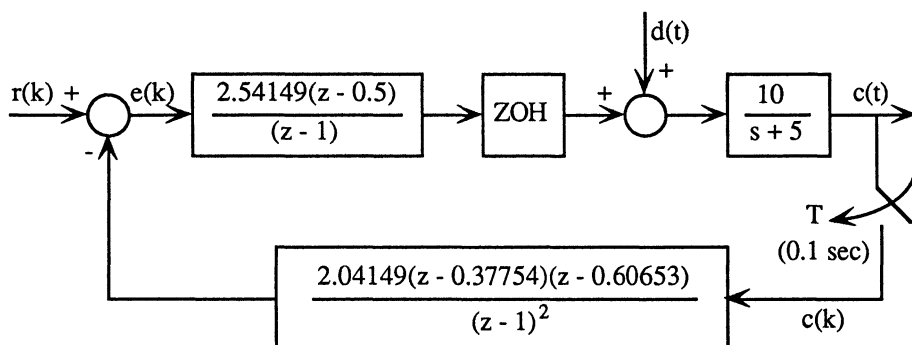


Fig. 6.9-8 Deadbeat control system for reference (ramp) and/or disturbance (step) inputs, as designed in Example 6.9-6.

6.10 SIMULATION OF COMPUTER CONTROL SYSTEMS

The experimental results presented earlier in this chapter were obtained in the laboratory by simulating the analog plants on an analog computer. The control algorithm was implemented on a digital microcomputer which used an ADC to sample the plant output, and a DAC to apply the controller output (manipulation) to the plant.

Even though it is possible to write a simulation program from the system block diagram using appropriate mathematical models for the plant and the control algorithm, this section emphasizes simulating a digital control system using problem-oriented languages.

6.10-1 Simulation Languages

Simulation languages such as ACSL and SIMULAB make simulating digital control systems practical as well as realistic. Analog plants can be simulated with ease, and their signals integrated with interface devices and digital components. Simulation results may be obtained at the sampling instants as well as between them. This approach resembles

methods followed when implementing computer control systems in actual real-time situations.

Example 6.10-1

Assume that it is required to control the plant shown in Fig. 6.10-1 in deadbeat manner using the closed-loop control scheme depicted by Fig. 6.8-1. Assume a sampling time of $T=0.1$ sec. Determine and plot the response of the control system when the reference and disturbance inputs are step functions.

The plant itself represents a linear model of an armature-controlled DC motor in block diagram form. The armature voltage $v_a(t)$ is the output of a unity-gain voltage amplifier, and $t_d(t)$ denotes the torque disturbance or load input. The parameters shown in the block diagram are assigned the following values:

$R_a=2 \Omega$, $L_a=0.1$ H, $K_m=7.04$ (N-m/A), $J=1.5$ (Kg-m²), $F=0.2$ (N-m/rad/sec), $K_b=1$ (v/rad/sec), and $T=0.1$ sec.

Let the feedback velocity ω_f be state variable $x_1(t)$, and the armature current $i_a(t)$ be state variable $x_2(t)$. Then, from Fig. 6.10-1, the continuous state and output equations are

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -0.133 & 4.693 \\ -10 & -20 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 10 \end{bmatrix} m(t)$$

and
$$c(t) = [1 \ 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = x_1(t)$$

Then, using the relationships developed in Section 4.5, with $T=0.1$ sec, the discrete state and output equations are

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.85927 & 0.18668 \\ -0.39778 & 0.069 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.12817 \\ 0.40142 \end{bmatrix} m(k)$$

and
$$c(k) = x_1(k)$$

From Eq. (6.6-57), the deadbeat controller is determined as

$$G_c(z) = \frac{5.14766 - 4.77844z^{-1} + 0.68746z^{-2}}{1 - 0.65978z^{-1} - 0.34022z^{-2}}$$

Simulation results for the program shown in Fig. 6.10-2a are plotted in Fig. 6.10-2b. Note that the sampling time is $T=0.1$ sec and that both reference as well as a torque-disturbance inputs enter the control loop.

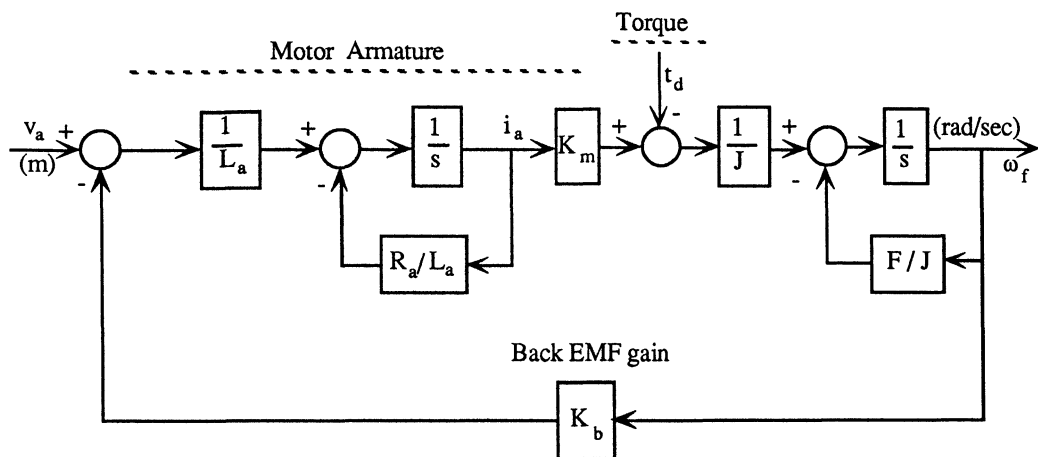


Fig. 6.10-1 Block diagram model of armature-controlled DC motor used in Example 6.10-1.

Example 6.10-2

Assume that the plain integrator plant used in Example 6.9-2 is controlled according to the PDF control scheme of Fig. 6.9-4b for deadbeat response. Simulation results are shown plotted in Fig. 6.10-3. The performance of the PDF control system is illustrated for both ramp and step inputs, and for step changes in disturbance.

The simulation results verify the analytical findings obtained earlier. As expected, the control system tracks step inputs and/or rejects step disturbances in deadbeat manner. However, being a type 1 system, it can not follow ramp inputs with zero steady-state error.

Example 6.10-3

Assume that the integrator plant used in Example 6.9-5 is controlled according to the PID scheme shown in Fig. 6.9-7b. Simulation results are shown in Fig. 6.10-4. Note how this control system tracks ramp inputs with zero steady-state error but, compared to the PDF system of Example 6.10-2, it overshoots excessively when excited by step inputs. The performance of both the PID and PDF control systems to steps in disturbance is identical. That is, both control schemes reject disturbances in deadbeat manner.

Example 6.10-4

As a final example, consider the control system of Fig. 6.9-6 which controls the second-order plant of Example 6.9-4 - for deadbeat response to reference and/or disturbance inputs (steps). Simulation results are shown in Fig. 6.10-5.

```

CINTERVAL CINT = 0.001      $ "DYNAMIC section interval time (sec.)"
CONSTANT RA = 2.0           $ "Motor armature resistance"
CONSTANT LA = 0.1           $ "Motor armature inductance"
CONSTANT KM = 7.04          $ "Motor constant"
CONSTANT JA = 1.5           $ "Motor inertia"
CONSTANT FM = 0.2           $ "Motor friction"
CONSTANT KB = 1.0           $ "Back EMF gain"
INITIAL $ "Initialize variables before start of simulation"
  IA = 0.0
  WF = 0.0
  VELFBK = 0.0
  E1 = 0.0
  E2 = 0.0
  M1 = 0.0
  M2 = 0.0
END $ "Initial"
DYNAMIC $ "Start simulation"
  DERIVATIVE $ "Continuous analog DC motor model"
    TD = STEP( 0.5 ) - STEP( 1.5 )
    A1 = VA - VELFBK
    A2 = ( 1.0 / LA ) * A1
    A3 = A2 - ( ( RA / LA ) * IA )
    IA = INTEG( A3, 0.0 )
    A4 = ( IA * KM ) - TD
    A5 = ( 1.0 / JA ) * A4
    A6 = A5 - ( ( FM / JA ) * WF )
    WF = INTEG( A6, 0.0 )
    VELFBK = KB * WF
    TERMT( T .GE. 2.5 )      $ "Length of simulation sec."
  END $ "Derivative"
DISCRETE $ "Discrete deadbeat control"
  INTERVAL TSAMP = 0.1      $ "How often deadbeat control runs"
  PROCEDURAL
    VELREF = STEP( 0.0 )
    E = VELREF - VELFBK
    MANIP = ( 5.14766 * E ) - ( 4.77844 * E1 ) + ( 0.68746 * E2 ) + ...
      ( 0.65978 * M1 ) + ( 0.34022 * M2 )
    E2 = E1
    E1 = E
    M2 = M1
    M1 = MANIP
    VA = MANIP
  END $ "Procedural"
END $ "Discrete deadbeat control"
END $ "Dynamic"
END $ "End of program"

```

Fig. 6.10-2a ACSL simulation program for system of Example 6.10-2.

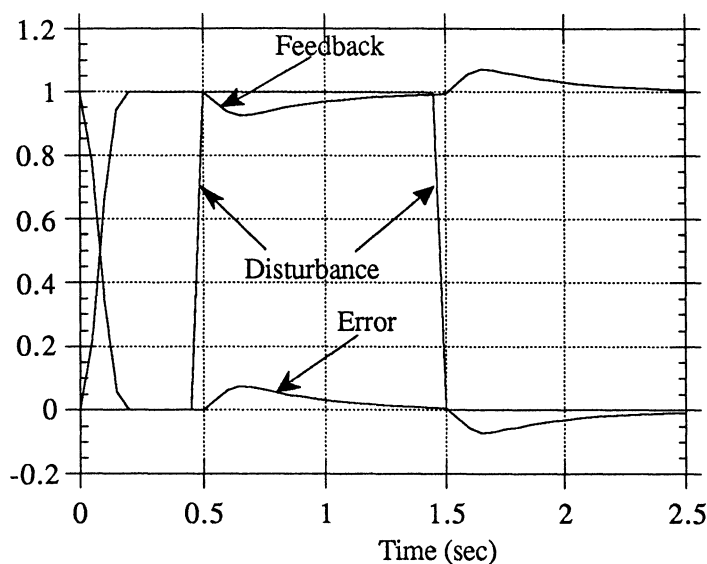


Fig. 6.10-2b Feedback, error and disturbance signals for system of Example 6.10-1.

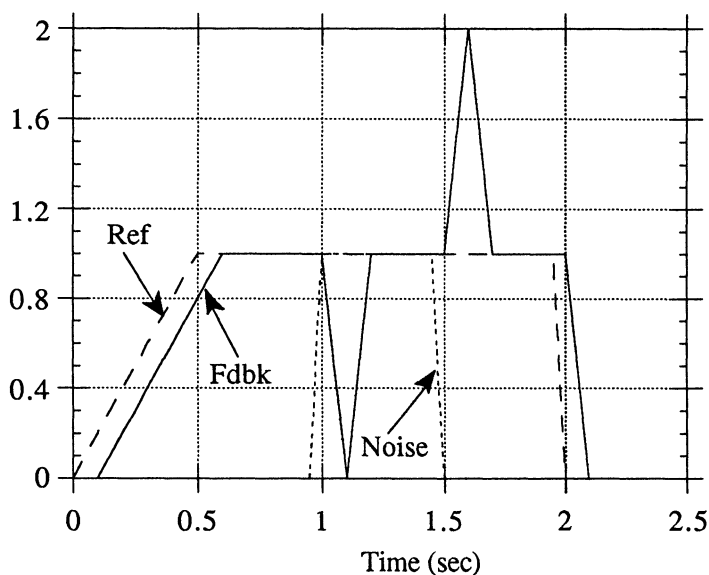


Fig. 6.10-3 Reference, feedback, and disturbance signals for PDF control system of Example 6.10-2.

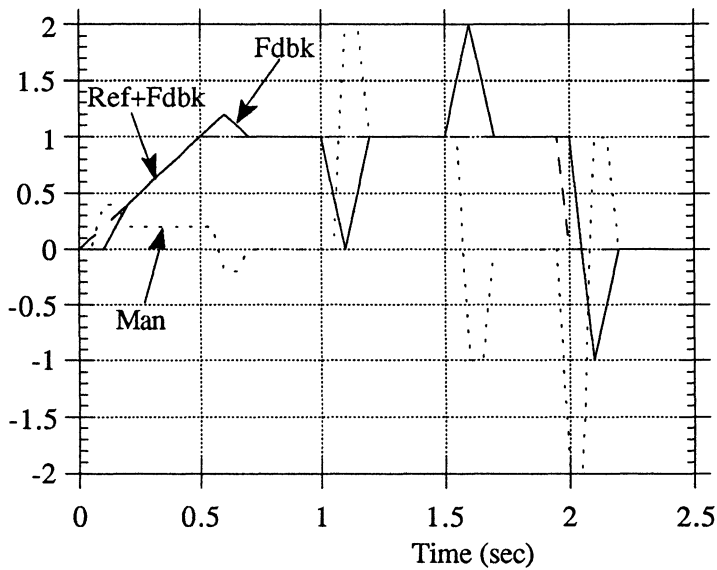


Fig. 6.10-4 Reference, feedback, and manipulation signals for PID control system.

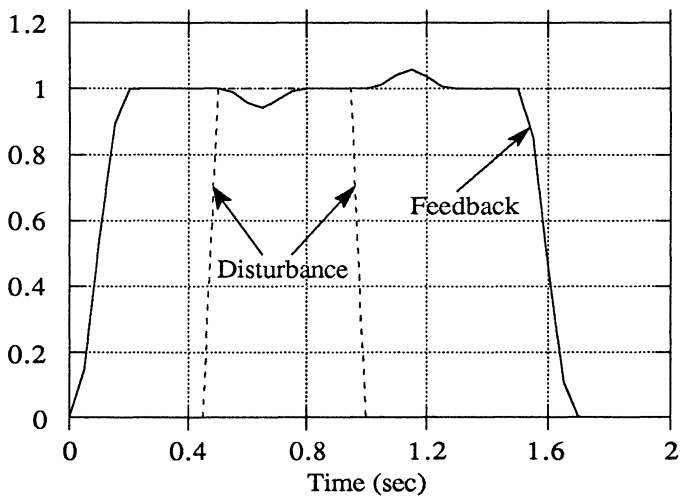


Fig. 6.10-5 Feedback and disturbance signals for system of Example 6.10-4.

6.10 SOME OBSERVATIONS ON DEADBEAT CONTROL

The main reason for considering deadbeat algorithms first is that these controllers require no "tuning" and offer the designer freedom to concentrate on other aspects of the control problem. The deadbeat design method is systematic and general, which makes it appropriate as a vehicle for illustrating the fundamental concepts of open-loop control, closed-loop control, and control of external disturbances.

The Laplace transform, z transform, and state-variable methods have been used to illustrate various design and simulation techniques. These approaches are also intended for the reader who wants to understand better the computer control problem and its alternative solutions.

In summary, in order to obtain deadbeat performance, the mathematical model of the plant must be known, otherwise the response will not be "true" deadbeat. In addition, the computer used to implement the control algorithm must have "adequate" accuracy and speed characteristics. In practice, however, plant models are known or can be determined only "approximately", and some microcomputers may not be fast and/or accurate enough for very demanding applications.

The sensitivity of a deadbeat control system to plant parameter changes can be readily determined by simulation. For instance, one can determine that by changing the gain K of the plant in Example 6.10-3 by 5% results in a brief 5% overshoot in the step response of the system. Similarly, changing the plant pole location by 5% results in overshoot or ripple of less than 2%.

Deadbeat algorithms are designed for a certain type of input and may or may not perform acceptably for a different input. For instance, the deadbeat algorithm designed for tracking ramp inputs overshoots excessively when the input is a step. However, a deadbeat algorithm designed for step inputs may prove acceptable when the input is a ramp function.

PROBLEMS

- 6-1. A plant is described by the transfer function

$$G_p(s) = \frac{1}{s+1}$$

Determine an open-loop deadbeat controller for a sampling time of $T=0.1$ sec. Express the controller in difference equation form and find the unit-step response of the control system.

- 6-2. Repeat Problem 6-1 for the plant

$$G_p(s) = \frac{4}{s^2+s+4}$$

and a sampling time of $T=0.1$ sec.

- 6-3. Determine closed-loop deadbeat controllers for (a) the plant of Problem 6-1, and (b) the plant of Problem 6-2. Use simulation to verify your findings. Determine and sketch/plot the response of each control loop to a step in disturbance.

- 6-4. For the plants of Problem 6-1 and 6-2, design algorithms that would remove disturbances in deadbeat fashion. Simulate the control loops to verify the results.
- 6-5. Design appropriate deadbeat controllers so that the plant of Problem 6-1 can follow step inputs and/or reject step disturbances, both in deadbeat manner. Incorporate these controllers in the block diagram of Fig. 6.9-2 and simplify it. Then, simulate the system to see that it works.
- 6-6. A plant is described by the differential equation

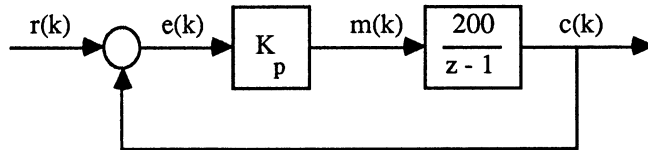
$$k_1 \frac{dc(t)}{dt} + c(t) = k_2 m(t)$$

where $m(t)$ and $c(t)$ represent the plant input and output, respectively. Develop the digital algorithms for closed-loop digital control so that

- (a) $c(k) = r(k)$, and (b) $c(k+1) = r(k)$

where $r(k)$ and $c(k)$ represent the control system input and output, respectively. For the plant, assume a time constant of 0.2 sec and a steady-state gain of 1. Then, using a sampling time of $T=0.1$ sec, determine and sketch $c(k)$, $m(k)$, $e(k)$, and $r(k)$ for the cases (a) and (b), above.

- 6-7. Consider the digital control system shown. Find the range of the gain K_p for stability. Also, determine the value of K_p that results in deadbeat response to a step input.



- 6-8. A plant is described by the transfer function

$$G_p(s) = \frac{aK}{s(s+a)}$$

Assuming that the plant is to be controlled using the PDFF scheme, determine (a) the transfer function $C(s)/R(s)$, the transfer function $C(s)/D(s)$, and (c) the transfer function $H(s)G(s)$.

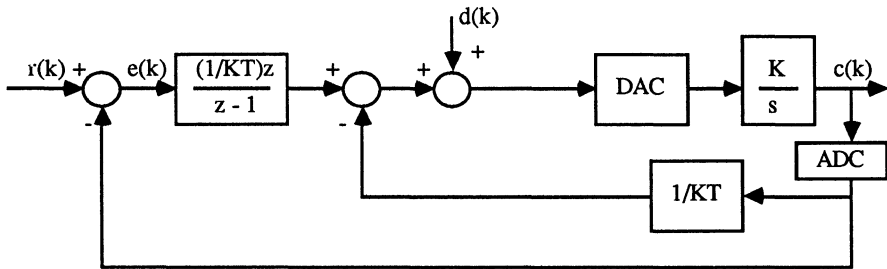
- 6-9. Assuming backward-difference approximation for integration/differentiation, draw the corresponding digital PDFF control system for Problem 6-8. Then, using the parameter values: $K=200$, $a=10$, and $T=0.01$ sec, determine the appropriate digital controller gains that will result in binomial-type filter response with a cutoff frequency of $\omega=20$ rad/sec.

6-10. A first-order lag with dead time plant is described by its transfer function

$$G_p(s) = \frac{Ke^{-sT}}{s+a}$$

Determine the digital feedback controller $G_c(z)$ that results in deadbeat response to step inputs.

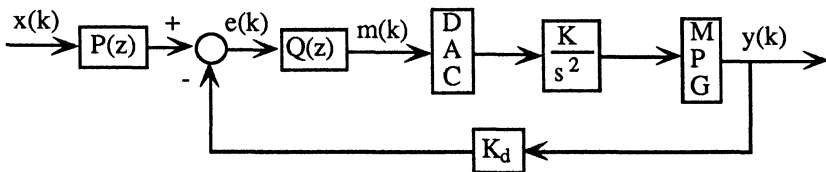
6-11. A computer controlled system is represented by the block diagram shown. Assuming that $d(k)=0$, determine (a) the transfer function $C(z)/R(z)$, (b) the corresponding difference equation model, and (c) the unit-step response of the system.

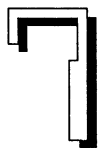


6-12. The control loop shown incorporates a double integrator plant with gain K . The interfaces consist of a DAC and a pulse generator (PG or MPG), and the feedback gain K_d is associated with derivative control.

There are two objectives in this problem. (1) To design an open-loop deadbeat controller $P(z)$, with $Q(z)=1$; and (2) to design the closed-loop deadbeat controller $Q(z)$ with $P(z)=1$. Determine both controllers and analyze the control system for each case. Evaluate your findings.

The gain of the DAC is called K_{da} , and the gain of the PG K_{dt} . Also, from Chapter 4, $K_\theta = (K_{da})(K)(K_{dt})$.





COMPUTER CONTROLLED SYSTEMS:

PID, PDFF , Phase-Lead & Other Control Algorithms

7.1 INTRODUCTION

Chapter 6 emphasizes digital controllers whose design depends on knowing the model of the plant. In such cases, the algorithm design can be carried out directly in the frequency domain using transform methods or in the time domain using state space methods.

When designing digital algorithms, an alternative approach is to ignore any zero-order-hold (ZOH) and samplers in the control loop and perform the preliminary design in the s -domain. The continuous controller is subsequently converted to its digital counterpart using appropriate approximation techniques. This latter method of design parallels the approach taken in Chapter 5 where digital filters were designed from their analog prototypes. The motivation is the same, since successful designs of analog controllers are well established, the objective is to design the discrete equivalent of the analog controller so that the performance of the digital control system resembles, as closely as possible, the performance of the corresponding analog control system.

In practice, many control systems are designed using conventional controllers such as the proportional-integral-derivative (PID) or the PDFF controllers encountered in previous chapters. Therefore, it is natural that many digital control systems are designed using the digital equivalent of the analog controllers. Generally, the performance of a digital controller can be made to approach that of its analog counterpart if the sampling time is small, relative to the plant dominant time constant. If this basic rule is not observed, however, deterioration in performance of the digital control system may be significant and alternative control schemes must be considered.

Compared to the deadbeat algorithms considered in Chapter 6, the PID and PDFF controllers can be used to control plants for which mathematical models are not readily available. Furthermore, the latter algorithms result in control systems which are relatively **robust**, *can withstand plant parameter variations and reasonable nonlinear operation*,

compared to deadbeat algorithms which are sensitive to both of these conditions. However, the PID/PDFF controllers must be **tuned**, *their control gains must be properly adjusted*, which can be a rather tedious undertaking.

Control systems, analog or digital, are usually classified by the type of action the control algorithm is programmed to take in order to achieve certain performance specifications. Assuming that the controller input is the error signal, $e(t)$, and the controller output is the manipulation or manipulated variable, $m(t)$, common types of control action are classified as follows.

Two-Position or On-Off Control Action. This is probably the most common type of control action, encountered both in home as well as in industrial environments. A home thermostat is an example of an on-off control action in which *the controller output is quickly changed to either a maximum or minimum value that starts and stops the furnace*.

Proportional Control Action. In general, *proportional control action implies that the controller output is proportional to the error input signal*. In some controller implementations, however, the controller output is proportional to the feedback or measured signal. For the former case, assuming an analog control system, the control algorithm for proportional control action is written in the form

$$m(t) = K_p e(t) \quad (7.1-1)$$

where K_p represents the **proportional gain**. As can be seen from the equation, tuning of a proportional controller is quite simple, because there is only one parameter to adjust.

Proportional control provides good speed of response and stability characteristics, but it exhibits steady-state error, called **offset**, between the reference input and the controlled variable or output.

In some areas of application, the proportional controller is expressed in terms of the **proportional band (PB)**, rather than the proportional gain K_p . By definition, *proportional band is the range of input values corresponding to a full change in the output*. In equation form, the proportional band is related to the proportional gain by

$$PB = \frac{100}{K_p} \quad (7.1-2)$$

Integral or Reset Control Action. The control algorithm for integral control action is given by

$$m(t) = K_I \int e(t) dt \quad (7.1-3)$$

where K_I is the **integral gain**. If Eq. (7.1-3) were differentiated, the result implies that the controller output, $m(t)$, is changed at a rate proportional to the error, $e(t)$. If the error value increases over a previous value, the controller output increases to make the error zero. If the output equals the input, the error becomes zero and the controller output is stationary. Thus, *with integral control action, the steady-state error (offset) is eliminated*.

Proportional-Integral (PI) Control Action. Proportional control action is usually combined with integral control action to form the proportional-integral (PI) control action, described by the equation

$$m(t) = K_P e(t) + K_I \int e(t) dt \quad (7.1-4)$$

The PI control algorithm offers some of the advantages of both the proportional and integral algorithms. It is more difficult to tune the PI controller, however, because there are two gains to adjust.

Derivative or Rate Control Action. The algorithm for derivative control action is given by

$$m(t) = K_D \frac{de(t)}{dt} \quad (7.1-5)$$

where K_D represents the **derivative gain**. From Eq. (7.1-5), one can see that the controller output is proportional to the rate of change of the input error signal. If the error signal is large and constant, the output of the derivative controller is zero, an undesirable consequence. Therefore, the derivative control action is not useful by itself and is found in combination with other control actions.

Proportional-Integral-Derivative (PID). Adding derivative control action to PI control action results in the proportional-integral-derivative (PID) controller, also known as the **three-mode controller**. In equation form, the PID algorithm can be written as

$$m(t) = K_P e(t) + K_I \int e(t) dt + K_D \frac{de(t)}{dt} \quad (7.1-6)$$

By adding derivative to the PI action, lead or anticipatory action has been added to the controller. This tends to compensate for any lag or computational delay in the control loop and thus enhance the speed and stability characteristics of the system. Obviously, with three gains to adjust, the PID algorithm is the most sophisticated.

The controller defined by Eq. (7.1-6) is sometimes referred to as the **ideal parallel PID controller**, and is the "conventional" PID controller used throughout this book. Another type is the **ideal noninteracting PID controller**, defined by

$$m(t) = K_C \left[e(t) + \frac{1}{T_i} \int e(t) dt + T_d \frac{de(t)}{dt} \right] \quad (7.1-7)$$

where K_C is the (proportional) **gain** of the controller, T_i is the **integral** or **reset time** (reciprocal of the **reset rate**) and T_d is the **derivative time** of the controller. There is also an **interacting PID controller**, defined by

$$m(t) = K_C \left[e(t) + \frac{1}{T_i} \int e(t) dt \right] \left[1 + T_d \frac{de(t)}{dt} \right] \quad (7.1-8)$$

The control algorithms considered in this chapter include the (parallel) PID algorithm and the PDF and PDFF algorithms, analog or digital. As a reminder, in the PDF algorithm the proportional (P) and derivative (D) control actions are applied to the feedback (F) variable, rather than the error signal. The more general PDFF algorithm is obtained from the PDF algorithm by adding feedforward (F) control action. Like the PID algorithm, both the PDF and PDFF algorithms include the integral control action in the forward path.

7.2 THE ANALOG AND DIGITAL PID ALGORITHMS

The analog PID algorithm is shown incorporated in the control system of Fig. 7.2-1a. In the block diagram, $r(t)$ represents the reference input, $c(t)$ the control output, $e(t)$ the error, $m(t)$ the manipulation, $d(t)$ the disturbance and $G_p(s)$ the transfer function of the plant to be controlled. In addition, K_p , K_I , and K_D denote the proportional, integral and derivative controller gains, respectively.

From Fig. 7.2-1a, the closed-loop transfer function $C(s)/R(s)$ is found to be

$$\frac{C(s)}{R(s)} = \frac{(K_D s^2 + K_p s + K_I) G_p(s)}{s + (K_D s^2 + K_p s + K_I) G_p(s)} \quad (7.2-1)$$

the open-loop (primary feedback loop open) transfer function $H(s)G(s)$ is

$$H(s)G(s) = \frac{(K_D s^2 + K_p s + K_I) G_p(s)}{s} \quad (7.2-2)$$

and the output-to-disturbance transfer function $C(s)/D(s)$ is

$$\frac{C(s)}{D(s)} = \frac{s G_p(s)}{s + (K_D s^2 + K_p s + K_I) G_p(s)} \quad (7.2-3)$$

Also, from the block diagram, the transfer function of the analog PID algorithm $M(s)/E(s)$ is

$$\frac{M(s)}{E(s)} = \frac{K_D s^2 + K_p s + K_I}{s} \quad (7.2-4)$$

Thus, the corresponding integro-differential equation model of the PID controller is

$$m(t) = K_p e(t) + K_I \int e(t) dt + K_D \frac{de(t)}{dt} \quad (7.2-5)$$

The objective is to approximate the solution of the above integro-differential equation by a difference equation or recursive relation, which is the proper form to implement or program on a digital computer.

Assume, for instance, a backward difference approximation for the derivative term and rectangular integration for the integral term of Eq. (7.2-5). Then, the solution of Eq. (7.2-5) can be approximated by the difference equation

$$m(k) = K_p e(k) + T K_I \sum_{j=0}^k e(j) + \frac{K_D}{T} [e(k) - e(k-1)] \quad (7.2-6)$$

where T is the sampling time. Eq. (7.2-6) is known as the **position form** of the PID algorithm, because its final output is the **manipulation**, also known as the **control valve position**. The **velocity** or **incremental form** of the PID algorithm can be determined from Eq. (7.2-6) by obtaining $m(k-1)$ from $m(k)$ and subtracting the resulting expressions. That is,

$$\begin{aligned}
 m(k) - m(k-1) &= K_p e(k) + TK_I \sum_{j=0}^k e(j) + \frac{K_D}{T} [e(k) - e(k-1)] \\
 &- \{ K_p e(k-1) + TK_I \sum_{j=0}^k e(j) + \frac{K_D}{T} [e(k-1) - e(k-2)] \} \\
 &= [K_p + TK_I + \frac{K_D}{T}] e(k) - [K_p + 2(\frac{K_D}{T})] e(k-1) + \frac{K_D}{T} e(k-2)
 \end{aligned} \tag{7.2-7}$$

Note that the analog controller gains are represented by upper-case subscripts. In contrast, the corresponding digital controller gains are denoted by lower-case subscripts. Using this convention, the digital PID algorithm can be expressed in the (velocity) form

$$m(k) = m(k-1) + (K_p + K_i + K_d)e(k) - (K_p + 2K_d)e(k-1) + K_d e(k-2) \tag{7.2-8}$$

The digital controller gains K_p , K_i , and K_d can be determined from the analog controller gains using the following relationships:

$$K_p = K_p \text{ (same)} \tag{7.2-9}$$

$$K_i = TK_I \tag{7.2-10}$$

$$\text{and} \quad K_d = \frac{K_D}{T} \tag{7.2-11}$$

Taking the $\mathcal{Z}$ transform of Eq. (7.2-8) and rearranging, results in the discrete transfer function form of the PID controller

$$G_c(z) = \frac{M(z)}{E(z)} = \frac{(K_p + K_i + K_d) - (K_p + 2K_d)z^{-1} + K_d z^{-2}}{(1 - z^{-1})} \tag{7.2-12}$$

Other forms of the digital PID algorithm can be obtained by changing, for instance, the integration scheme. Thus, trapezoidal rather than rectangular integration may prove to be a more conservative choice for many applications. A digital control system incorporating such a PID algorithm is shown in Fig. 7.2-1b. Notice that ADC and DAC interfaces have been assumed.

An alternate way of expressing the difference equation representing the PID controller can be obtained by evaluating the integral control action first. From Fig 7.2-1b, with trapezoidal integration, the resulting equations are

$$x(k) = x(k-1) + 0.5[e(k) + e(k-1)] \tag{7.2-13}$$

$$\text{and} \quad m(k) = K_i x(k) + K_p e(k) + K_d [e(k) - e(k-1)] \tag{7.2-14}$$

This form of the PID algorithm is more flexible, because it allows the designer to place limits on the integrator to control the phenomenon known as **integrator** or **reset windup**. Windup occurs when the integrator builds up a large value, which results in excessive overshoots and related problems.

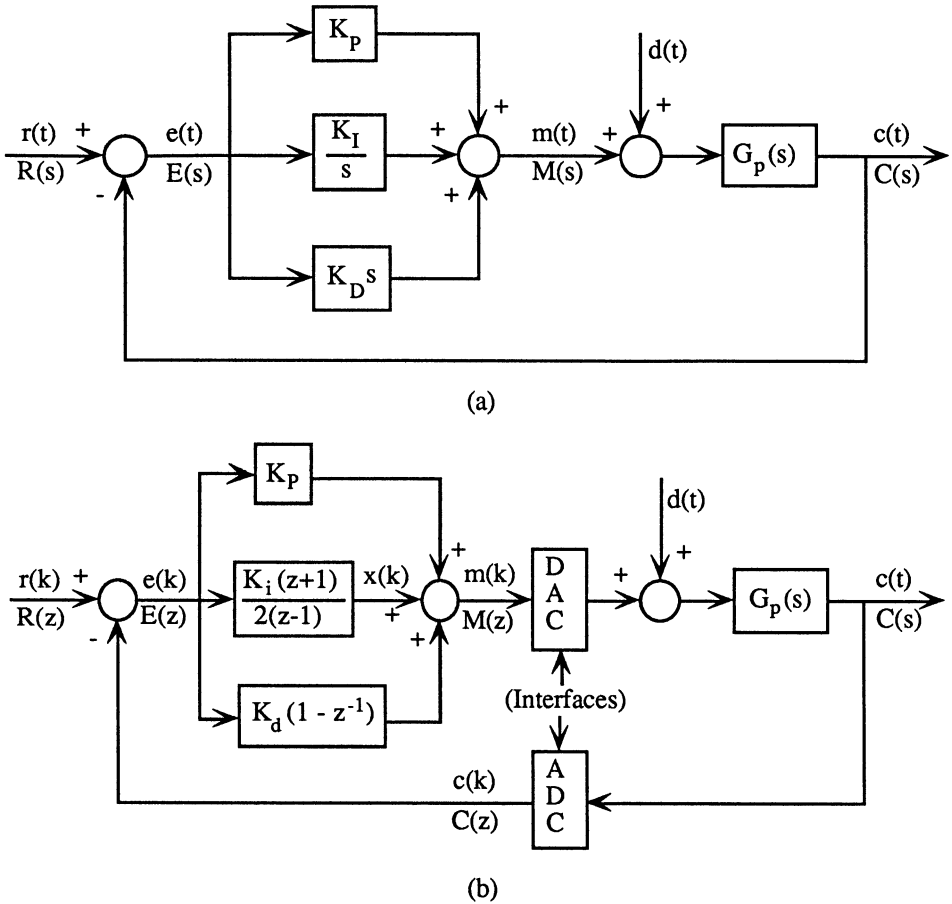


Fig. 7.2-1 (a) Analog and (b) digital PID control systems.

7.3 THE ANALOG AND DIGITAL PDF AND PDFF ALGORITHMS

The PID control system represented by the analog transfer function of Eq. (7.2-1) may be difficult to tune for acceptable step response characteristics, mainly because of the numerator zeros. The effect of the zeros is to add an early peak or overshoot to the step response of the control system. The overshoot can be appreciable and it increases as the zeros get closer to the origin.

An alternative is to remove the numerator zeros altogether. To accomplish this, consider the transfer function of Eq. (7.2-1), rewritten in the form

$$\frac{C(s)}{R(s)} = \frac{\left(\frac{K_D}{K_I} s^2 + \frac{K_P}{K_I} s + 1 \right) K_I G_p(s)}{s + (K_D s^2 + K_P s + K_I) G_p(s)} \quad (7.3-1)$$

To cancel the numerator zeros, the second-order numerator term, precede the PID block diagram of Fig. 7.2-1a with a block whose transfer function is the inverse of the numerator term we wish to cancel. If the output of this block is $R'(s)$, with input $R(s)$, the transfer function of the block is

$$\frac{R'(s)}{R(s)} = \frac{1}{\left(\frac{K_D}{K_I} s^2 + \frac{K_P}{K_I} s + 1 \right)} \quad (7.3-2)$$

The new/modified PID control system is described by the transfer function

$$\frac{C(s)}{R(s)} = \frac{K_I G_p(s)}{s + (K_D s^2 + K_P s + K_I) G_p(s)} \quad (7.3-3)$$

which, after some block diagram manipulation, becomes identical to the PDF control system shown in Fig. 7.3-1a.

The output-to-disturbance transfer function of the PDF control system is

$$\frac{C(s)}{D(s)} = \frac{s G_p(s)}{s + (K_D s^2 + K_P s + K_I) G_p(s)} \quad (7.3-4)$$

which is the same as the $C(s)/D(s)$ transfer function of the PID control system. Finally, the open-loop transfer function of the PDF control system is found to be

$$H(s)G(s) = \frac{K_I G_p(s)}{s[1 + (K_D s + K_P) G_p(s)]} \quad (7.3-5)$$

which differs from the $H(s)G(s)$ of the PID control system, given by Eq. (7.2-2).

Assuming backward-difference differentiation and trapezoidal integration, leads to the digital PDF control system shown in Fig. 7.3-1b. From the block diagram, the PDF control algorithm can be expressed by the difference equations

$$x(k) = x(k-1) + 0.5[e(k) + e(k-1)] \quad (7.3-6)$$

and

$$m(k) = K_I x(k) - K_P c(k) - K_d [c(k) - c(k-1)] \quad (7.3-7)$$

The analog PDFF control system is shown in Fig. 7.3-2a. The PDFF control algorithm may be visualized as obtained from the PDF algorithm by adding the feedforward branches K_V (proportional) and $K_A s$ (derivative). A simplified/reduced block diagram of the PDFF control system of Fig. 7.3-2a is shown in Fig. 7.3-2b. From this latter diagram, we can readily obtain the closed-loop transfer function of the PDFF control system as

$$\frac{C(s)}{R(s)} = \frac{(K_A s^2 + K_V s + K_I) G_p(s)}{s + (K_D s^2 + K_P s + K_I) G_p(s)} \quad (7.3-8)$$

The open-loop transfer function $H(s)G(s)$ is

$$H(s)G(s) = \frac{(K_A s^2 + K_V s + K_I) G_p(s)}{s[1 + (K_D s + K_P - K_A s - K_V) G_p(s)]} \quad (7.3-9)$$

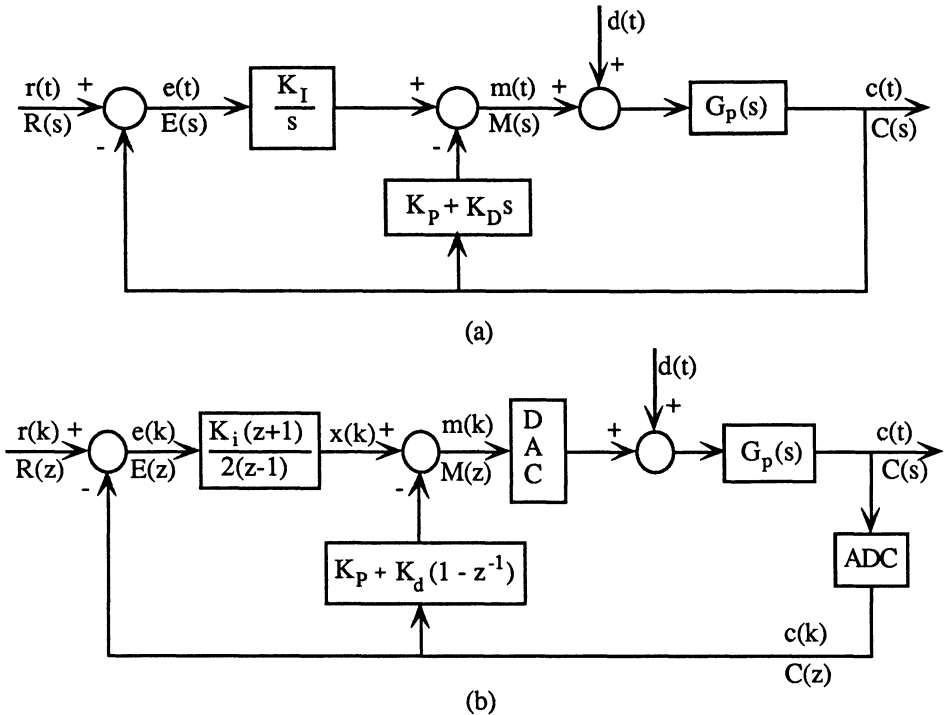


Fig. 7.3-1 (a) Analog, and (b) digital PDF control systems.

It can be shown that the output-to-disturbance transfer function, $C(s)/D(s)$, for the PDF control system is given, again, by Eq. (7.2-3).

The digital PDF control system can be obtained from its analog counterpart using the backward-difference approximation for the differentiation and the trapezoidal rule for the integration operations. The resulting control system is shown in Fig. 7.3-2c.

From Fig. 7.3-2c, the digital PDF algorithm can be written in difference equation form as

$$x(k) = x(k-1) + 0.5[e(k) + e(k-1)] \quad (7.3-10)$$

and

$$m(k) = K_i x(k) + K_v r(k) + K_a [r(k) - r(k-1)] - K_p c(k) - K_d [c(k) - c(k-1)] \quad (7.3-11)$$

In Eq. (7.3-11), the digital feedforward gains (lower-case subscripts) are related to the analog feedforward gains (upper-case subscripts) by the expressions

$$K_v = K_v \text{ (same)} \quad (7.3-12)$$

and

$$K_a = \frac{K_A}{T} \quad (7.3-13)$$

Examining the PDF control system transfer function given by Eq. (7.3-8), it is obvious that the zeros which were removed in order to derive the PDF from the PID

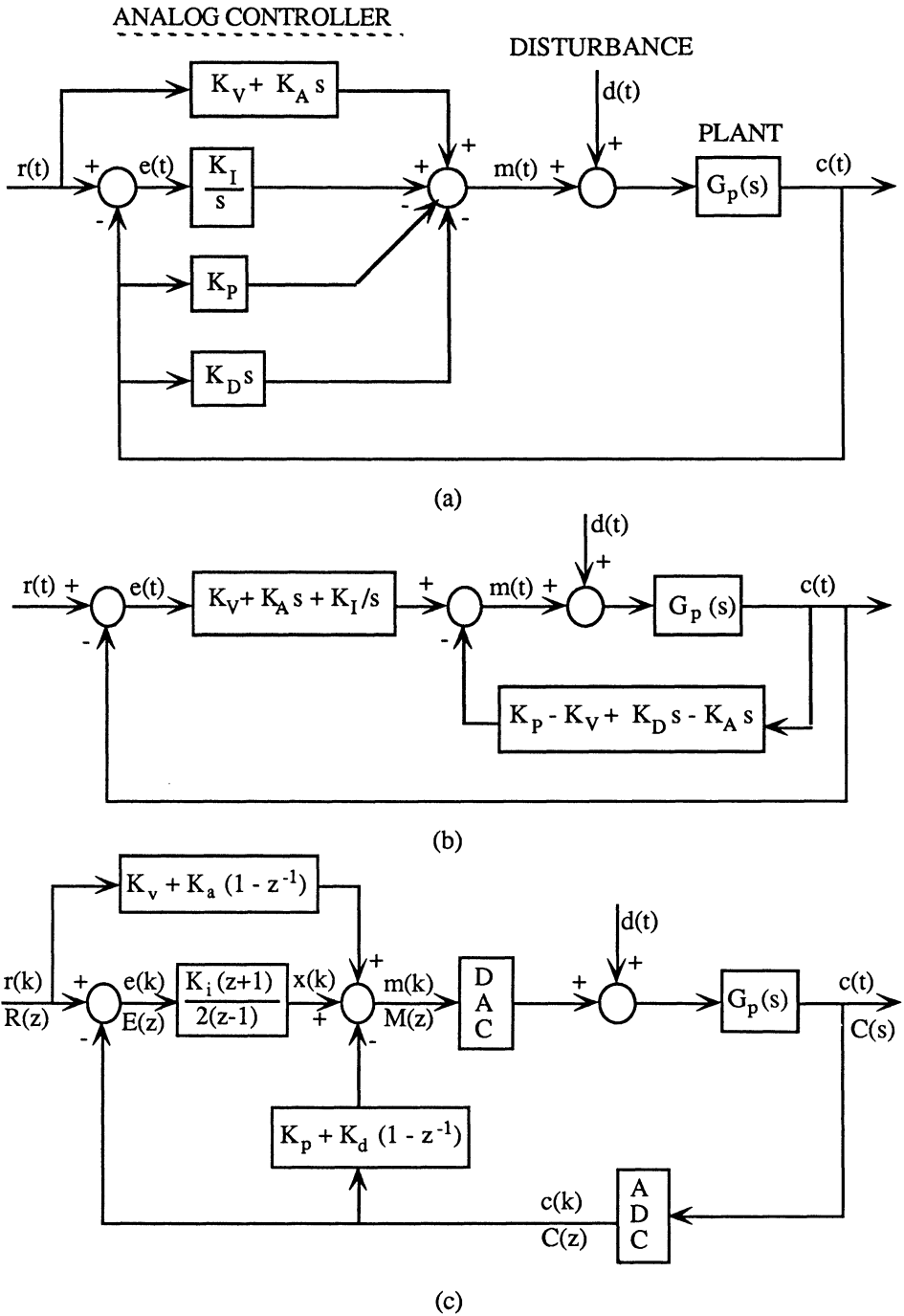


Fig. 7.3-2 (a) Analog, (b) reduced form and (c) digital PDF control system.

control system are back again. There is an important difference, however, between the PID and PDFF control systems and their respective controllers. With the PID control scheme, there is no control over the placement of the control system zeros. For instance, if the designer locates the system poles to obtain desirable frequency response characteristics, the system zeros are placed automatically, also. This is not the case with the PDFF control system, however, because the placement of the zeros is independent of the placement of the poles.

From Eq. (7.3-10) and Eq. (7.3-11), it can be seen that the PDFF algorithm reduces to the PDF algorithm if the

$$K_v = K_a = 0 \quad (7.3-14)$$

Also, the PDFF algorithm becomes identical to the PID algorithm if

$$K_v = K_p \text{ and } K_a = K_d \quad (7.3-15)$$

Furthermore, the PDFF algorithm gains K_v and K_a can be adjusted to obtain system performance characteristics that fall between the characteristics of the PID and the PDF control systems. Thus, the PDFF control scheme is quite versatile.

Tuning techniques and performance characteristics for the PID, PDF and PDFF algorithms are illustrated next by general and specific examples.

7.3-1 PDFF Control of the Plain Integrator Plant

When designing algorithms to control a plant by a computer, it is common to assume that a simplified model of the plant is adequate for much of the preliminary work. In the case of a motor drive plant, for instance, if a high-order model is available, the model can be simplified by assuming that parameters such as friction and/or inductance can be neglected. It is also possible to simplify the high-order model systematically by employing a model-reduction methods such as that presented in Chapter 8.

A simplified model for the velocity plant of a DC motor drive is often represented by the single integrator transfer function

$$G_p(s) = \frac{K_\omega}{s} \quad (7.3-16)$$

where K_ω denotes the velocity plant gain whose value can be determined analytically or experimentally (see Case Study in Chapter 4 and Chapter 8). Substituting Eq. (7.3-16) into the PDFF control system transfer functions given by Eq. (7.3-8) and Eq. (7.3-9) results in the velocity loop relationships

$$\frac{\Omega(s)}{R(s)} = \frac{K_\omega(K_A s^2 + K_V s + K_I)}{s^2 + (K_D s^2 + K_P s + K_I)K_\omega} \quad (7.3-17)$$

and

$$H_\omega(s)G(s) = \frac{K_\omega(K_A s^2 + K_V s + K_I)}{s[s + (K_D s + K_P - K_A s - K_V)K_\omega]} \quad (7.3-18)$$

In addition, from Eq. (7.2-3), the output-to-disturbance transfer function becomes

$$\frac{\Omega(s)}{D(s)} = \frac{sK_{\omega}}{s^2 + (K_D s^2 + K_P s + K_I)K_{\omega}} \quad (7.3-19)$$

From Eq. (7.3-17), it is easy to see that unless $K_A=0$, the velocity response will have an impulse at the origin, an undesirable characteristic. Also, when the denominators of Eq. (7.3-17) - Eq. (7.3-19) are normalized, the common factor $1/(1+K_{\omega}K_D)$ suggests that we can fix K_D at any convenient value. For instance, using $K_D=0$ yields the following velocity loop relationships:

$$\frac{\Omega(s)}{R(s)} = \frac{K_{\omega}(K_V s + K_I)}{s^2 + K_{\omega}K_P s + K_{\omega}K_I} \quad (7.3-20)$$

$$H_{\omega}(s)G(s) = \frac{K_{\omega}(K_V s + K_I)}{s[s + K_{\omega}(K_P - K_V)]} \quad (7.3-21)$$

and

$$\frac{\Omega(s)}{D(s)} = \frac{sK_{\omega}}{s^2 + K_{\omega}K_P s + K_{\omega}K_I} \quad (7.3-22)$$

These transfer functions demonstrate the flexibility of the PDF control algorithm. If $K_V=K_P$, the transfer functions are identical to those corresponding to the conventional PID control system. If $K_V=0$, the transfer functions are the same as those of the PDF control system and if $0 < K_V < K_P$, the PDF control system performance characteristics can be set anywhere between those of the PID and PDF control systems. The transfer functions of the PID and PDF control systems for a single integrator plant are summarized in Table 7.3-3. Note that the gain K_{ω} has been replaced by K in the table.

Table 7.3-3 Relationships for PID & PDF control systems; $G_p(s) = \frac{K}{s}$

Relationship	PID Control	PDF Control
$\frac{C(s)}{R(s)}$	$\frac{KK_P s + KK_I}{s^2 + KK_P s + KK_I}$	$\frac{KK_V s + KK_I}{s^2 + KK_P s + KK_I}$
$\frac{C(s)}{D(s)}$	$\frac{Ks}{s^2 + KK_P s + KK_I}$	$\frac{Ks}{s^2 + KK_P s + KK_I}$
$\frac{E(s)}{R(s)}$	$\frac{s^2}{s^2 + KK_P s + KK_I}$	$\frac{s^2 + KK_P s - KK_V s}{s^2 + KK_P s + KK_I}$
$H(s)G(s)$	$\frac{KK_P s + KK_I}{s^2}$	$\frac{KK_V s + KK_I}{s^2 + KK_P s - KK_V s}$

7.3-2 PDFF Control of the Double Integrator Plant

A simplified model of a DC motor drive plant considered often in the design of position control systems is given by the double integrator transfer function

$$G_p(s) = \frac{K_\theta}{s^2} \quad (7.3-23)$$

where K_θ represents the position plant gain whose value, like K_ω , can be determined analytically or experimentally. Substituting Eq. (7.3-23) into the PDFF control system transfer functions, we obtain the position loop relationships

$$\frac{\Theta(s)}{R(s)} = \frac{K_\theta(K_A s^2 + K_V s + K_I)}{s^3 + K_\theta K_D s^2 + K_\theta K_P s + K_\theta K_I} \quad (7.3-24)$$

$$H_\theta(s)G(s) = \frac{K_\theta(K_A s^2 + K_V s + K_I)}{s[s^2 + K_\theta(K_D - K_A)s + K_\theta(K_P - K_V)]} \quad (7.3-25)$$

and

$$\frac{\Theta(s)}{D(s)} = \frac{K_\theta s}{s^3 + K_\theta K_D s^2 + K_\theta K_P s + K_\theta K_I} \quad (7.3-26)$$

Appropriate relationships can also be derived for the PID and PDF control systems by applying the relevant conditions. Specifically, the PID control relationships are obtained by substituting $K_A = K_D$ and $K_V = K_P$ in the previous equations. This results in the PID transfer function expressions:

$$\frac{\Theta(s)}{R(s)} = \frac{K_\theta(K_D s^2 + K_P s + K_I)}{s^3 + K_\theta K_D s^2 + K_\theta K_P s + K_\theta K_I} \quad (7.3-27)$$

and

$$H_\theta(s)G(s) = \frac{K_\theta(K_D s^2 + K_P s + K_I)}{s^3} \quad (7.3-28)$$

with the $\Theta(s)/D(s)$ transfer function being the same as that given by Eq. (7.3-26). The transfer functions of the PID and PDFF control systems for a double integrator plant are summarized in Table 7.3-4. Note that the gain K_θ has been replaced by K in the table.

7.3-3 Tuning of Control Algorithms by Standard Filter Forms

Since control systems are inherently time domain systems, their transient and steady-state performance characteristics can be adjusted to match the characteristics of standard filter forms. The standard forms for the low-pass Butterworth, Binomial, Bessel, and ITAE filters were introduced in Chapter 5. Their respective coefficients are given in Tables 5.3-1, 5.3-2, 5.3-3 and 5.3-4.

Table 7.3-4 Relationships for PID & PDFF control systems; $G_p(s) = \frac{K}{s^2}$

Relationship	PID Control	PDFF Control
$\frac{C(s)}{R(s)}$	$\frac{KK_D s^2 + KK_P s + KK_I}{s^3 + KK_D s^2 + KK_P s + KK_I}$	$\frac{KK_A s^2 + KK_V s + KK_I}{s^3 + KK_D s^2 + KK_P s + KK_I}$
$\frac{C(s)}{D(s)}$	$\frac{Ks}{s^3 + KK_D s^2 + KK_P s + KK_I}$	$\frac{Ks}{s^3 + KK_D s^2 + KK_P s + KK_I}$
$\frac{E(s)}{R(s)}$	$\frac{s^3}{s^3 + KK_D s^2 + KK_P s + KK_I}$	$\frac{s^3 + K(K_D - K_A)s^2 + K(K_P - K_V)s}{s^3 + KK_D s^2 + KK_P s + KK_I}$
$H(s)G(s)$	$\frac{KK_D s^2 + KK_P s + KK_I}{s^3}$	$\frac{KK_A s^2 + KK_V s + KK_I}{s^3 + K(K_D - K_A)s^2 + K(K_P - K_V)s}$

It was stated in Section 5.3, that if the filter coefficients are used in the transfer function

$$\frac{C(s)}{R(s)} = \frac{b_n}{s^n + b_1 s^{n-1} + \dots + b_{n-2} s^2 + b_{n-1} s + b_n} \quad (7.3-29)$$

the resulting system is referred to as a **zero-displacement-error (ZDE)** system, because its transfer function can only track (follow) step inputs with zero steady-state error. If zero steady-state error due to a ramp input is desired, on the other hand, the appropriate transfer function is

$$\frac{C(s)}{R(s)} = \frac{b_{n-1} s + b_n}{s^n + b_1 s^{n-1} + \dots + b_{n-2} s^2 + b_{n-1} s + b_n} \quad (7.3-30)$$

Systems represented by Eq. (7.3-30) are known as **zero-velocity-error (ZVE)** systems. Finally, to track a parabolic input, the proper transfer function for a **zero-acceleration-error (ZAE)** system is

$$\frac{C(s)}{R(s)} = \frac{b_{n-2} s^2 + b_{n-1} s + b_n}{s^n + b_1 s^{n-1} + \dots + b_{n-2} s^2 + b_{n-1} s + b_n} \quad (7.3-31)$$

In addition to the standard forms for the Butterworth, Binomial and ITAE filters, Graham and Lathrop have also determined the coefficients that minimize the ITAE filter criterion for the ZVE and ZAE systems. A partial list of the relevant coefficients, in normalized polynomial form, is given in Table 7.3-1 and Table 7.3-2. Note that s must be replaced by s/ω_c in the normalized transfer function expressions, where ω_c is the corner/cutoff frequency.

Table 7.3-1 Minimum ITAE Zero-Velocity-Error (ZVE) forms

n	Denominators
2	$s^2 + 3.2s + 1$
3	$s^3 + 1.75s^2 + 3.25s + 1$
4	$s^4 + 2.41s^3 + 4.93s^2 + 5.14s + 1$
5	$s^5 + 2.19s^4 + 6.5s^3 + 6.3s^2 + 5.24s + 1$

Table 7.3-2 Minimum ITAE Zero-Acceleration-Error (ZAE) forms

n	Denominators
3	$s^3 + 2.97s^2 + 4.94s + 1$
4	$s^4 + 3.71s^3 + 7.88s^2 + 5.93s + 1$
5	$s^5 + 3.81s^4 + 9.94s^3 + 13.44s^2 + 7.36s + 1$

To tune a digital PDF/PDFF control system, the following algorithm can be used.

- (1) Depending on the problem requirements, determine whether a PDF or a PDFF controller is appropriate.
- (2) Set the denominator coefficients of the closed-loop system transfer function equal to the coefficients in the corresponding table of standard filter forms.
- (3) Once the analog controller gains have been determined, the digital controller gains can be calculated using the sampling time, plant gain, and appropriate relationships.

The method is illustrated using examples of control systems incorporating the plain/single integrator and double integrator as plants.

Plain Integrator Plant: The objective is to tune a PDF algorithm to control a plain integrator plant in order to achieve, for instance, performance characteristics similar to the characteristics of a known filter. For a single integrator plant, the appropriate transfer function is expressed by Eq. (7.3-20). Then, for a specific filter response, the denominator polynomial of the transfer function in Eq. (7.3-20) is set equal to the proper filter denominator. Thus, for second-order Bessel filter response characteristics, the relevant equation becomes

$$s^2 + K_{\omega}K_Ps + K_{\omega}K_I = s^2 + \sqrt{3}\omega_c s + \omega_c^2 \quad (7.3-32)$$

where ω_c represents the control system corner/cutoff frequency. Equating coefficients in the above equation, specifies the analog controller gains as

$$K_P = \frac{\sqrt{3}\omega_c}{K_{\omega}} \quad (7.3-33)$$

and

$$K_I = \frac{\omega_c^2}{K_{\omega}} \quad (7.3-34)$$

Then, from Eq. (7.2-9) and Eq. (7.2-10), the digital controller gains, denoted by lower-case subscripts, are

$$K_p = \frac{\sqrt{3}\omega_c}{K_\omega} \quad (7.3-35)$$

and

$$K_i = TK_I = \frac{T\omega_c^2}{K_\omega} \quad (7.3-36)$$

Note that if Butterworth or ITAE filter response characteristics were selected, instead of Bessel, the integral gain stays the same, but the relation for the proportional gain becomes

$$K_p = \frac{\sqrt{2}\omega_c}{K_\omega} \quad (7.3-37)$$

Unit-step responses versus time (nondimensional, i.e., $\omega_n t = 1$) for the second-order binomial, Bessel, and Butterworth/ITAE low-pass filter forms are shown in Fig. 7.3-3.

It is worth noting that the second-order Butterworth/ITAE step response has some overshoot, while the overshoot of the Bessel step response is smaller. Comparing Eq. (7.3-35) and Eq. (7.3-37) suggests that *increasing the proportional gain has the effect of decreasing or "squashing" any overshoot that may be present in the step response*. The inverse is also true, that is, *decreasing the proportional gain increases the overshoot*. Such observations can be helpful when tuning digital controllers in real time.

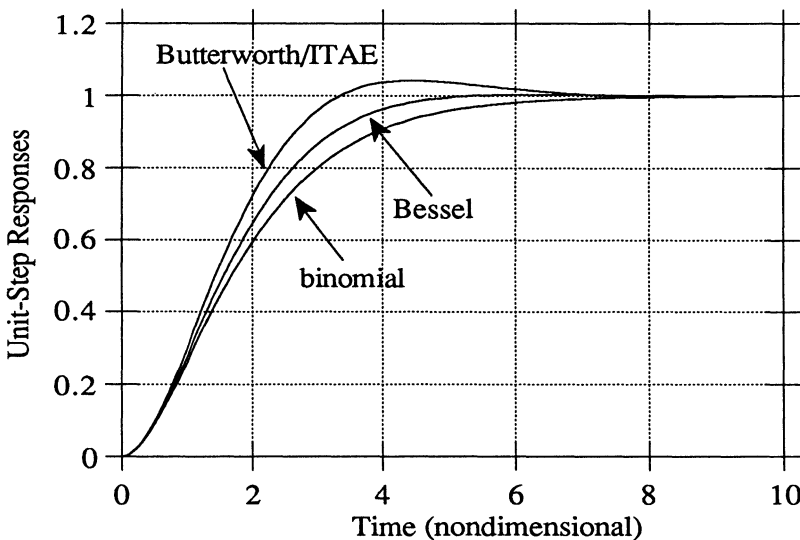


Fig. 7.3-3 Unit-step responses of 2nd-order standard filter forms.

Example 7.3-1

A plain integrator plant is described by the transfer function

$$G(s) = \frac{20}{s}$$

Design digital PDF and PDFF controllers for a sampling time of $T=0.005$ sec and a cutoff frequency of 70.71 rad/sec. Assume a Butterworth filter type of response for the control system.

From Eq. (7.3-37) and Eq. (7.3-34), the respective analog PDF controller gains are

$$K_P = \frac{100}{20} = 5, \text{ and } K_I = \frac{5000}{20} = 250$$

For a sampling time of $T=0.005$ sec, the corresponding digital controller gains are obtained from Eq. (7.3-33) and Eq. (7.3-36) as

$$K_p = 5 \text{ and } K_i = \frac{250}{200} = 1.25$$

The design of the digital PDF controller is complete. The only other variable to be adjusted is the proportional (feedforward) gain K_v . For instance, if

$$K_v = K_p = 5$$

the PDFF control system behaves like a conventional PID control system - without derivative action. Such a control algorithm is usually referred to as proportional-integral (PI), rather than PID. If $0 < K_v < 5$, various responses between the PID and PDF types can be obtained. The analog form of the control system of this example was analyzed previously in Sections 2.6 and 2.7. The digital control system version was analyzed in Section 3.7.

Double Integrator Plant: Tuning a PDFF control algorithm for a double integrator plant implies matching denominator coefficients between the transfer function of Eq. (7.3-24) and the appropriate filter transfer function. For instance, for a zero-displacement-error (ZDE) binomial filter response, the digital PDFF controller gains are given by

$$K_p = \frac{3\omega_c^2}{K_\theta} \quad (7.3-38)$$

$$K_i = \frac{T\omega_c^3}{K_\theta} \quad (7.3-39)$$

and

$$K_d = \frac{3\omega_c}{TK_\theta} \quad (7.3-40)$$

with $K_a=0$ and $K_v=0$.

Alternatively, if a minimum ITAE zero-acceleration-error (ZAE) filter response is desired, the digital PDFF controller gains are calculated from the relationships

$$K_p = \frac{4.94\omega_c^2}{K_\theta} \quad (7.3-41)$$

$$K_i = \frac{T\omega_c^3}{K_\theta} \quad (7.3-42)$$

$$K_d = \frac{2.97\omega_c}{TK_\theta} \quad (7.3-43)$$

with

$$K_v=K_p, \text{ and } K_a=K_d \quad (7.3-44)$$

7.4 CASE STUDY: DIGITAL CONTROL OF A MOTOR DRIVE PLANT

The main objective of this section is to design and implement appropriate control algorithms for the DC motor drive plant analyzed in the case studies of Chapter 4 and Chapter 8. A secondary objective is to develop practical models for the motor drive velocity and position plants. These models can be used to verify design results analytically or by computer simulation.

7.4-1 Velocity and Position Plant Models

A block diagram of the plant to be controlled, including the PG and DAC interfaces, is shown in Fig. 7.4-1. As can be seen from the diagram, the position of the motor shaft is monitored by a pulse generator (PG), which returns change of position (velocity) per sampling time T , and the controller output is communicated to the analog plant via a DAC.

It is shown in Section 8.4 that this analog position plant can be approximated by the double integrator transfer function

$$G_p(s) = \frac{\Theta(s)}{I_r(s)} = \frac{K_m}{s^2} \quad (7.4-1)$$

where the analog position plant gain K_m depends on motor drive parameters. If the DAC and PG interfaces are taken into consideration, as shown in Fig. 7.4-1, then the digital position plant transfer function is found to be

$$\frac{\Theta(z)}{M(z)} = \mathcal{Z}\left[\frac{1-e^{-sT}}{s} \frac{K_\theta}{s^2}\right] = \frac{\left(\frac{K_\theta T^2}{2}\right)(z+1)}{(z-1)^2} \quad (7.4-2)$$

where the digital position plant gain K_θ is given by

$$K_{\theta} = (K_{da})(K_m)(K_{dt}) \quad (7.4-3)$$

K_{θ} can be calculated from given motor drive and interface parameters, or it can be determined experimentally.

From Fig. 7.4-1, the digital velocity plant transfer function can be determined by differentiating the digital position plant transfer function. That is,

$$\frac{\Omega(z)}{M(z)} = \frac{\left(\frac{K_{\theta}T^2}{2}\right)(z+1)(z-1)}{z(z-1)^2} = \frac{\left(\frac{K_{\omega}T}{2}\right)(z+1)}{z(z-1)} \quad (7.4-4)$$

where K_{ω} is the digital velocity plant gain defined by

$$K_{\omega} = TK_{\theta} \quad (7.4-5)$$

Like K_{θ} , K_{ω} can also be determined analytically from motor drive parameter values, or experimentally (see Section 8.4).

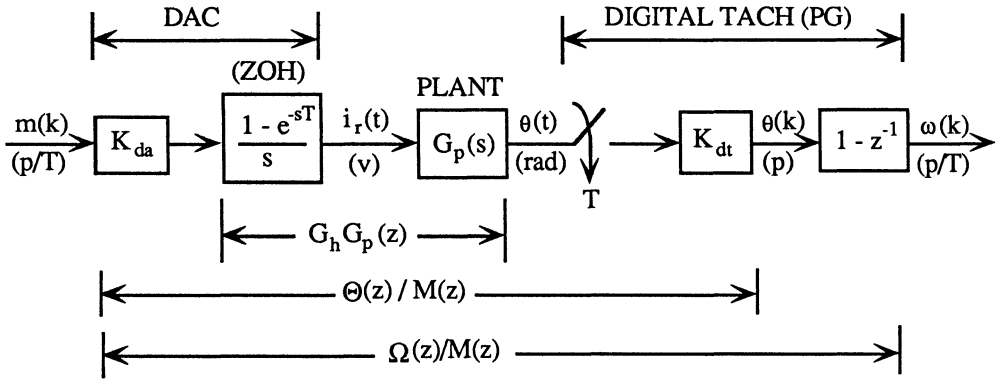


Fig. 7.4-1 Block diagram of plant, including DAC and PG interfaces.

Substituting the values of $K_{\theta}=1070$ and $T=0.005$ sec into Eq. (7.4-2), results in the position plant transfer function

$$\frac{\Theta(z)}{M(z)} = \frac{\left(\frac{K_{\theta}T^2}{2}\right)(z+1)}{(z-1)^2} = \frac{0.0134(z+1)}{(z-1)^2} \quad (7.4-6)$$

From Eq. (7.4-4), the velocity plant transfer function is

$$\frac{\Omega(z)}{M(z)} = \frac{\left(\frac{K_{\omega}T}{2}\right)(z+1)}{z(z-1)} = \frac{0.0134(z+1)}{z(z-1)} \quad (7.4-7)$$

These transfer function models can be used to test and/or simulate the motor drive plant. For instance, if a step of 204 pulses/T is applied to the velocity plant, the corresponding output transform is

$$\Omega(z) = \frac{0.0134(z+1)}{z(z-1)} \frac{204z}{(z-1)} = \frac{2.7336(z+1)}{(z-1)^2} \quad (7.4-8)$$

Then, the velocity plant step response can be evaluated using long division.

7.4-2 Digital Control of Velocity and Position Plants

It is required to perform the following tasks assuming a position plant gain of $K_{\theta}=1070$ (calculated from motor drive data) and a sampling time of $T=0.005$ sec.

- (1) Design and implement a PDF algorithm to control the shaft velocity of the DC motor drive plant. Assume a cutoff frequency of $\omega_c=50$ rad/sec, and a trapezoidal velocity profile with acceleration time of 0.15 sec.
 - (a) Use the zero-displacement-error (ZDE) criterion for Bessel filter response
 - (b) Repeat part (1)-(a) except add the feedforward control action $K_v=K_p/3$
 - (c) Repeat part (1)-(a) except make the feedforward gain $K_v=K_p$ (i.e. PI control)
- (2) Design and implement a PDF algorithm to control the shaft position of the DC motor drive plant. Assume a cutoff frequency of $\omega_c=30$ rad/sec, and a trapezoidal velocity profile with acceleration time of 0.15 sec. For tuning, use the following criteria.
 - (a) Minimum ITAE zero-displacement-error (ZDE)
 - (b) Minimum ITAE zero-velocity-error (ZVE)
 - (c) Minimum ITAE zero-acceleration-error (ZAE)

A real-time program is written to implement the digital PDF control system diagram shown in Fig. 7.4-2. In addition to "conventional" terminology, there are some items in Fig. 7.4-2 that need clarification.

The block labeled "filter" with transfer function $F(z)$ can be used to implement a digital filter when the quantization noise in the derivative (K_d) path becomes excessive (see Section 7.4-3).

The block marked "software" denotes software integration if the value of the parameter $K_I=1$, which is the default value. If $K_I=0$, the software integrator is essentially bypassed, which is desirable if velocity rather than position is to be controlled.

Furthermore, if $0 < K_r < 1$, the "software" integrator is changed into a first-order lag term, which can be useful if the position plant model is not a double integrator but a first-order lag followed by a single integrator.

The $K_j(1-z^{-1})^2$ term in the feedforward block represents a second-order derivative (see Section 7.4-4), not a welcome sight in practice. If desired, this term can also be added to the PID control system as a feedforward branch.

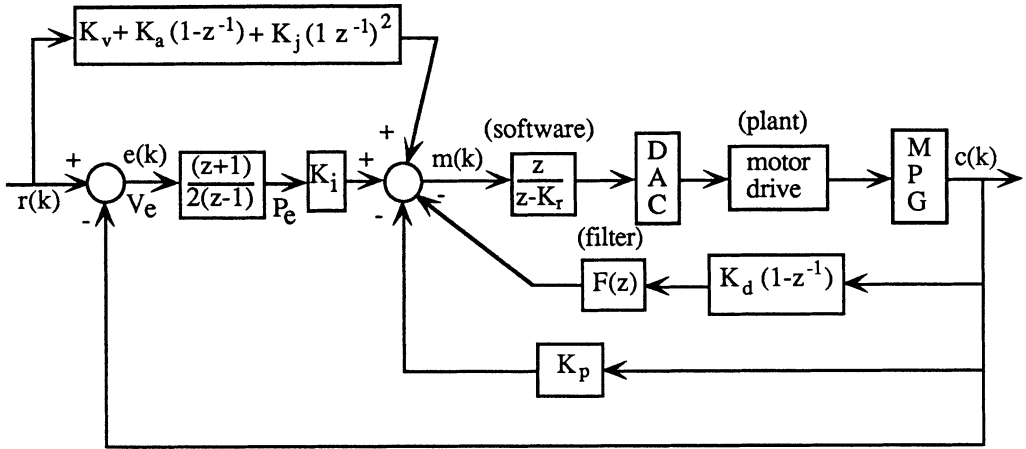


Fig. 7.4-2 Block diagram of the digital PDF control system.

In the case studies of Chapters 4 and 8, it has been shown that this analog position plant behaves like a double integrator. Since the PG differentiates and thus eliminates one of the integrators, the software integrator can be viewed as the integrator that replaces the one lost to the PG. Finally, the parameters V_e and P_e denote velocity and position errors, which are important quantities to monitor and/or limit when controlling motor drive plants. An advantage of the block diagram in Fig. 7.4-1 is that it can be used to control either velocity or position, using velocity command and feedback. It can be easily modified, however, for position command and/or feedback.

(1) Velocity Control ($K_r=0$)

The velocity plant is described by the transfer function

$$G_\omega(s) = \frac{K_\omega}{s}, \text{ where, for this case, } K_\omega = TK_\theta = 0.005 \cdot 1070 = 5.35$$

- (a) Using the Bessel-ZDE criterion with $\omega_c = 50$ rad/sec, we obtain the PDFF controller gains as

$$K_p = \frac{\sqrt{3}\omega_c}{K_\omega} = 16.187 \text{ and } K_i = \frac{T\omega_c^2}{K_\omega} = 2.34, \text{ with } K_v = 0 \text{ and } K_d = 0$$

- (b) Using the Bessel-ZDE criterion with $K_v = \frac{K_p}{3}$, the gains are
 $K_p = 16.187$, $K_i = 2.34$, $K_v = \frac{K_p}{3} = 5.4$ and $K_d = 0$
- (c) Using the Bessel-ZDE criterion with $K_v = K_p$, that is, for PI control, the gains are
 $K_p = 16.187$, $K_i = 2.34$, $K_v = 16.187$ and $K_d = 0$

This motor drive plant was controlled in the laboratory by a microcomputer programmed in the C language. The relevant signals were "captured" in real-time and downloaded into a data file for plotting. Results for case (1)-(a) are shown in Fig. 7.4-3, for case (1)-(b) are shown in Fig. 7.4-4, and for case (1)-(c) in Fig. 7.4-5.

A note concerning the 200 (pulse/T) for the step size of the trapezoid shown in the figures. The rated speed of this motor drive is 1000 RPM. Thus, for a sampling time of $T = 0.005$ sec, the rated speed in (pulses/T) is found to be

$$1000 \text{ (RPM)} * 8000 \text{ (pulses/rev)} * \frac{0.005 \text{ sec}}{T} = 667 \text{ (pulses/T)}$$

Therefore, the 200 (pulse/T) used in this example corresponds to 300 RPM.

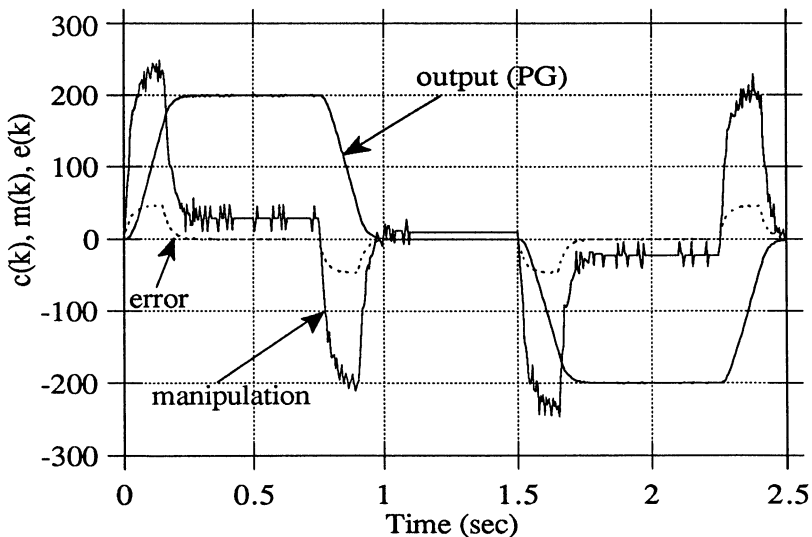


Fig. 7.4-3 Digital PDFF velocity system without feedforward action.

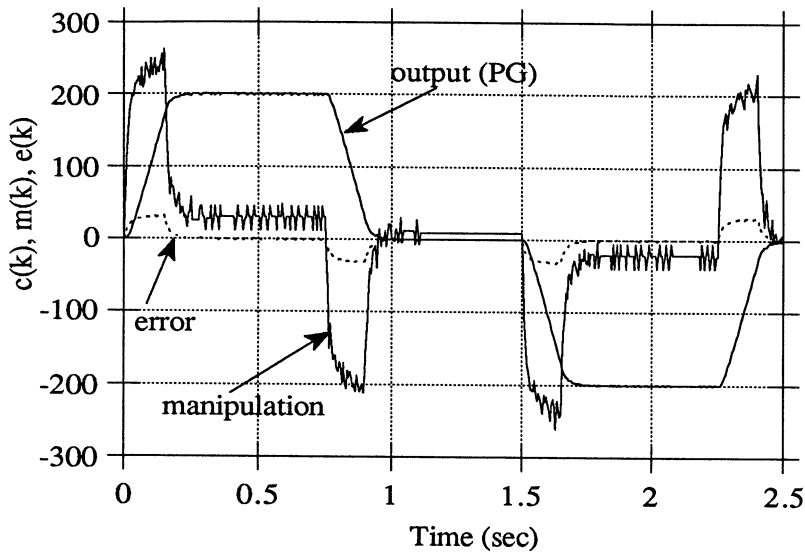


Fig. 7.4-4 Digital PDFF velocity system with $K_v = K_p/3$ added.

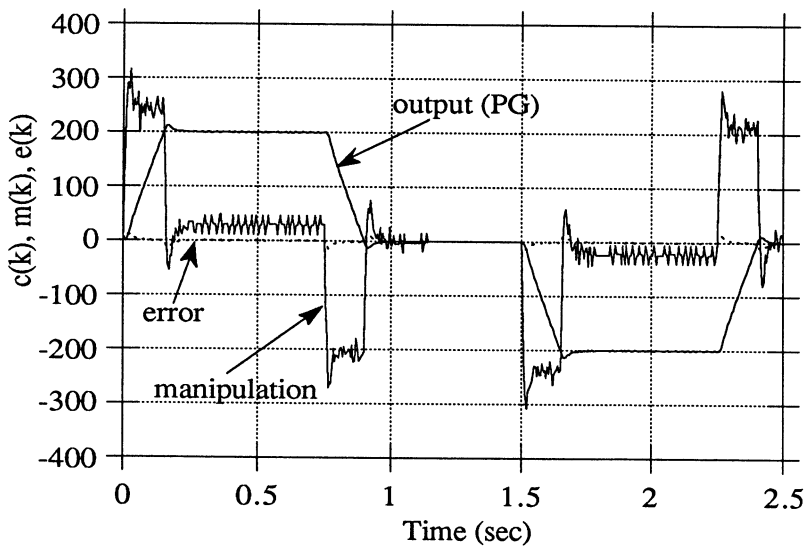


Fig. 7.4-5 Digital PDFF velocity system with $K_v = K_p$, PI control.

(2) Position Control ($K_r=1$)

The position plant is described by the transfer function

$$G_\theta(s) = \frac{K_\theta}{s^2} = \frac{1070}{s^2}$$

- (a) Using the ITAE-ZDE criterion with $\omega_c=30$ rad/sec, we obtain the PDFF controller gains for the position loop as

$$K_p = \frac{2.15\omega_c^2}{K_\theta} = 1.81, K_i = \frac{T\omega_c^3}{K_\theta} = 0.126 \text{ and } K_d = \frac{1.75\omega_c}{K_\theta} = 9.8$$

Note that $K_v=0$ and $K_a=0$.

- (b) Using the ITAE-ZVE criterion and Table 7.3-1, the gains are

$$K_p=K_v=2.733, K_i=0.126, K_d=9.8 \text{ and } K_a=0.$$

- (c) Using the ITAE-ZDE criterion and Table 7.3-2, the gains are

$$K_p=K_v=4.155, K_i=0.126, K_d=K_a=16.65 \text{ (PID control!)}$$

Experimental results for these three cases are shown in Fig. 7.4-6, Fig. 7.4-7 and 7.4-8, respectively. For comparison purposes, the respective position errors are shown in Fig. 7.4-9. The ITAE-ZAE case, that is, case (2)-(c), which also corresponds to the conventional PID controller case, has the smallest position error but its response overshoots. This behavior may not be acceptable in certain applications where, for instance, overshoot may be associated with gear backlash. If any of the responses needs modification, this "fine-tuning" is usually done in real-time by adjusting the appropriate controller gains.

The tuning scheme proposed worked well for the motor drive plant of this example. But, what if the position plant did not behave like a double integrator? Or, what if the standard filter forms presented here did not satisfy performance requirements? What if the person called upon to tune a PDFF control system out in the field does not have analytical or computer skills? Is there a practical alternative for tuning such a control system?

Looking at the PDFF block diagram of Fig. 7.4-2, one can see that tuning the PDFF controller implies adjusting several gains, a rather complex task. A more practical and systematic alternative is to tune the control system by adjusting the controller gains one at a time, using an "inside-out" tuning approach. Using this approach, first, set $K_p=K_i=K_v=0$ and apply the reference input via the K_a path. Then adjust K_d for desired response characteristics. After K_d has been set, and with $K_i=K_a=0$, adjust K_p by applying the input via the K_v path. Finally, adjust K_i without the feedforward gains. Any feedforward control action(s) can be added afterwards.

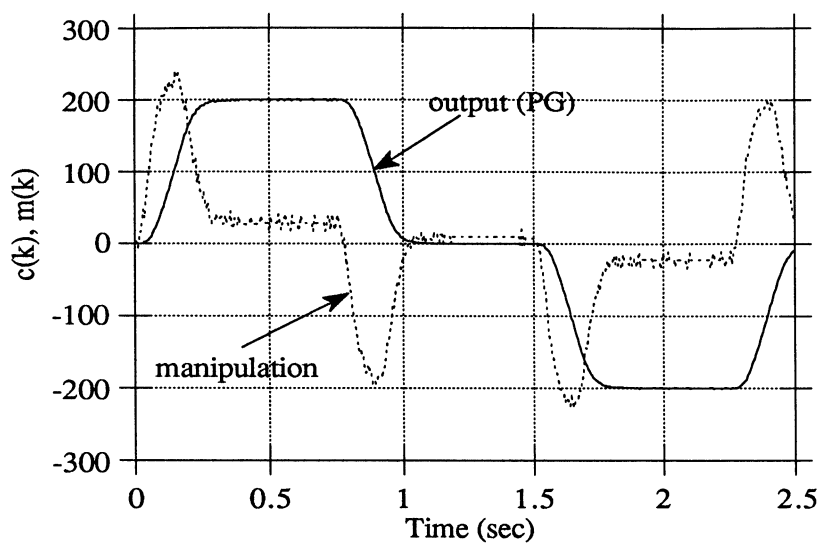


Fig. 7.4-6 Performance of ITAE-ZDE position control system.

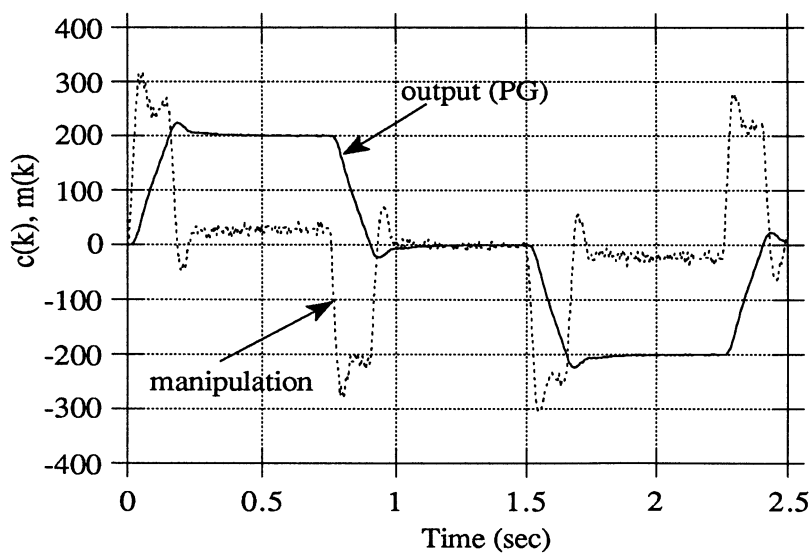


Fig. 7.4-7 Performance of ITAE-ZVE position control system.

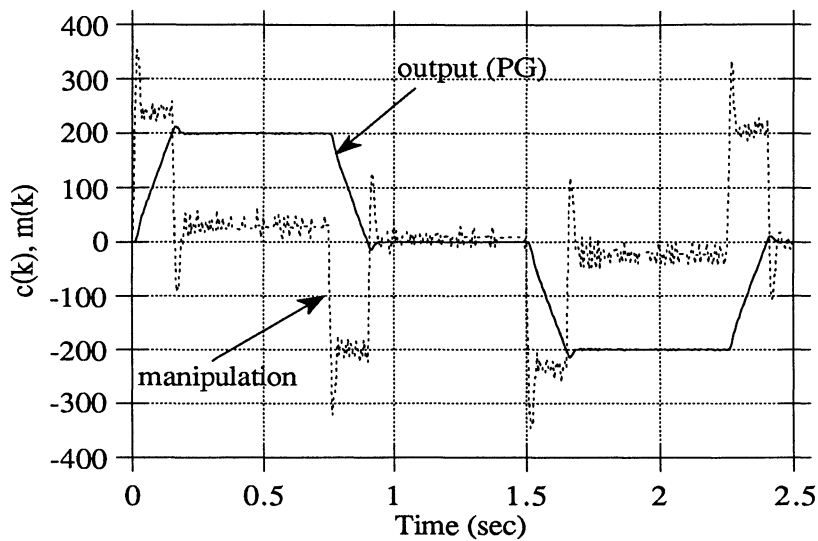


Fig. 7.4-8 Performance of ITAE-ZAE position control system.

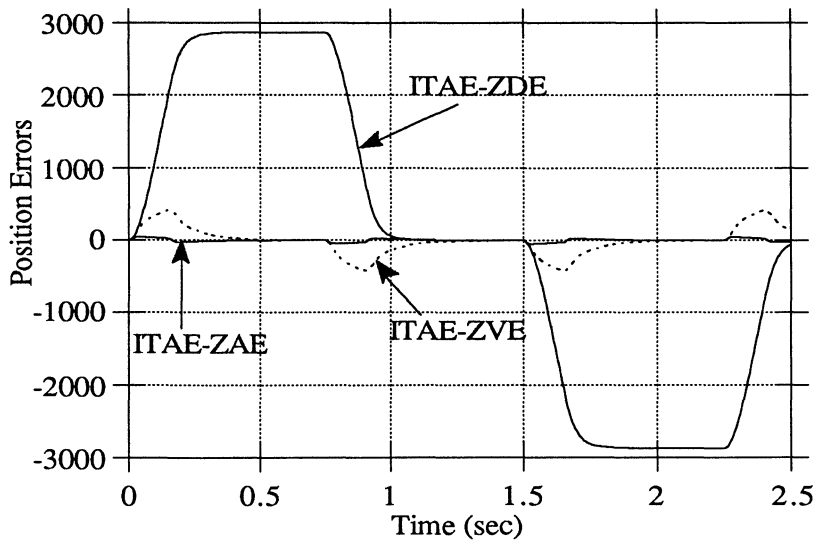


Fig. 7.4-9 Position errors of digital PDFF position control systems.

7.4-3 Filtering the Derivative (K_d) Path

Quantization noise created by interface devices of finite wordlength can adversely affect the performance of a digital control system. Digital filtering schemes for minimizing the effect of quantization noise, usually present in the derivative path of the control algorithm, can be used effectively. The main objective of this section is to design and implement a practical filter that will minimize the effect of quantization noise on the performance of a digital control system.

Since many computer controlled systems include an analog plant and a digital processor, it becomes necessary to convert the analog feedback signal into digital form by means of devices such as the incremental encoder or pulse generator (PG), which is used for many motor shaft position-sensing applications. Another interface device is the analog-to-digital converter (ADC), which quantizes the incoming signal and encodes it into a digital word of finite wordlength.

The error caused by quantizing the feedback signal becomes worse when the signal is differentiated. Digital differentiation is, of course, necessary in order to implement algorithms such as the PID or the PDFF. In motor control applications, digital differentiation can cause excessive ripple in the current or torque command signal, which results in undesirable audible noise and/or heating problems. This is particularly true in high-performance applications which require relatively high sampling frequencies (low sampling times). In such cases, some form of filtering may become necessary to limit the derivative gain and, thus, reduce the effect of digital noise.

Derivative action is usually included in control algorithms to compensate for excessive phase lag. With digital algorithms, derivative action can help improve the performance of the control system whenever the plant dead-time and computational delays become significant relative to the sampling time.

The control system considered here is the digital PDFF control system shown in Fig. 7.4-2. In the block diagram, MPG represents the incremental encoder (PG) which is used to measure the motor shaft velocity in pulses/T. The control algorithm uses this measure to calculate the manipulation action and output it as current command to the amplifier of the motor drive plant via the DAC. Having the derivative action in the feedback path helps minimize the proportional and derivative "kicks" on the error signal.

The feedback signal contains usual plant noise as well as truncation noise due to the nature of the PG. The PG-generated noise is amplified when the quantized feedback signal is differentiated. Since the derivative gain, K_d , is proportional to the sampling frequency (inversely proportional to the sampling time T), digital differentiation becomes a problem for high-frequency applications. In such cases, filtering the derivative path signal may prove to be a practical and effective remedy.

Both IIR and FIR filters may be considered. The type of filter emphasized in this section is attractive for being cost-effective and practical. That is, the filter form is kept relatively simple in order to minimize the computation time required for implementation. Furthermore, it is assumed that the control system is already tuned and inserting a digital filter in the derivative path should not require re-tuning of the control algorithm.

The simplest form for a derivative path IIR filter is the first-order, low-pass, unity-gain filter with analog prototype transfer function given by

$$F(s) = \frac{D(s)}{Q(s)} = \frac{\omega_c}{s + \omega_c} \quad (7.4-9)$$

where ω_c is the cutoff or corner frequency in rad/sec. Clarke has shown that the analog cutoff frequency ω_c is related to the digital derivative gain K_d by the expression

$$\omega_c = \frac{10}{TK_d} \quad (7.4-10)$$

where T is the control sampling time. The factor of 10 is generally acceptable, but it can be adjusted (tuned) in real time, like the other algorithm gains, if desired.

Using the analog filter transfer function $F(s)$ as a starting point, the next objective is to determine the corresponding digital filter transfer function, $F(z)$, which closely approximates the performance of its analog counterpart.

There are several techniques available for approximating an analog filter transfer function by a digital filter transfer function that satisfies the stated requirements. As it was shown in Chapter 5, a particular technique is usually selected on the basis of the desired impulse response, step response and frequency response characteristics.

One of the more effective methods for designing digital filters is the bilinear transformation method. For the analog filter transfer function given by Eq. (7.4-9), the prewarped analog frequency is given by

$$\omega = \tan \left(\frac{\omega_d T}{2} \right) \quad (7.4-11)$$

where ω_d is the cutoff frequency of the digital filter. The desired first-order digital filter transfer function can be calculated from the relation

$$F(z) = \frac{\tan(\frac{\omega_d T}{2})}{s + \tan(\frac{\omega_d T}{2})} \bigg|_{s=\frac{z-1}{z+1}} \quad (7.4-12)$$

The digital filter design and implementation concepts are illustrated by the following example.

Example 7.4-1

Design and implement a PDFF algorithm to control the shaft position of the DC motor drive plant used in Section 7.4-2. Use a sampling time of $T=0.001$ sec and assume Bessel-ZDE filter response characteristics.

The controller gains for this case are found to be

$$K_p = 2.07, \quad K_i = 0.025 \quad \text{and} \quad K_d = 68$$

Note that the top speed (step size of the trapezoid) must be reduced by a factor of 5, from 200 to 40 (pulses/T), because T has been reduced from 5 to 1 msec.

Reference input and feedback signals obtained experimentally are shown in Fig. 7.4-10. As can be seen from the figure, the ripple in the feedback signal has become appreciable and requires filtering.

From Eq. (7.4-10), the analog filter corner frequency is

$$\omega_c = \frac{10}{TK_d} = 147 \text{ rad/sec}$$

This is also the digital filter frequency ω_d . Using Eq. (7.4-11) and Eq. (7.4-122), leads to the digital filter transfer function

$$F(z) = \frac{Q(z)}{P(z)} = \frac{0.07(z+1)}{z-0.86}$$

where $q(k)$ is assumed to be the output and $p(k)$ the input of the digital filter.

For instance, if the feedback of the digital control system, $c(k)$, is a unit-step function, the response of the digital filter is

$$q(k) = \{4.76, 8.85, 7.61, 6.55, 5.63, 4.84, 4.17, 3.58, 3.08, 2.65, \dots\}$$

Thus, the initial "jump" or "kick" is 4.76 units, compared with 68 units (K_d value) for the unfiltered algorithm.

The analytical results were verified experimentally. Fig. 7.4-11 shows the reference input and velocity feedback signals with the digital filter $F(z)$ included in the derivative path. The controller tuning had not been altered.

7.4-4 A Second-Order Derivative (K_j) in the Feedforward Path

In the block diagram of the digital PDFF control system shown in Fig. 7.4-2, the feedforward block includes a second-order derivative term with a controller gain denoted by K_j . If this were an all analog control system with a double integrator plant given by

$$G_p(s) = \frac{K}{s^2} \quad (7.4-13)$$

it can be shown that the closed-loop transfer function is

$$\frac{C(s)}{R(s)} = \frac{KK_j s^3 + KK_A s^2 + KK_V s + KK_I}{s^3 + KK_D s^2 + KK_P s + KK_I} \quad (7.4-14)$$

Notice that, for simplicity, K is used in place of K_θ . Thus, if $K_A = K_D$, $K_V = K_P$, and $KK_j = 1$, output is equal to the input, which is nice but unreasonable to expect of a physical system.

Consider next the digital PDFF control system of Fig. 7.4-2 and assume that the K_j term is to be used. The last relationship, i.e., $KK_j = 1$, implies that the digital second-derivative gain K_j is given by

$$K_j = \frac{1}{KT^2} \quad (7.4-15)$$

The use of the K_j term is illustrated by the following example.

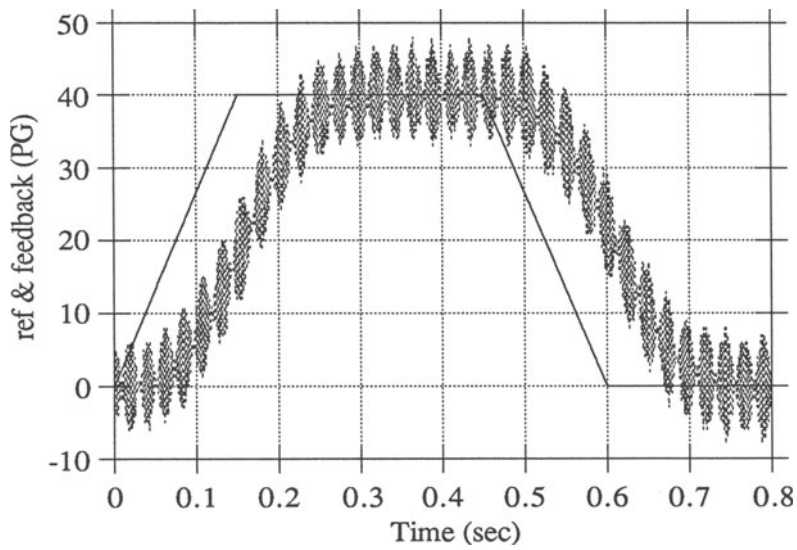


Fig. 7.4-10 Reference and feedback of 1 msec loop without K_d filter.

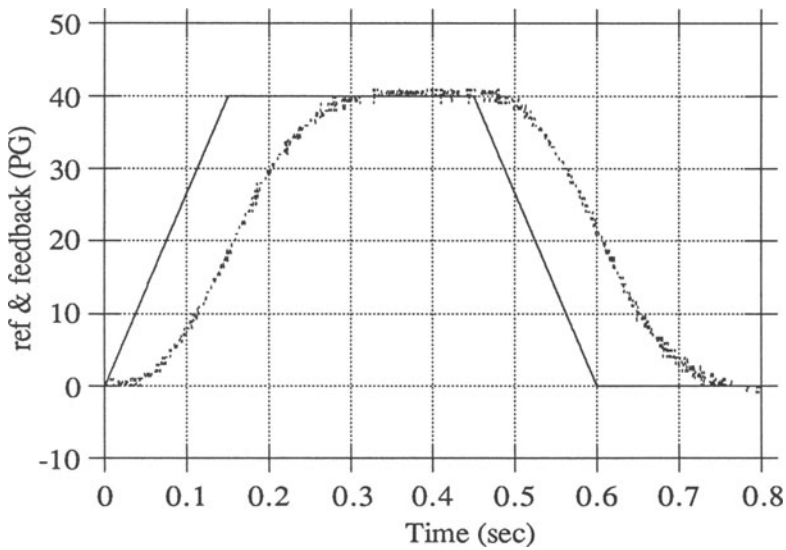


Fig. 7.4-11 Reference and feedback of 1 msec loop with K_d filter.

Example 7.4-2

A motor drive plant with DAC and PG interfaces has a position plant gain (measured in the laboratory) of $K_\theta=2380$. It is required to control the plant by a digital PDFF algorithm using a control sampling time of $T=0.001$ sec. Assume a control system corner frequency of 100 rad/sec, acceleration time of 0.15 sec, top speed of 150 pulses/T (i.e., 1800 RPM), and travel time at top speed of 0.3 sec. Determine maximum position error values for ITAE-ZDE, ITAE-ZVE, ITAE-ZAE and ITAE-ZAE-with- K_j filter responses.

The tuning values for the four cases are as follows:

- (1) ITAE-ZDE: $K_p=9.03$, $K_i=0.42$, $K_d=73.5$, $K_v=0$, $K_a=0$, $K_j=0$
- (2) ITAE-ZVE: $K_p=13.66$, $K_i=0.42$, $K_d=73.5$, $K_v=13.66$, $K_a=0$, $K_j=0$
- (3) ITAE-ZAE: $K_p=20.76$, $K_i=0.42$, $K_d=124.79$, $K_v=20.76$, $K_a=124.79$, $K_j=0$
- (4) ITAE-ZAE+ K_j : $K_p=20.76$, $K_i=0.42$, $K_d=124.79$, $K_v=20.76$, $K_a=124.79$, $K_j=1/(KT^2)=420.17$

The experimental results for case (1), ITAE-ZDE, are shown in Fig. 7.4-12. The experimental results for case (3), ITAE-ZAE and case (4), ITAE-ZAE-plus- K_j , are shown in Fig. 7.4-13 and Fig 7.4-14, respectively. As can be seen from the figures, the control seems rather tight, with evidence of ripple/noise. The K_j term has helped reduce tracking error, but only the practitioner can evaluate its effectiveness. The corresponding maximum position error values, recorded in real-time, are as follows:

SCHEME	ITAE-ZDE	ITAE-ZVE	ITAE-ZAE	ITAE-ZAE+ K_j
Pos. Error	3225 pulses	175 pulses	15 pulses	6 pulses

7.4-5 Time and Frequency Analysis of Digital Control Systems

The time and frequency response characteristics of the single integrator plant control system were evaluated in Chapter 3. For the control system with the double integrator plant, the plant pulse transfer function has been determined before and is given by

$$G_h G_p(z) = \left(\frac{K_\theta T^2}{2} \right) \frac{(z+1)}{(z-1)^2} \quad (7.4-16)$$

To determine the closed-loop transfer function $C(z)/R(z)$ of the PDFF control system of Fig. 7.4-2, one can use block diagram manipulation techniques or signal flow graphs and Mason's gain formula. The discrete transfer function can then be used to determine how closely the time and frequency response characteristics of the digital control system resemble the characteristics of their analog counterparts. Consequently, it can be checked whether the digital control system meets the problem requirements.

Assuming that $K_j=0$ and $K_r=1$, it can be shown that the transfer function of the double integrator plant digital control system is given by

$$\frac{C(z)}{R(z)} = \frac{Kz^{-1}[(K_i+K_v+K_a)+(K_i-K_a)z^{-1}-(K_v+K_a)z^{-2}+K_az^{-3}]}{\{1+[K(K_i+K_p+K_d)-3]z^{-1}+[K(K_i-K_d)+3]z^{-2}-[K(K_p+K_d)+1]z^{-3}+KK_dz^{-4}\}} \quad (7.4-17)$$

where

$$K = \frac{K_\theta T^2}{2} \quad (7.4-18)$$

From Eq. (7.4-17), the magnitude and phase response characteristics of the digital control system can be readily determined using a discrete frequency analysis program. If the step response of the system is desired, then substitute the expression $R(z)=z/(z-1)$ in Eq. (7.4-17) and use long division on the output transform $C(z)$ to find the time sequence $c(k)$.

The digital control system of Fig. 7.4-2 can also be simulated on a digital computer using languages such as ACSL or SIMULAB, or by writing a computer program directly from the system block diagram. For the latter approach, one can use either the analog plant model or its corresponding pulse transfer function model. The advantage of the simulation approach is that other signals in addition to input and output can be monitored and studied.

Example 7.4-3

A double integrator plant is described by the transfer function

$$G(s) = \frac{1070}{s^2}$$

Determine the frequency response characteristics of the corresponding digital PDFF control systems obtained for the following minimum ITAE criteria:

- (a) zero-displacement-error (ZDE)
- (b) zero-velocity-error (ZVE)
- (c) zero-acceleration-error (ZAE)

Assume a system cutoff frequency of $\omega_c=30$ rad/sec and a sampling time of $T=0.005$ sec.

For the frequency response characteristics of the digital PDFF control system, the appropriate parameter values can be substituted in Eq. (7.4-17) and the resulting transfer functions evaluated using the digital computer. The magnitude vs. frequency responses for the ZDE, ZVE and ZAE systems are shown in Fig. 7.4-15.

From the frequency responses, one can deduce that the feedforward term(s) increase the bandwidth of the control system. They also increase the possibility of overshoot. Thus, such systems can track reference inputs of higher frequencies, but their transient performance may not be acceptable because of the overshoot.

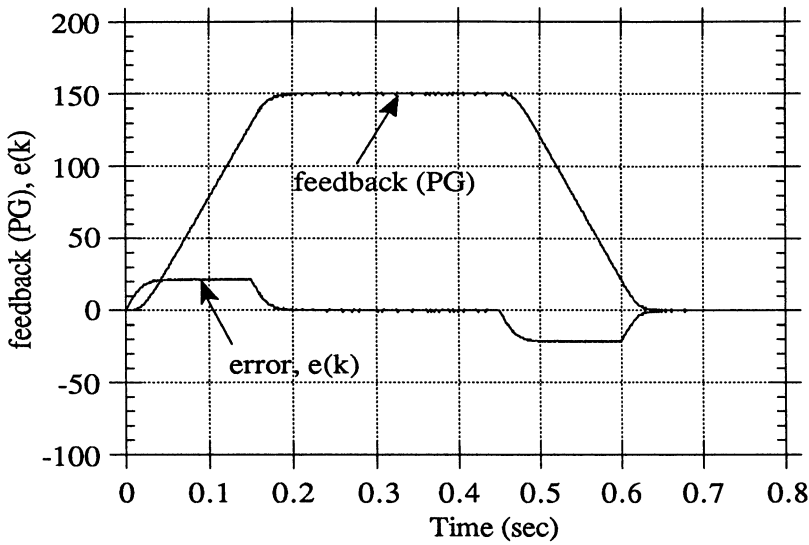


Fig. 7.4-12 PDFF position system of Example 7.4-2, tuned for ITAE-ZDE response.

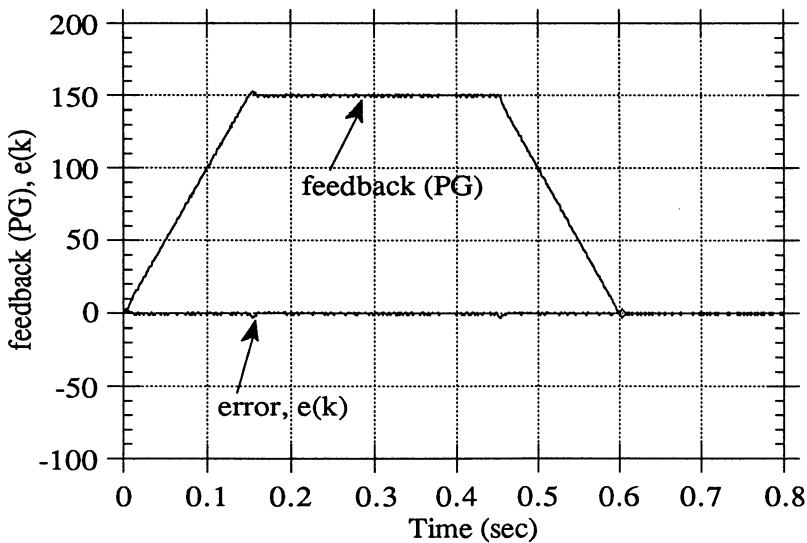


Fig. 7.4-13 PDFF position system of Example 7.4-2, tuned for ITAE-ZAE response.

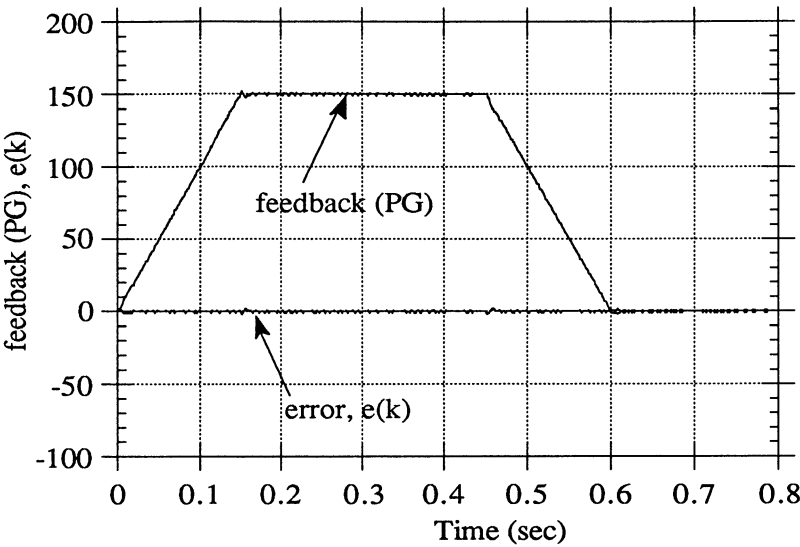


Fig. 7.4-14 PDFF position system of Example 7.4-2, tuned for ITAE-ZAE response with second-order derivative term K_j implemented.

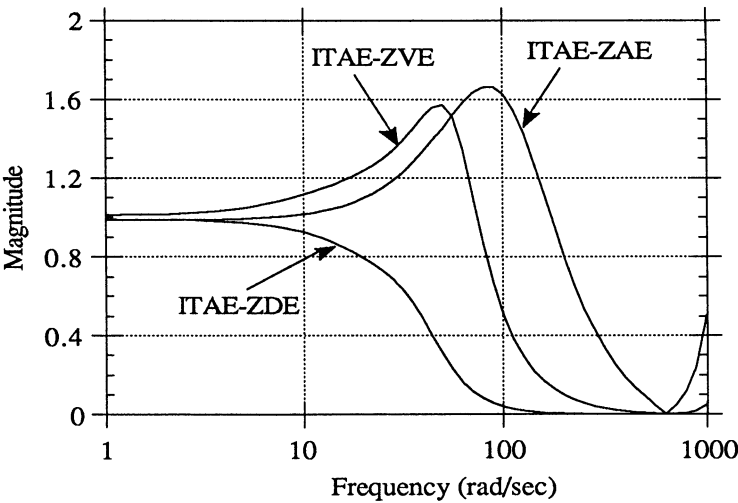


Fig. 7.4-15 Frequency characteristics of the digital PDFF control systems.

7.4-6 Steady-State Error Analysis of Digital Control Systems

In control system applications, the ability of a system to follow or track a given input with minimum or zero steady-state error is significant. The steady-state error is a measure of the difference between a system's input and output at steady state. Some applications require the steady-state error to be zero, while others may simply require that it has an upper bound. If there is no upper bound, the system is unacceptable because it cannot track the reference input.

From Section 2.6-5, the steady-state error due to a step function input of size R_o is given by

$$E_{ss}(\text{step}) = \frac{R_o}{1+C_p} \quad (7.4-19)$$

where the positional or displacement error constant C_p is given by

$$C_p = \lim_{s \rightarrow 0} H(s)G(s) \quad (7.4-20)$$

If the input is a ramp of slope R_o , the steady-state error is

$$E_{ss}(\text{ramp}) = \frac{R_o}{C_v} \quad (7.4-21)$$

where the velocity error constant C_v is given by

$$C_v = \lim_{s \rightarrow 0} sH(s)G(s) \quad (7.4-22)$$

Finally, if the input is parabolic, given by $(R_o t^2/2)$, then the steady-state error is

$$E_{ss}(\text{parabolic}) = \frac{R_o}{C_a} \quad (7.4-23)$$

where the acceleration error constant C_a is given by

$$C_a = \lim_{s \rightarrow 0} s^2 H(s)G(s) \quad (7.4-24)$$

The error constants C_p , C_v and C_a as well as the respective steady-state errors for the PID and PDFF control systems for various plants can be calculated from the block diagrams. For the plain or single integrator plant case, for instance, the PID control system is type 2, which implies that it can follow step and ramp inputs with zero steady-state error. The PDFF control system without feedforward gains is type 1 and, thus, it can follow step inputs with zero steady-state error and ramp inputs with finite steady-state error. When the feedforward gain K_v is added to the PDFF control system, however, it can follow ramp inputs with variable steady-state error when $K_v < K_p$ and zero steady-state error when $K_v = K_p$.

Using the open-loop transfer functions, the error constants and steady-state-error values can be readily obtained. Error transfer functions for single and double integrator plants are shown in Table 7.3-3 and Table 7.3-4, respectively.

From the corresponding open-loop transfer functions, the error constants and steady-state error values for the single integrator plant, and for step, ramp and parabolic inputs, are shown in Table 7.4-1.

The steady-state error values of the PID and PDFF control systems for the double integrator plant and for step, ramp and parabolic inputs are shown summarized in Table 7.4-2.

Table 7.4-1 *Control system and steady-state errors for the single integrator plant. K_ω is the single integrator gain.*

System	C_p	C_v	C_a	$E_{ss}(\text{step})$	$E_{ss}(\text{ramp})$	$E_{ss}(\text{par})$
PID ($K_d=0$)	∞	∞	$K_\omega K_i$	0	0	$\frac{R_o}{K_\omega K_i}$
PDFF ($K_v=K_d=0$)	∞	$\frac{K_i}{K_p}$	0	0	$\frac{R_o K_p}{K_i}$	∞
PDFF ($K_v=K_p$)	∞	∞	$K_\omega K_i$	0	0	$\frac{R_o}{K_\omega K_i}$

Table 7.4-2 *Control system and steady-state errors for the double integrator plant.*

System	C_p	C_v	C_a	$E_{ss}(\text{step})$	$E_{ss}(\text{ramp})$	$E_{ss}(\text{par})$
PID	∞	∞	∞	0	0	0
PDFF ($K_v < K_p$, $K_a=0$)	∞	$\frac{K_i}{K_p - K_v}$	0	0	$\frac{R_o(K_p - K_v)}{K_i}$	∞
PDFF ($K_v=K_p$, $K_a=0$)	∞	∞	$\frac{K_i}{K_d}$	0	0	$\frac{R_o K_d}{K_i}$
PDFF ($K_p=K_v$, $K_a=K_d$)	∞	∞	∞	0	0	0

From the $C(s)/D(s)$ relationships shown in Table 7.3-3 and Table 7.3-4, using the final-value theorem, it can be seen that both the PID and PDF control schemes remove disturbances that may enter the control loop in stepwise fashion. Note, also, that the disturbance-rejection characteristics of the PDF control algorithm are independent of any feedforward control action(s).

It should be noted, again, that the relationships between the analog control gains (upper-case subscripts) and the corresponding digital control gains (lower-case subscripts) are

$$K_p = K_P, \quad K_i = TK_I, \quad \text{and} \quad K_d = \frac{K_D}{T}, \quad (7.4-25)$$

where T represents the control sampling time.

As can be seen from Table 7.4-1 and Table 7.4-2, tracking errors can be reduced and/or eliminated by appropriate control strategies, such as by using feedforward control actions in the PDF control. For the position plant case, for instance, ramp inputs can be tracked with zero steady-state error by setting $K_v = K_p$, or parabolic inputs can be followed with zero steady-state error when additionally $K_a = K_d$. Furthermore, the feedforward control action K_v can be set to achieve specific phase margin characteristics.

Manipulating the reference (setpoint) input using the steady-state error values, determined as functions of controller gains, can also be used effectively to reduce tracking errors. Specifically, the ramp command can be adjusted by adding the steady-state value given in the second row of Table 7.4-2, without the need for a feedforward gain in the control algorithm.

The feedforward gains in the PDF control algorithm can be set for specific steady-state error objectives, as is illustrated by the following example.

Example 7.4-4

Consider the motor drive plant of Example 7.4-2, Section 7.4-4, used in the digital PDF control system of Fig. 7.4-2. The feedback velocity is monitored by a 5000 (pulses/rev) incremental encoder and the current command is applied to the amplifier input via a ± 10 volt, 12-bit DAC.

Assume also that the digital control system is first tuned for jerk-free (no overshoot) acceleration and deceleration using a trapezoidal velocity profile. The acceleration and deceleration times are 0.15 sec each, that is, 150 sampling times, and the top speed is 150 (pulses/T), where T is the sampling time set at $T=0.001$ sec.

It has been determined that this position plant transfer function can be approximated by the double integrator model

$$G_p(s) = \frac{2380}{s^2}$$

That is, the position plant gain is $K_0=2380$ and incorporates the DAC and incremental encoder (PG) gains.

The controller is tuned for an overall bandwidth (corner frequency) of 100 rad/sec, using the ITAE-ZDE tuning strategy. The corresponding digital controller gains are found to be

$$K_p = 9.03, \quad K_i = 0.42 \quad \text{and} \quad K_d = 73.5$$

Then, from Table 7.4-2 and assuming ramp inputs, the steady-state velocity error value is given by

$$E_{ss}(\text{ramp}) = \frac{R_o(K_p - K_v)}{K_i}$$

where R_o , the slope of the ramp, is 1 and $K_v=0$ for this case. Thus,

$$E_{ss}(\text{ramp}) = \frac{K_p}{K_i} = \frac{9.03}{0.42} = 21.5 \text{ (pulses/T)}$$

If $K_v = K_p/3$ and $K_a=0$, for another example, then

$$E_{ss}(\text{ramp}) = \frac{(K_p - K_v)}{K_i} = \frac{9.3 - 3.01}{0.42} = 14.3 \text{ (pulses/T)}$$

Finally, if $K_v = K_p = 9.03$ and $K_a = K_d = 73.5$, then

$$E_{ss}(\text{ramp}) = 0$$

The numerical values for the steady-state errors obtained above were verified experimentally.

7.5 PHASE-LEAD CONTROL ALGORITHMS

Another analog control algorithm used often in practice is shown in Fig. 7.5-1. This controller is referred to as **phase-lead** if the magnitude of its zero is smaller than the magnitude of its pole; otherwise, it is called a **phase-lag** controller.

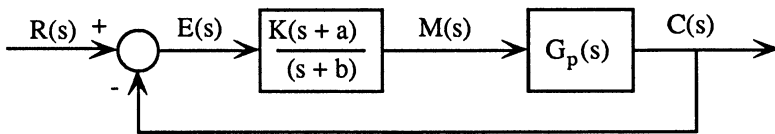


Fig. 7.5-1 Analog control system showing phase-lag/lead controller.

For instance, if the plant is the familiar double integrator with gain K_θ , the closed-loop transfer function of the control system is given by

$$\frac{C(s)}{R(s)} = \frac{KK_\theta s + KK_\theta a}{s^3 + bs^2 + KK_\theta s + KK_\theta a} \quad (7.5-1)$$

This transfer function looks like the zero-velocity-error transfer functions encountered earlier in this chapter. Thus, we can tune the controller using, for instance, the minimum ITAE-ZVE criterion.

As an example, consider the case where $K_\theta=509.65$ and $\omega_c=20$ rad/sec. The tuning scheme suggests the following analog controller parameter values:

$$b = 1.75\omega_c = 35, \quad K = \frac{3.25\omega_c^2}{K_0} = 2.55 \quad \text{and} \quad a = \frac{\omega_c^3}{KK_0} = 6.156$$

The transfer function of the analog controller is

$$G_c(s) = \frac{K(s+a)}{(s+b)} = \frac{2.55(s+6.157)}{(s+35)}$$

which is of the phase-lead type (i.e., $a < b$). To find the corresponding digital controller, use the pole-zero matching method, applied in Chapter 5 to design digital filters from analog filters. Choosing $T=0.005$ sec, results in the digital controller transfer function

$$G_c(z) = \frac{0.4189(z+0.9697)}{(z+0.8395)}$$

where the steady-state gain has been adjusted to match that of the analog controller. For higher-order plants, the previous approach could not be applied unless the higher-order plant can be approximated first by a lower-order model (see Chapter 8). Alternatively, we can apply the well-established phase-lead design technique.

Since the double integrator plant has a phase margin of zero degrees, a compensator must be designed that supplies a desired phase margin for the control system. Assume that it is required for the compensator to provide a phase margin of 38 degrees. Then, according to the phase-lead controller design procedure, first solve for a from the equation

$$\sin 38^\circ = \frac{(a-1)}{(a+1)}$$

This gives the value $a=4.2037$. Next, from the frequency response characteristics of the open-loop transfer function, $H(s)G(s)$, determine the frequency value (in rad/sec) such that

$$-10\log a = -6.236 \text{ db}$$

is satisfied. For this example, the frequency is

$$\omega = 32 \text{ rad/sec}$$

Then, set

$$\omega = 32 = \frac{1}{\sqrt{a} \tau}$$

and solve for τ . This yields $\tau=0.01524$. Hence, the analog phase-lead controller transfer function is

$$G_c(s) = \frac{1+a\tau s}{1+\tau s} = \frac{4.2037(s+15.6093)}{(s+65.6168)}$$

Assuming $T=0.005$ sec, and using the method of pole-zero matching for conversion, the corresponding digital phase-lead controller transfer function is

$$G_c(z) = \frac{0.8937(z+0.9249)}{(z+0.7203)}$$

Both of the previous controllers seem reasonable and may satisfy requirements of certain problem. It should be pointed out, however, that such controllers will not remove disturbances that may enter the control loop. If disturbance rejection with zero steady-state error is an important consideration, then the PDF control scheme can be applied.

Digital controller transfer functions such as those designed above can also be designed directly in the z -domain by experienced designers using root locus or other techniques. It is not easy to visualize, however, the effect that placing poles in the z -domain may have on the characteristics of a system. Therefore, most designers prefer designing it in the s -domain and then converting the design into the z -domain.

7.6 CONTROL ALGORITHMS FOR PLANTS WITH DEAD TIME

Consider the second-order plant described by the transfer function

$$G_p(s) = \frac{Kabe^{-ds}}{(s+a)(s+b)} \quad (7.6-1)$$

where d represents a time delay or lag and a, b are arbitrary positive numbers. From the control system block diagram shown in Fig. 7.6-1, the output transform of the system can be written as

$$C(s) = \frac{Kabe^{-ds}G_c(s)R(s)}{s^2+(a+b)s+ab+Kabe^{-ds}G_c(s)} \quad (7.6-2)$$

Assuming that the output must follow the input, say, d seconds later implies that

$$C(s) = R(s)e^{-ds} = \frac{Kabe^{-ds}G_c(s)R(s)}{s^2+(a+b)s+ab+Kabe^{-ds}G_c(s)} \quad (7.6-3)$$

or

$$1 = \frac{KabG_c(s)}{s^2+(a+b)s+ab+Kabe^{-ds}G_c(s)} \quad (7.6-4)$$

Solving the last equation for $G_c(s)$ gives

$$G_c(s) = \frac{s^2+(a+b)s+ab}{Kab(1-e^{-ds})} \quad (7.6-5)$$

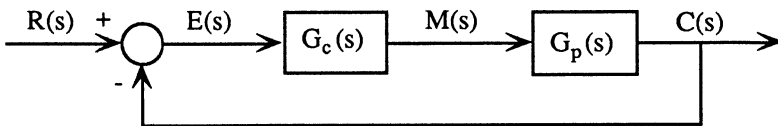


Fig. 7.6-1 Block diagram of a control system.

If the exponential function is approximated by truncating the series expansion after the first two terms, assuming that d is not very large, we can set

$$e^{-ds} \approx 1 - ds \quad (7.6-6)$$

Substituting this result into Eq. (7.6-5), leads to the controller transfer function

$$G_c(s) = \left(\frac{1}{Kab} \right) \left[\frac{s^2 + (a+b)s + ab}{ds} \right]$$

or

$$G_c(s) = \left(\frac{1}{Kabd} \right) \left[(a+b) + \frac{ab}{s} + s \right] \quad (7.6-7)$$

The last expression is similar to the transfer function of the conventional PID algorithm, if the tuning values are

$$K_P = \frac{(a+b)}{Kabd} \quad (7.6-8)$$

$$K_I = \frac{1}{dK} \quad (7.6-9)$$

and

$$K_D = \frac{1}{abdK} \quad (7.6-10)$$

The above relations can be modified to apply to a first-order plant with time delay, also.

Example 7.6-1

A second-order plant has time constants of 0.1 sec and 0.4 sec. It also has a gain of 0.1 and a deadtime (delay) of 2 sampling intervals. Determine a digital PID and a PDFF controller for the system; assume a control sampling time of $T=0.05$ sec.

The transfer function of the plant can be expressed in the form

$$G_p(s) = \frac{Ke^{-ds}}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

where K is the plant gain, τ_1, τ_2 are the time constants and d is the time delay or deadtime. For this case, $d=2T=0.1$ sec. Substituting the given parameter values results in the transfer function

$$G_p(s) = \frac{0.1e^{-0.1s}}{(0.1s+1)(0.4s+1)} = \frac{2.5e^{-0.1s}}{(s+10)(s+2.5)}$$

From Eq. (7.6-8) - (7.6-10), the analog PID controller gains are found to be

$$K_P = \frac{(a+b)}{Kabd} = \frac{10+2.5}{(0.1*10*2.5*0.1)} = 50$$

$$K_I = \frac{1}{dK} = \frac{1}{0.1*0.1} = 100$$

and
$$K_D = \frac{1}{abdK} = \frac{1}{(10 \cdot 2.5 \cdot 0.1 \cdot 0.1)} = 4$$

For a sampling time of $T=0.05$ sec, the digital PID controller gains are

$$K_P = 50, K_I = TK_I = 100 \cdot 0.05 = 5 \text{ and } K_d = \frac{K_D}{T} = \frac{4}{0.05} = 80$$

If a simulation program/language is not readily available, one can analyze the digital control system using conventional techniques. A useful relationship, for instance, is the pulse transfer function of the plant which for this case is determined to be

$$G_h G_p(z) = \frac{0.0025z + 0.00207}{z^3 - 1.489z^2 + 0.535z}$$

The digital PDFF control system was simulated using SIMULAB. Using the controller gains calculated above, a PID controller is implemented first, by setting $K_v=K_p$ and $K_a=K_d$ in the PDFF controller. The simulation results are shown in Fig. 7.6-2. As can be seen from the figure, the system is stable but there is room for improvement. Lowering the integral gain or increasing the proportional gain, or both, may be logical initial actions to take. Either way, some "fine-tuning" of the PID algorithm seems desirable.

Instead of adjusting the PID controller gains, the PDFF algorithm is reduced to its PDF form, that is, the feedforward control actions are removed. The simulation results are shown in Fig. 7.6-3; they seem much better. Both the PID and PDFF control systems will reject disturbances in the same manner.

Control systems with plants like the one used in this example can also be tuned using the inside-out method described earlier. A simulation language makes this type of exercise practical and attractive.

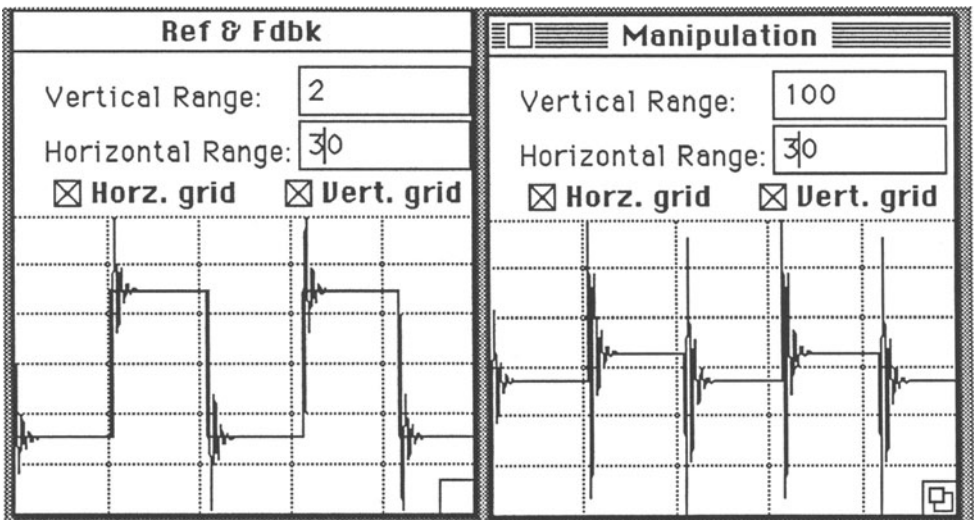


Fig. 7.6-2 Simulation results for the PID control system of Example 7.6-1.

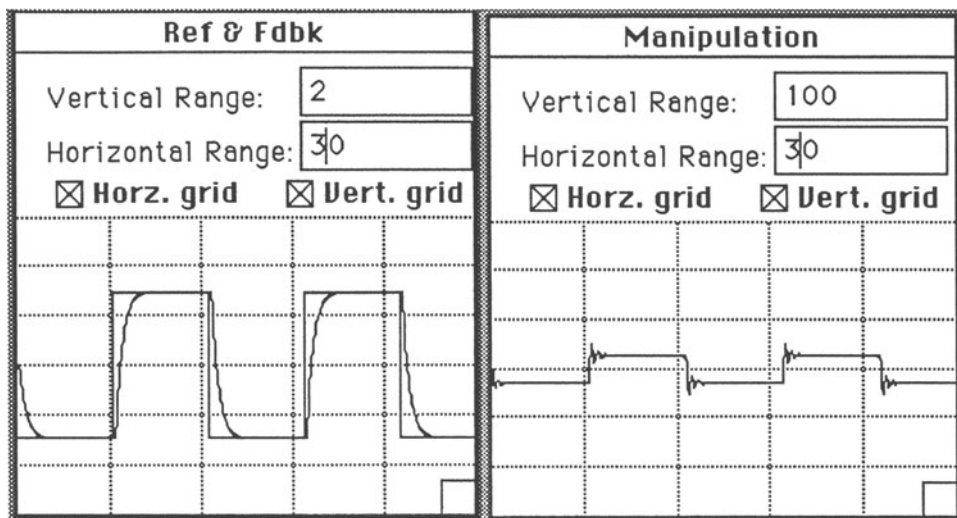


Fig. 7.6-3 Simulation results for the PDFF control system of Example 7.6-1.

7.7 ZIEGLER-NICHOLS TUNING METHOD FOR PID ALGORITHMS

When designing a controller, the main objective is to find a set of tuning values so that the control system meets transient and steady-state specifications for tracking reference inputs and/or rejecting disturbances. Some applications require a relatively slow response without overshoot; in other cases, speed of response is essential and oscillations may not be a problem. We have seen how to approach the tuning problem systematically, but every problem, new or old, brings with it new challenges for the designer, particularly in the digital controls field.

A method for tuning feedback controllers such as the PID was proposed by Ziegler and Nichols in 1942. It is called the **ultimate method** because it requires finding the so-called **ultimate gain** and the **ultimate period** for the control loop.

To find the ultimate gain K_u , use the following approach. With only proportional control in operation, increase the proportional gain until the response oscillates, that is, until the system becomes marginally stable. *The maximum value of the proportional gain for which the system oscillates is the ultimate gain K_u . The period of the sustained oscillation is the ultimate period T_u .*

Ziegler and Nichols used the ultimate gain and ultimate period values to calculate controller settings for various forms of the PID controller given by Eq. (7.1-7), which is repeated here for convenience

$$m(t) = K_C \left[e(t) + \frac{1}{T_i} \int e(t) dt + T_d \frac{de(t)}{dt} \right] \quad (7.7-1)$$

Recall that the type of PID controller used in this book is given by

$$m(t) = K_P e(t) + K_I \int e(t) dt + K_D \frac{de(t)}{dt} \quad (7.7-2)$$

Thus, the proportional gains are the same for both controllers and the integral and derivative gains are related by

$$K_I = \frac{K_C}{T_i} \quad \text{and} \quad K_D = K_C T_d \quad (7.7-3)$$

According to Ziegler and Nichols, the corresponding controller gains can be calculated from the following relationships:

(1) Proportional Controller:

$$K_C = \frac{K_u}{2} = 0.5K_u \quad (7.7-4)$$

(2) Proportional-Integral (PI):

$$K_C = 0.45K_u \quad \text{and} \quad T_i = \frac{T_u}{1.2} \quad (7.7-5)$$

(3) Proportional-Derivative (PD):

$$K_C = 0.6K_u \quad \text{and} \quad T_d = \frac{T_u}{8} \quad (7.7-6)$$

(4) Proportional-Integral-Derivative (PID):

$$K_C = 0.6K_u, \quad T_i = 0.5T_u \quad \text{and} \quad T_d = \frac{T_u}{8} \quad (7.7-7)$$

In many cases, sustained oscillations are not allowed and the ultimate method cannot be used to tune the algorithm. A common criterion for what has been termed "good" control in some industries is to adjust the controller gains so that the step response of the control system has an amplitude ratio of one-quarter (1/4).

Harriott proposed the following modification of the ultimate method to achieve this four to one ratio. The proportional gain K_C is adjusted until the decay ratio of one-quarter is obtained. The period of the step response, call it T_p , is noted and the gains of the PID controller of Eq. (7.7-1) are calculated from the relations

$$T_i = \frac{T_p}{6} \quad \text{and} \quad T_d = \frac{T_p}{1.5} \quad (7.7-8)$$

The Ziegler-Nichols method is illustrated by the following example.

Example 7.7-1

A typical heating plant for a building is characterized by two time constants and a deadtime. For this example, assume time constants of 100 seconds and 20 seconds, and a deadtime of 30 seconds. A plant gain of 1 and a sampling time of 10 seconds are also assumed. It is required to design a digital PI controller using the Ziegler-Nichols method of tuning.

For the given parameter values, the plant transfer function can be written as

$$G_p(s) = \frac{e^{-30s}}{(100s + 1)(20s + 1)} = \frac{(0.01)(0.05)e^{-30s}}{(s + 0.01)(s + 0.05)}$$

For simulation/analysis purposes, the pulse transfer function of this plant is

$$G_h G_p(z) = \frac{(0.02059z + 0.01686)z^{-3}}{z^2 - 1.5114z + 0.54881}$$

The control system was simulated using only proportional control action first. The proportional gain was increased until the response oscillated continuously. The ultimate gain observed is $K_u = 3.8$ and the ultimate period is $T_u = 215$ seconds. From Eq. (7.7-5), the PI controller gains are

$$K_C = 0.45K_u = 0.45 \cdot 3.8 = 1.71 \text{ and } T_i = \frac{T_u}{1.2} = \frac{215}{1.2} = 179.2$$

For the conventional (analog) PI controller, the gains are

$$K_P = 1.71, \text{ and } K_I = \frac{K_C}{T_i} = \frac{1.71}{179.2} = 0.00954$$

Therefore, the corresponding digital PI controller gains are

$$K_P = 1.71, \text{ and } K_I = TK_I = 10 \cdot 0.00954 = 0.0954$$

To evaluate the results, the system was simulated using SIMULAB. Feedback, reference and manipulation signals are shown in Fig. 7.7-1. The results seem reasonable and, if desired, can be used as a starting point for further fine-tuning.

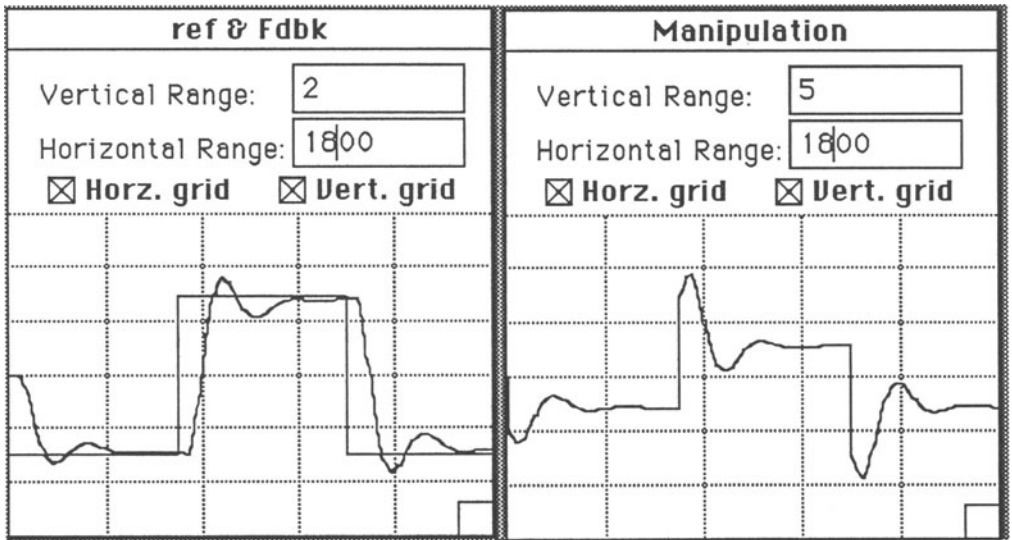


Fig. 7.7-1 Performance characteristics of PI control of heating plant.

7.8 DIGITAL CONTROLLER IMPLEMENTATION CONSIDERATIONS

A programmed digital controller is flexible, reliable and its accuracy can be set to meet problem objectives. However, the software implementation has limitations that affect the performance of the digital control system and must be taken into consideration by the system designer.

It is important when designing a digital controller from an analog prototype that the digital control system resembles the analog control system as closely as possible. In order for this to happen, the control sampling time must be relatively small compared to the dominant time constant of the plant. If this basic rule is not observed, aliasing and other related effects may lead to degraded performance for the system.

Selecting a sampling time that is small, relative to the dominant time constant of the plant, may not be straightforward because an exact model of the plant may not be readily available. This suggests the possibility of selecting the sampling time by some kind of an iterative scheme. That is, make an educated guess for a starting value and then keep changing it, up or down, until an acceptable compromise has been reached.

At the low end, the sampling frequency cannot be smaller than twice the highest frequency of the reference input the digital control system is expected to track. Equivalently, the sampling frequency must be at least twice as large, and perhaps 10-20 times larger than the system bandwidth. Furthermore, for proper disturbance-rejection characteristics, the sampling frequency should be relatively high when compared to the highest frequency of the disturbance signal.

For a digital control system, the sampling frequency, the computer wordlength, the wordlength of the interface devices and coefficient resolution are interdependent. If the encoding of the coefficients is not accurate enough, the resulting controller may exhibit characteristics which differ substantially from those of the original design. For instance, the poles and zeros of the implemented controller may end up in locations which are not close enough to the intended ones. This movement of the poles and zeros may in turn cause the frequency response characteristics to shift appreciably, leading to unsatisfactory performance.

Like in the case of digital filters, coefficient quantization errors depend on wordlength size as well as on controller realization forms. Specifically, if the wordlength is small and/or the controller is of relatively high order, the direct form realization should be avoided. The reason for this is that a coefficient error in a direct form realization will affect all poles and zeros of its transfer function. In a cascade/parallel form realization, on the other hand, the change of one coefficient affects only the poles and zeros of that section.

To implement a digital controller in the laboratory, it is necessary that the computer must be fast enough so that the arithmetic calculations implied by the difference equation form of the controller can be performed within the sampling time interval. If the computational delay is too large relative to the sampling time, it can have adverse effects on the performance of the digital control system. The computational delay can be determined experimentally and, once known, it can be dealt with analytically by the system designer.

To summarize, the sampling frequency, the wordlength of the computer system and the wordlengths of the interface devices must be sufficiently large and properly coordinated. It is important that the system designer selects a sampling rate that properly balances the system cost and system performance characteristics for the application at hand. Internal precision should be adequate so that quantization effects, overflows and limit cycles do not degrade the performance of the digital control system.

PROBLEMS

- 7-1. A plant is described by

$$G_p(s) = \frac{aK}{s(s+a)}$$

Assuming that the above plant is to be controlled using the analog PDFF control scheme shown in Fig. 7.3-2, determine (a) the closed-loop transfer function $C(s)/R(s)$, (b) the output-to-disturbance transfer function $C(s)/D(s)$, and (c) the open-loop transfer function $H(s)G(s)$.

- 7-2. Repeat Problem 7-1 assuming that the plant is to be controlled using the analog PID control scheme shown in Fig. 7.2-1.
- 7-3. Draw the digital control systems corresponding to the analog PDFF and PID control systems of problems 7-1, 7-2, assuming ADC and DAC interfaces and backward-difference integration/differentiation rules. Then, using the parameter values of $K=1000$, $a=10$ and $T=0.01$ sec, design a digital controller, that is, determine appropriate controller gains that result in Bessel-ZAE filter response with a cutoff frequency of $\omega_c=30$ rad/sec. Determine and plot step responses both to reference and disturbance inputs.

- 7-4. For the PID and PDFF control systems of problems 7-3, the plant is a double integrator described by the transfer function

$$G_p(s) = \frac{K_\theta}{s^2}$$

Assuming that $K_\theta=1070$, $T=0.005$ sec and $\omega_c=30$ rad/sec, determine and plot step responses to both reference and disturbance inputs.

- 7-5. It is sometimes assumed that a double integrator plant has enough integral action and the K_i term in the PID/PDFF algorithm is not needed. Design such a (PD) controller for the plant of Problem 7-4. Compare the performance characteristics.
- 7-6. Determine Bode plots for the digital control systems of Problem 7-4. See Eq. (7.4-17).
- 7-7. Determine Bode plots for the digital control systems of Problem 7-5.
- 7-8. Incorporate the double-derivative term K_d in the control algorithms of Problem 7-4. Determine and plot step responses to both reference and disturbance inputs.
- 7-9. Derive the closed-loop transfer function given by Eq. (7.4-17) with K_i added to the PDFF/PID control algorithms. Then, using this expression, repeat problems 7-6 and 7-7.
- 7-10. Design a PDFF-ZVE controller (i.e., PDFF with only the K_v feedforward control action) for the plant of Problem 7-4 to satisfy the following specifications: (1) Phase margin of 50 degrees, (2) attractive disturbance rejection characteristics, and (3) a bandwidth of 30 rad/sec. $K_\theta=1070$, and $T=0.005$ sec.

7-11. Design a phase-lead controller for the plant of Problem 7-4 to satisfy the following specifications: (1) Phase margin of 50 degrees, (2) attractive disturbance rejection characteristics, and (3) a bandwidth of 30 rad/sec. $K_0=1070$, and $T=0.005$ sec.

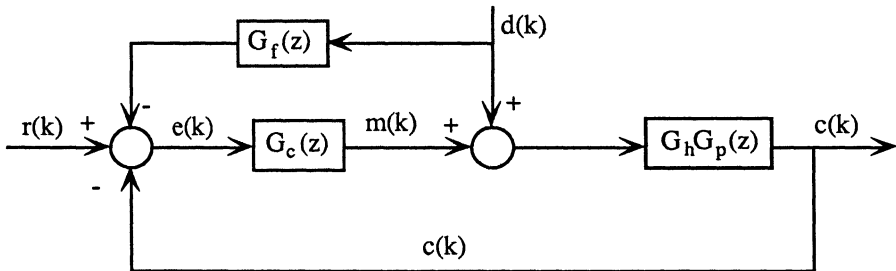
7-12. A heating plant is described by the transfer function

$$G_p(s) = \frac{e^{-40s}}{(100s + 1)(10s + 1)}$$

Assuming a sampling time of 10 seconds, design a digital PI controller using the Ziegler-Nichols method of tuning. Determine and plot system step responses to reference and disturbance inputs.

7-13. Repeat Problem 7-12 for a PID controller.

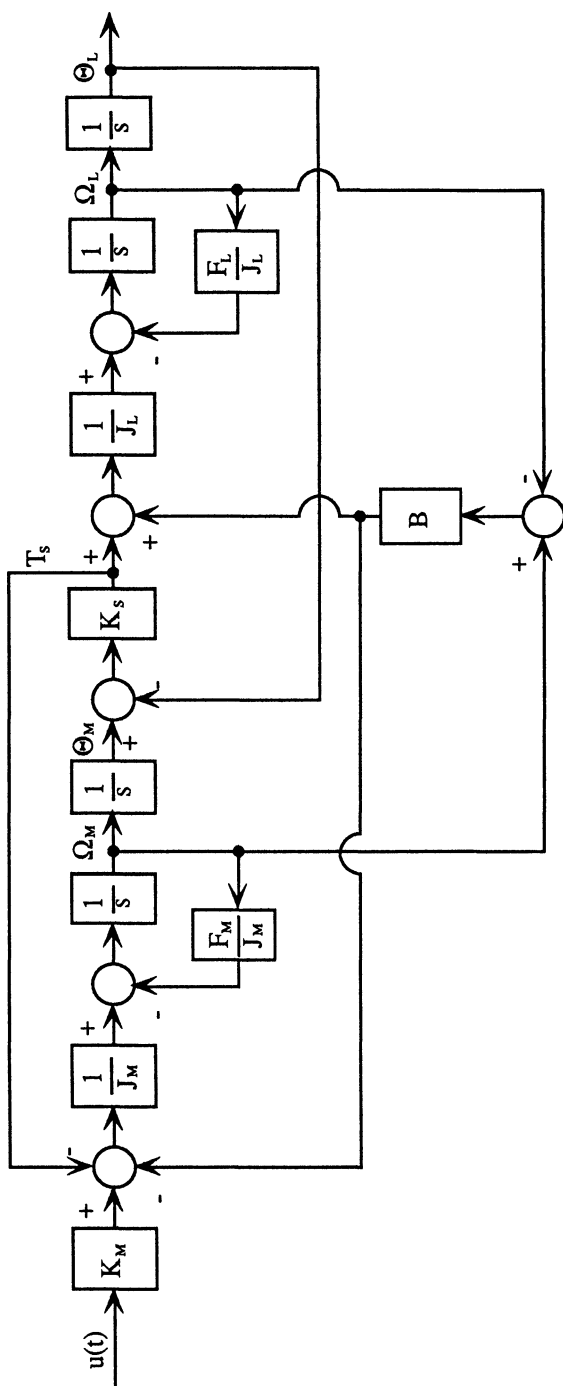
7-14. Assume that the disturbance $d(k)$ entering a control loop is measurable. Then, it is possible to feed the disturbance forward and subtract it from the plant as shown in the block diagram. Determine the controller $G_c(z)$ that will eliminate/minimize the effect of the disturbance. Note, $G_h G_p(z)$ is the pulse transfer function.



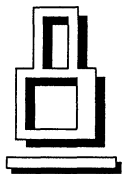
7-15. A block diagram for modeling a motor drive plant with structural flexibility (resonant load) is shown on the next page. The parameters shown in the diagram denote the motor and load inertia J_M and J_L , the motor and load friction F_M and F_L , the spring constant K_s , the load torque T_s , and the damping coefficient B . The parameter K_M is a function of the motor drive current/voltage controller. From the block diagram shown, determine the following:

- (1) Determine a state space representation of the system
- (2) The motor position-to-input transfer function $\Theta_M(s)/U(s)$
- (3) The load position-to-input transfer function $\Theta_L(s)/U(s)$
- (4) If $F_M=F_L=0$, Repeat parts (2) and (3)
- (5) If $F_M=F_L=B=0$, Repeat parts (2) and (3)
- (5) If $F_M=F_L=B=0$, and also $K_s=0$, Repeat parts (2) and (3). This answer can also be obtained by assuming that $J_M \gg J_L$.

7-16. Proposed Case Study: Determine digital control strategies for the plant of Problem 7-15 if (1) a resonant frequency exists outside the bandwidth of the control loop, and (2) if the resonant frequency is near or inside the bandwidth.



Block diagram representation of a motor drive plant with resonant load.



SYSTEM IDENTIFICATION & MODEL REDUCTION

8.1 INTRODUCTION

This chapter has two objectives. The first is to develop a practical scheme for determining a suitable mathematical model for a system from its transient response. Such a scheme is particularly useful when neither a block diagram nor a mathematical description of the system is readily available. In such a case, a system model can be identified from response characteristics obtained experimentally and subsequently used for system analysis and/or controller design.

The second objective is to present a model reduction method that can be applied to obtain simplified models for a system which is described either by a transfer function or a state variable model. As demonstrated in chapter 7, reduced-order models can be used advantageously in the design of algorithms for computer control applications.

8.2 System Identification: A Time Domain Approach

System identification in the time domain consists of finding a transfer function model of a system from its transient response. The requirements usually are that the transfer function must fulfil appropriate realizability conditions, and that it must also meet problem specifications regarding the performance of the system. Furthermore, the transfer function must be in rational polynomial form, with as few poles as possible in order that the desired system be practical and economical.

For a continuous system, the objective of identification may be to find $G(s)$, where $G(s)$ is the Laplace transform of $g(t)$, the impulse response of the system. For a discrete system on the other hand, the objective may be to find $G(z)$, the $\mathcal{Z}$ transform of the discrete impulse sequence $g(k)$, or to find the pulse transfer function $G_h G(z)$.

Time-domain identification methods have not been used widely, mainly because of the mathematically involved nature of the problem. With the advent and availability of digital computers, coupled with the flexibility offered by the application of numerical techniques, the emphasis on system identification is shifting from frequency domain to time domain.

Numerical methods for time-domain approximation are particularly useful where the problem specifications are given in discrete rather than analytic form, or where a rapid and practical solution is desired without the use of extensive tables.

Another important reason that has helped emphasize system identification in the time domain is the increasing use of transient signals, such as short pulses and impulses, in fields where the application of lengthy frequency tests might interfere with the normal operation (equilibrium condition) of the system. This is particularly true in aircraft, spacecraft and many other areas where real-time control systems are frequently used.

The time domain approximation method presented next employs numerical techniques that make it versatile, practical and suitable for implementation on a digital computer.

8.2-1 Identifying the Discrete Transfer Function $G(z)$

The objective of this section is to develop a method for determining a discrete transfer function $G(z)$ which describes the impulse response of a continuous-time system at the sampling instants, according to a least-squares criterion. Therefore, assume that the input function, $x(t)$, is an impulse occurring at $t=0$, and that the output function, $y(t)$, is a bounded time function, of finite duration, possibly corrupted by noise.

A system can be specified by its impulse response, $g(t)=y(t)$, which may be given in analytic form, specified graphically, or described by a sequence of values, $g(k)$. For the present discussion, the accepted form of $g(t)$ is a tabulated series of amplitudes equally-spaced in time, and conversion to this form is required whenever necessary.

If, for instance, the impulse response is given graphically or analytically as $g(t)$, data points must be sampled at equally-spaced time intervals. The interval should be chosen to bring out the salient features of the response, making sure that $g(t)$ is relatively smooth between the sample points.

The given impulse response is approximated by a homogeneous, finite-order difference equation whose constant coefficients are calculated to obtain the least-squared error between the given and desired impulse responses. Solving the difference equation by $\mathcal{Z}$ transforms, results in the determination of the discrete transfer function $G(z)$. The corresponding continuous transfer function $G(s)$ can be derived either by expanding $G(z)$ into partial fractions and using transform tables, or by mapping the poles of $G(z)$ into the s -plane and solving a set of equations. The method is developed in detail in the following paragraphs.

The set of impulse response points, $g(k)$, is first approximated by the homogeneous, n th-order difference equation given by

$$y(k) = \sum_{i=1}^n a_i y(k-i) \quad ; \quad n+1 \leq k \leq N \quad (8.2-1)$$

where N represents the number of points in the sample set $\{g(k)\}$, $k=1, 2, \dots, N$, and

$$y(k-i) = g(k-i) \quad (8.2-2)$$

for all values of k . The criterion for determining the constant coefficients a_i is to minimize the uncorrelated noise in the data. A suitable approach for noisy situations is the principle of **least-squares** approximation. *It states that the best estimate of a number is that number which minimizes the sum of the squares of the deviations of the data from their estimate.*

In terms of Eq. (8.2-1), the least-squares approximation criterion implies finding the coefficients $a_1, \dots, a_n$ which minimize the expression

$$\epsilon = \sum_{k=n+1}^N [y(k) - \sum_{i=1}^n a_i y(k-i)]^2 \quad (8.2-3)$$

Setting the n partial derivatives equal to zero, we obtain

$$\frac{\partial \epsilon}{\partial a_i} = 2 \sum_{k=n+1}^N [y(k) - \sum_{i=1}^n a_i y(k-i)] [-y(k-i)] = 0 \quad (8.2-4)$$

for $i = 1, 2, \dots, n$. Eq. (8.2-4) can be written as

$$\frac{\partial \epsilon}{\partial a_i} = 2 \sum_{k=n+1}^N \sum_{i=1}^n a_i y(k-i) \sum_{i=1}^n y(k-i) - \sum_{k=n+1}^N y(k) \sum_{i=1}^n y(k-i) = 0 \quad (8.2-5)$$

which in matrix form is

$$\begin{bmatrix} \sum y(k-1)y(k-1) & \dots & \sum y(k-1)y(k-n) \\ \vdots & \dots & \vdots \\ \sum y(k-1)y(k-n) & \dots & \sum y(k-n)y(k-n) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \sum y(k)y(k-1) \\ \vdots \\ \sum y(k)y(k-n) \end{bmatrix} \quad (8.2-6)$$

where the summations are from $k=n+1$ to $k=N$.

To develop a criterion for selecting the order n of the desired system model, the following procedure can be followed. For convenience, express Eq. (8.2-6) using the symbolic notation

$$\mathbf{B}\mathbf{A} = \mathbf{C} \quad (8.2-7)$$

where $\mathbf{B}$ denotes the left-hand side square matrix, $\mathbf{A}$ represents the column coefficient matrix and $\mathbf{C}$ the right-hand side column matrix. The coefficients of the difference equation that approximates the sample data points are determined by solving the matrix equation for $\mathbf{A}$, namely

$$\mathbf{A} = \mathbf{B}^{-1}\mathbf{C} \quad (8.2-8)$$

assuming that the inverse exists.

Before Eq. (8.2-6) can be solved, the order n of the difference equation and, therefore, of the desired system must be specified. A minimum of $2n$ samples are required for n independent equations in terms of the coefficients $a_1, \dots, a_n$. If more than $2n$ points are used to describe the impulse response of an n th-order system, then, any set of $(n+1)$ equations in terms of the coefficients is dependent and, thus, the value of the $(n+1)$ st determinant of $\mathbf{B}$ is identically equal to zero. This suggests the following criterion for choosing a proper value of n .

If $g(k)$, for all k , represents the impulse response of a finite-order system, then the order of the system is equal to n whenever the value of the $(n+1)$ st determinant of $\mathbf{B}$ approaches zero, in relation to other determinant values of order less than $n+1$. Selecting n this way avoids the undesirable task of trial and error, which involves solving Eq. (8.2-6) using all $n \leq j$, for any $N=2j$ given points.

To find the discrete transfer function corresponding to $y(k)$, substitute $(k+n)$ for k in Eq. (8.2-1) and use the $\mathcal{Z}$ transform property 12 from Table 3.5-1, that is,

$$\mathcal{Z}[y(k+n)] = z^n[Y(z) - \sum_{k=0}^{n-1} y(k)z^{-k}] \quad (8.2-9)$$

This results in

$$\mathcal{Z}[y(k+n)] = \sum_{i=1}^n a_i \mathcal{Z}[y(k+n-i)] \quad (8.2-10)$$

or

$$z^n[Y(z) - \sum_{k=0}^{n-1} y(k)z^{-k}] = \sum_{i=1}^n a_i z^{n-i} [Y(z) - \sum_{k=0}^{n-i-1} y(k)z^{-k}] \quad (8.2-10)$$

Upon rearranging Eq. (8.2-10), we obtain the desired z -domain function as

$$Y(z) = G(z) = \frac{z^n \sum_{k=0}^{n-1} y(k)z^{-k} - \sum_{i=1}^n a_i z^{n-i} \sum_{k=0}^{n-i-1} y(k)z^{-k}}{z^n - \sum_{i=1}^n a_i z^{n-i}} \quad (8.2-11)$$

For $n=2$, the discrete transfer function becomes

$$G(z) = \frac{z^2 y(0) + z[y(1) - a_1 y(0)]}{z^2 - a_1 z - a_2} \quad (8.2-12)$$

since $y(k)=g(k)=0$, for $k < 0$.

8.2-2 Determining $G(s)$ from $G(z)$

The discrete transfer function $G(z)$ can be determined by applying an impulse function input to the system described by $G(s)$ and then taking the $\mathcal{Z}$ transform of the sampled output, as described above. In this section, we derive the continuous transfer function $G(s)$ corresponding to the discrete transfer function $G(z)$ given by Eq. (8.2-11).

The impulse response $g(t)$ should approximate $g(k)$ acceptably at the sampling instants, without oscillating between them, thus meeting some preset specifications. Eq. (8.2-11) can be expressed in the general form

$$G(z) = \frac{b_0 z^n + b_1 z^{n-1} + \dots + b_{n-1} z + b_n}{z^n + d_1 z^{n-1} + \dots + d_{n-1} z + d_n} \quad (8.2-13)$$

where

$$b_j = y(j) - \sum_{i=0}^j a_i y(j-i) \quad , \text{ and } d_j = -a_j \quad ; \text{ for } j=1,2,\dots,n \quad (8.2-14)$$

The transfer function $G(z)$ can be interpreted as either a cascade or a parallel arrangement of first- and/or second-order systems in the z -domain. One method for obtaining $G(s)$ from the already-known $G(z)$ is to first express $G(z)$ of Eq. (8.2-11) in its parallel form

$$G(z) = \sum_{i=1}^p A_i \frac{z}{(z-r_i)} + \sum_{i=1}^q \frac{B_i z^2 + D_i z}{(z-u_i)^2 + v_i^2} \quad (8.2-15)$$

and then use Table 3.5-1 to find $G(s)$, $g(t)$ and $g(k)$.

An alternative procedure for finding $G(s)$ from $G(z)$ is to map the poles of $G(z)$ from the z -plane into the s -plane by the familiar transformation

$$s = \frac{1}{T} \ln z \quad (8.2-16)$$

where $s = \sigma \pm j\omega$, and $z = u \pm jv$. This mapping specifies the denominator of $G(s)$. To find its numerator, write $G(s)$ in the form

$$G(s) = \sum_{i=1}^p \frac{E_i}{s + \alpha_i} + \sum_{i=1}^q \frac{C_i(s + \sigma_i)}{(s + \sigma_i)^2 + \omega_i^2} + \sum_{i=1}^q \frac{S_i \omega_i}{(s + \sigma_i)^2 + \omega_i^2} \quad (8.2-17)$$

where the coefficients E_i , C_i , and S_i are to be determined. The integers p and q denote the number of real and complex poles, respectively. Therefore, for an n th-order system, $p + 2q = n$. Taking the inverse Laplace transform of Eq. (8.2-17), results in the desired continuous impulse response $g(t)$

$$g(t) = \sum_{i=1}^p E_i e^{-\alpha_i t} + \sum_{i=1}^q C_i e^{-\sigma_i t} \cos \omega_i t + \sum_{i=1}^q S_i e^{-\sigma_i t} \sin \omega_i t \quad (8.2-18)$$

Finally, evaluating Eq. (8.2-18) for n values of $t = kT$, results in the system of algebraic equations described by Eq. (8.2-19), where $t_i = 0, T, \dots, (n-1)T$.

The coefficients $E_1, \dots, E_p$, $C_1, \dots, C_q$ and $S_1, \dots, S_q$, obtained by solving equation (8.2-19), are then substituted into Eq. (8.2-17) and Eq. (8.2-18) in order to specify the continuous functions $G(s)$ and $g(t)$.

An alternative approach for finding the coefficients is to fit the calculated impulse response function of Eq. (8.2-18) to the given data points by the method of least squares. This approach offers the designer the flexibility of emphasizing a region of the response by applying more weight to the respective points.

From Eq. (8.2-16), a complex conjugate pair of poles in the z -plane corresponds to a complex conjugate pair of poles in the s -plane, and a positive real pole in the z -plane corresponds to a real pole in the s -plane. A negative real pole in the z -plane, however, maps into an s -plane pole that must be multiplied by its conjugate to constitute a meaningful expression in the frequency domain. Such poles are normally of high-frequency content and can be disregarded in many cases. In such a case, the steady-state gain of the system transfer function may need adjustment.

$$\begin{bmatrix} e^{-\alpha_1 t_1} \dots e^{-\alpha_p t_1} & e^{-\sigma_1 t_1} \cos \omega_1 t_1 \dots e^{-\sigma_q t_1} \cos \omega_q t_1 & e^{-\sigma_1 t_1} \sin \omega_1 t_1 \dots e^{-\sigma_q t_1} \sin \omega_q t_1 \\ \dots & \dots & \dots \\ e^{-\alpha_1 t_n} \dots e^{-\alpha_p t_n} & e^{-\sigma_1 t_n} \cos \omega_1 t_n \dots e^{-\sigma_q t_n} \cos \omega_q t_n & e^{-\sigma_1 t_n} \sin \omega_1 t_n \dots e^{-\sigma_q t_n} \sin \omega_q t_n \end{bmatrix} \begin{bmatrix} E_1 \\ \vdots \\ E_p \\ C_1 \\ \vdots \\ C_q \\ S_1 \\ \vdots \\ S_q \end{bmatrix} = \begin{bmatrix} g(t_1) \\ \vdots \\ \vdots \\ q(t_n) \end{bmatrix} \quad (8.2-19)$$

The identification procedure presented above is practical and easy to computerize. It requires the designer's judgement when selecting the order of the approximation, the number of sampling points and the sampling time or spacing of the points. If a well-behaved function is given in tabular form, the system identification method using difference equations is straightforward. This is demonstrated by the following example.

Example 8.2-1

The second column of the data shown in Table 8.2-1 represents the impulse response of a system we wish to identify. Determine both discrete and continuous transfer functions, $G(z)$ and $G(s)$, whose impulse response approximates the given impulse response.

As can be seen from Table 8.2-1, the spacing of the points is 0.01 sec, corresponding to a sampling frequency of 100 Hz.

Even though the determinant criterion would have suggested a third-order approximation, a second-order approximation is considered first. This results in the second-order difference equation

$$y(k) = 1.96095y(k-1) - 0.98038y(k-2)$$

To determine the discrete transfer function $G(z)$, one can $\mathcal{Z}$ transform the above difference equation and apply initial conditions obtained from Table 8.2-1. An alternative approach is to use Eq. (8.2-11). This yields the second-order discrete transfer function

$$G(z) = \frac{0.34928z + 0.52937}{z^2 - 1.96095z + 0.98038}$$

The impulse response of this second-order transfer function, obtained by longdividing the numerator by the denominator, is shown in Fig. 8.2-1. As can be seen from the plots, the impulse response of the second-order transfer function is not a good approximation of the given impulse response.

A third-order approximation gives the difference equation

$$y(k) = 2.60133y(k-1) - 2.27821y(k-1) + 0.67033$$

and the transfer function

$$G(z) = \frac{0.34928z^2 + 0.3057z}{z^3 - 2.60133z^2 + 2.27821z - 0.67033}$$

As can be seen from Fig. 8.2-1, the impulse response of this third-order transfer function is identical to the given impulse response. The identified transfer function $G(z)$ can now be converted to a corresponding continuous transfer function $G(s)$ by one of the methods suggested in this section. The result is

$$G(s) = \frac{-0.0079s + 8000.16}{s^3 + 40s^2 + 800.01s + 8000.2} \approx \frac{8000}{s^3 + 40s^2 + 800s + 8000}$$

Table 8.2-1 Data for Example 8.2-1

TIME (sec)	IMPULSE	STEP
0.00	0.00000	0.00000
0.01	0.34928	0.00121
0.02	1.21429	0.00870
0.03	2.36305	0.02643
0.04	3.61479	0.05630
0.05	4.83373	0.09861
0.06	5.92291	0.15254
0.07	6.81830	0.21642
0.08	7.48323	0.28813
0.09	7.90313	0.36527
0.10	8.08081	0.44539
0.11	8.03210	0.52613
0.12	7.78206	0.60536
0.13	7.36172	0.68120
0.14	6.80524	0.75214
0.15	6.14769	0.81697
0.16	5.42317	0.87487
0.17	4.66349	0.92532
0.18	3.89712	0.96811
0.19	3.14861	1.00332
0.20	2.43817	1.03121
0.21	1.78165	1.05226
0.22	1.19060	1.06707
0.23	0.67254	1.07632
0.24	0.23135	1.08077
0.25	-0.13228	1.08120

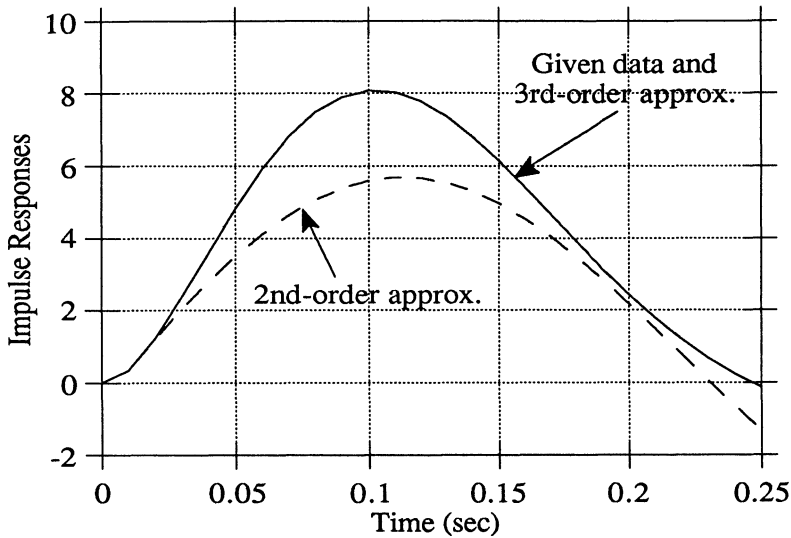


Fig. 8.2-1 Plots of given and approximating impulse responses.

Even though we identified the given system from its impulse response, the system transfer function can also be obtained from its step response, given as column three in Table 8.2-1. The difference is that a third-order approximation is used, instead of a second-order, and the resulting discrete transfer function $G(z)$ is differentiated (multiplied by $(z-1)/z$) in order to obtain the impulse response from the step response.

8.2-3 Identifying the Pulse Transfer Function $G_h G(z)$

If the test signal input is a step function applied by a digital computer via a DAC and the output is sampled by an ADC, long-dividing the output sequence by the input sequence results in the pulse response of the system. If this function is approximated by the difference equation method, the result is the pulse transfer function $G_h G(z)$ rather than the transfer function $G(z)$.

Assume that the input and output signals are given in tabular form, representing amplitudes equally-spaced in time. If $(m+1)$ represents the number of output values, and $(n+1)$ the number of input values, the input and output sequences can be represented by m th-degree and n th-degree polynomials in z^{-1} , respectively. The ratio of the polynomials represents the pulse transfer function of the system, given by

$$G_h G(z) = \frac{\sum_{i=1}^m a_i z^{-i}}{\sum_{i=1}^n b_i z^{-i}} = \sum_{i=1}^n h_i z^{-i} \quad (8.2-20)$$

where $m \leq n$.

The a_i and b_i coefficients correspond to the given values of the output and input sequences, respectively, and the h_i coefficients represent the sequence of numbers that can be obtained from the output and input sequences by long division. The h_i sequence of numbers represents the **pulse response** of the system and can be approximated by the difference equation method.

The pulse transfer function model may be advantageous in applications where it can be obtained during the real-time operation of a system whose input is a sequence of pulses applied by a DAC. This model can also be used to check whether or not the approximation meets time-domain as well as frequency-domain problem specifications.

8.3 A MODEL REDUCTION SCHEME

In designing computer algorithms for controlling plants, a simplified model of the plant is usually helpful. In many cases, a given higher-order model can be simplified by making certain assumptions. In the case of a motor drive plant, for instance, it is often assumed that the friction and/or inductance parameters can be neglected. In other instances, certain system poles may be "thrown away" because, compared to some dominant pole(s), they are relatively large. There are situations, however, where this type of approximation is not practical and a more general and systematic model reduction procedure can prove beneficial.

A model reduction method developed by Rao is presented in this section. Rao's model-reduction method makes use of Routh's approximant simplification schemes and can be applied to obtain reduced-order models for systems represented in either transfer function or state variable form.

8.3-1 Model Reduction from Transfer Function Models

Let the given higher-order system be described by the transfer function

$$G(s) = \frac{b_1 s^{n-1} + b_2 s^{n-2} + \dots + b_n}{s^n + a_1 s^{n-1} + \dots + a_n} \quad (8.3-1)$$

Also, let the desired lower-dimension system, which is to approximate $G(s)$, be represented by the transfer function

$$G_r(s) = \frac{\hat{b}_1 s^{r-1} + \hat{b}_2 s^{r-2} + \dots + \hat{b}_r}{s^r + \hat{a}_1 s^{r-1} + \dots + \hat{a}_r} \quad (8.3-2)$$

The problem is to find the parameters $\hat{b}_i$ and $\hat{a}_i$ ($1 \leq i \leq r$) of the lower-dimension model using the criterion that the lower-dimension system approximates the higher-dimension system satisfactorily in the low-frequency (steady-state) region.

The coefficients $\hat{b}_i$ and $\hat{a}_i$ are determined by using the γ (Gamma) Table (Routh Table with coefficients reversed), and the δ (Delta) Table, shown in Table 8.3-1 and Table 8.3-2, respectively.

Table 8.3-1 Gamma Table (Routh Table with coefficients reversed)

$\gamma_1 \left\{ \begin{array}{l} a_0^0 = a_n \\ a_0^1 = a_{n-1} \end{array} \right.$	$a_2^0 = a_{n-2}$	$a_4^0 = a_{n-4}$	...	$a_n^0 = 1$
$\gamma_2 \left\{ \begin{array}{l} a_0^2 = a_2^0 - \gamma_1 a_2^1 \\ \vdots \end{array} \right.$	$a_2^1 = a_{n-3}$	$a_4^1 = a_{n-5}$	...	
$\gamma_n \left\{ \begin{array}{l} a_0^n = 1 \end{array} \right.$	$a_2^2 = a_4^0 - \gamma_1 a_4^1 \dots$			

Table 8.3-2 Delta Table

$b_0^0 = b_n$	$b_2^0 = b_{n-2}$	$b_4^0 = b_{n-4}$	...	$b_n^0 = 1$
$b_0^1 = b_{n-1}$	$b_2^1 = b_{n-3}$	$b_4^1 = b_{n-5}$	...	
$b_0^2 = b_2^0 - \delta_1 b_2^1$	$b_2^2 = b_4^0 - \delta_1 b_4^1 \dots$			
$\vdots$				
$b_0^n = 1$				

The characteristic equation of the higher-order system is given by

$$s^n + a_1 s^{n-1} + a_2 s^{n-2} + \dots + a_n = 0 \quad (8.3-3)$$

The γ Table is the Routh Table prepared in reverse order, that is, by viewing the characteristic equation from right-to-left. The γ coefficients are evaluated from the elements of the first column of the Table using the relation

$$\gamma_i = \frac{a_0^{i-1}}{a_0^i} ; \text{ for } 1 \leq i \leq n \quad (8.3-4)$$

The δ Table shown in Table 8.3-2 is constructed from the numerator coefficients of the transfer function and from entries of the Routh Table. The δ coefficients are obtained from elements of the first columns of the δ and γ Tables using the relation

$$\delta_i = \frac{b_o^{i-1}}{b_o^i} ; \text{ for } 1 \leq i \leq n \quad (8.3-5)$$

If the reduced-order transfer function is expressed in the form

$$G_r(s) = \frac{B_r(s)}{A_r(s)} \quad (8.3-6)$$

then, the denominator and numerator polynomials are given by

$$A_r(s) = s^2 A_{r-2}(s) + \gamma_r A_{r-1}(s) \quad (8.3-7)$$

and
$$B_r(s) = \delta_r s^{r-1} + s^2 B_{r-2}(s) + \gamma_r B_{r-1}(s) \quad (8.3-8)$$

where
$$B_{-1}(s) = B_o(s) = 0; \quad A_{-1}(s) = \frac{1}{s}, \text{ and } A_o(s) = 1 \quad (8.3-9)$$

Thus, using the γ and δ Tables and the recursive relations of Eq. (8.3-7) and Eq. (8.3-8), we can derive the reduced-order model described by the transfer function of Eq. (8.3-6). The model-reduction method is illustrated in the following example.

Example 8.3-1

Consider a fourth-order system described by its transfer function

$$G(s) = \frac{7.5 \cdot 10^5 s + 7.5 \cdot 10^7}{s^4 + 420.7 s^3 + 57245 s^2 + 2.996 \cdot 10^6 s + 7.5 \cdot 10^7}$$

The problem is to approximate the above system with a second-order model.

The Gamma Table (Routh Table with reversed coefficients) is

	75000000	57245	1
γ_1	2996000	420.7	
γ_2	46710.5	1	
γ_3	356.56		
γ_4	1		

From Eq. (8.3-4), the computed Gammas of the system are

$$\gamma_1 = 25.0334, \quad \gamma_2 = 64.1398, \quad \gamma_3 = 131.003, \quad \text{and} \quad \gamma_4 = 356.56$$

Next, with $r=1$ in Eq. (8.3-7) - Eq. (8.3-9), we obtain

$$A_1(s) = s^2 A_{-1}(s) + \gamma_1 A_o(s) = s + \gamma_1$$

For a second-order approximation ($r=2$),

$$A_2(s) = s^2 + \gamma_2(s + \gamma_1) = s^2 + \gamma_2 s + \gamma_1 \gamma_2$$

Thus, using the previously calculated γ values, the denominator of the second-order transfer function becomes

$$s^2 + 64.1398s + 1605.64$$

To determine the numerator, proceed in a similar fashion. That is, from Eq. (8.3-8), we get

$$B_1(s) = \delta_1 + s^2 B_{-1}(s) + \gamma_1 B_o(s) = \delta_1,$$

and

$$B_2(s) = \delta_2 s + s^2 B_o(s) + \gamma_2 B_1(s) = \delta_2 s + \gamma_2 \delta_1$$

To find δ_1 and δ_2 , use Eq. (8.3-5). That is,

$$\delta_1 = \frac{b_o^0}{a_1^0} = \frac{75000000}{2996000} = 25.0334, \quad \text{and} \quad \delta_2 = \frac{b_o^1}{a_2^0} = \frac{750000}{46710.5} = 16.0564$$

This results in the following second-order approximation of the original system

$$G_2(s) = \frac{\delta_2 + \gamma_2 \delta_1}{s^2 + \gamma_2 s + \gamma_1 \gamma_2} = \frac{16.0564s + 1605.64}{s^2 + 64.1398s + 1605.64}$$

The unit-step responses of the given fourth-order system and the above second-order approximation are shown in Fig. 8.3-1.

8.3-2 Model Reduction from a State Space Model

Assume that an n th-order (higher-order) system is represented by the state and output equations

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{8.3-10}$$

and

$$\mathbf{y} = \mathbf{E}\mathbf{x} \tag{8.3-11}$$

Also, let the simplified r th-order model be given by the state and output equations

$$\mathbf{w}' = \mathbf{F}\mathbf{w} + \mathbf{G}\mathbf{u} \tag{8.3-12}$$

and

$$\mathbf{y}_r = \mathbf{L}\mathbf{w} \tag{8.3-13}$$

Without loss of generality, the matrices $\mathbf{A}$ and $\mathbf{B}$ are given by

$$\mathbf{A} = \begin{bmatrix} 0 & : & \mathbf{I} \\ \dots & : & \dots \\ -a_n & : & -a_{n-1} \dots -a_1 \end{bmatrix}, \text{ and } \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ : \\ 1 \end{bmatrix} \quad (8.3-14)$$

with the characteristic equation of the system given by Eq. (8.3-3).

The γ Table can be prepared as before. Next, consider the transformation

$$\mathbf{v} = \mathbf{P}\mathbf{x} \quad (8.3-15)$$

$$\text{Then,} \quad \mathbf{v}' = \mathbf{R}\mathbf{v} + \mathbf{M}\mathbf{u} \quad (8.3-16)$$

$$\text{and} \quad \mathbf{y} = \mathbf{E}_1\mathbf{v} \quad (8.3-17)$$

$$\text{where} \quad \mathbf{R} = \mathbf{P}\mathbf{A}\mathbf{P}^{-1}, \mathbf{M} = \mathbf{P}\mathbf{B}, \text{ and } \mathbf{E}_1 = \mathbf{E}\mathbf{P}^{-1} \quad (8.3-18)$$

The matrix $\mathbf{P}$, available from the Gamma Table, is given by

$$\mathbf{P} = \begin{bmatrix} a_o^0 & 0 & a_2^0 & 0 & \dots & 1 \\ 0 & a_o^1 & 0 & a_2^1 & \dots & 0 \\ 0 & 0 & a_o^2 & 0 & a_2^2 & \dots & 1 \\ : & : & : & : & : & : & 1 \end{bmatrix} \quad (8.3-19)$$

For n even, the matrices $\mathbf{R}$ and $\mathbf{M}$ have the following structure

$$\mathbf{R} = \begin{bmatrix} 0 & \gamma_2 & 0 & \dots & \gamma_n \\ -\gamma_1 & -\gamma_2 & 0 & \dots & -\gamma_n \\ 0 & 0 & 0 & \dots & \gamma_n \\ -\gamma_1 & -\gamma_2 & -\gamma_3 & \dots & -\gamma_n \\ : & : & : & \dots & : \\ -\gamma_1 & -\gamma_2 & -\gamma_3 & \dots & -\gamma_n \end{bmatrix}, \mathbf{M} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ : \\ 1 \end{bmatrix} \quad (8.3-20)$$

$$\mathbf{E}_1 = [\delta_1 \quad \delta_2 \quad \delta_3 \quad \dots \quad \delta_n] \quad (8.3-21)$$

and for n odd, the relevant matrices are

$$\mathbf{R} = \begin{bmatrix} -\gamma_1 & 0 & -\gamma_3 & \dots & -\gamma_n \\ 0 & 0 & \gamma_3 & \dots & \gamma_n \\ -\gamma_1 & -\gamma_2 & -\gamma_3 & \dots & -\gamma_n \\ 0 & 0 & 0 & \dots & \gamma_n \\ : & : & : & \dots & : \\ -\gamma_1 & -\gamma_2 & -\gamma_3 & \dots & -\gamma_n \end{bmatrix}, \mathbf{M} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ : \\ 1 \end{bmatrix} \quad (8.3-22)$$

The matrix $\mathbf{E}_1$ is the same as in Eq. (8.3-21).

The system representations given by Eq. (8.3-16) and Eq. (8.3-17) are called the **Gamma-Delta representations**. The lower-dimension model is obtained by

suppressing the less dominant (n-r) states in the γ - δ canonical form. Thus, the lower-dimension model state vector is

$$\mathbf{w} = \mathbf{C}_1 \mathbf{v} \quad (8.3-23)$$

where

$$\mathbf{C}_1 = [\mathbf{I}_r : \mathbf{0}]_{r \times n} \quad (8.3-24)$$

and $\mathbf{I}_r$ is the identity matrix of order r.

The lower-dimension model matrices $\mathbf{F}$, $\mathbf{G}$ and $\mathbf{L}$ are submatrices of $\mathbf{R}$, $\mathbf{M}$ and $\mathbf{E}$, and are given by

$$\mathbf{F} = \mathbf{C}_1 \mathbf{R} \mathbf{C}_1^T ; \quad \mathbf{G} = \mathbf{C}_1 \mathbf{M} ; \quad \mathbf{L} = \mathbf{E}_1 \mathbf{C}_1^T \quad (8.3-25)$$

It can be seen that the lower-order matrices can be obtained by truncating the higher-order system matrices. This is valid if the given system is of odd order and the desired model is also odd, or if the system and model orders are even. However, in those situations when an odd-order model is required for an even-order system (or even-order model for an odd-order system), it is necessary to generate the lower-order model matrices from the available values of the computed Gammas and Deltas.

Example 8.3-2

As an example of the state space approach, consider again the fourth-order system of Example 8.3-1, and assume that it is required to approximate it by lower-order models using the state space representation.

For this system, the transformation matrix $\mathbf{P}$ is

$$\mathbf{P} = \begin{bmatrix} 2999000 & 0 & 420.7 & 0 \\ 0 & 46710.5 & 0 & 1 \\ 0 & 0 & 356.56 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and the system matrix $\mathbf{A}$ can be written in the form

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -25.0334 & -64.1398 & -131.003 & -356.56 \end{bmatrix}$$

From Eq. (8.3-20), the Gamma-Delta matrix $\mathbf{R}$ becomes

$$\mathbf{R} = \begin{bmatrix} 0 & 64.1398 & 0 & 356.56 \\ -25.0334 & -64.1398 & 0 & -356.56 \\ 0 & 0 & 0 & 356.56 \\ -25.0334 & -64.1398 & -131.003 & -356.56 \end{bmatrix}$$

and the $\mathbf{M}$ and $\mathbf{E}$ matrices are

$$\mathbf{M} = [0 \quad 1 \quad 0 \quad 1]^T, \text{ and } \mathbf{E}_1 = [25.0334 \quad 16.0564 \quad -29.5356 \quad -16.0564]$$

The desired second-order model is obtained in state space form by properly truncating the above matrices. This results in the state and output equations

$$\begin{bmatrix} w_1' \\ w_2' \end{bmatrix} = \begin{bmatrix} 0 & 64.1398 \\ -25.0334 & -64.1398 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

and

$$y_2 = [25.0334 \quad 16.0564] \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

The corresponding second-order transfer function is then obtained from the state and output equations as

$$G_2(s) = \frac{Y_2(s)}{U(s)} = \frac{16.0564s+1605.64}{s^2+64.1398s+1605.64}$$

which, as expected, is the same as that obtained in Example 8.3-1.

For a third-order approximation (i.e., even to odd), use Eq. (8.3-22) to obtain the relevant matrices

$$\mathbf{R} = \begin{bmatrix} -25.0334 & 0 & -131.003 \\ 0 & 0 & 131.003 \\ -25.0334 & -64.1398 & -131.003 \end{bmatrix}, \quad \mathbf{M} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

and

$$\mathbf{E}_1 = [25.0334 \quad -64.1398 \quad -131.003]$$

Equivalently, the third-order approximation can also be expressed by the transfer function

$$G_3(s) = \frac{-4.5002s^2+2103.44s+210343}{s^3+156.036s^2+8402.51s+210343}$$

The first-order approximation is given in transfer function form by

$$G_1(s) = \frac{25.0334}{s+25.0334}$$

Using the approach of section 8.3-1, one can show that the general expressions for the third-order, and first-order transfer functions are

$$G_3(s) = \frac{(\delta_1+\delta_3)s^2+\gamma_3\delta_2s+\gamma_2\gamma_3\delta_1}{s^3+(\gamma_1+\gamma_3)s^2+\gamma_2\gamma_3s+\gamma_1\gamma_2\gamma_3}, \quad \text{and} \quad G_1(s) = \frac{\delta_1}{s+\gamma_1}$$

The unit-step response of the given fourth-order system, along with the first-order and second-order approximations are shown in Fig. 8.3-1 for comparison. The third-order approximation is not plotted because it is very similar to the original system.

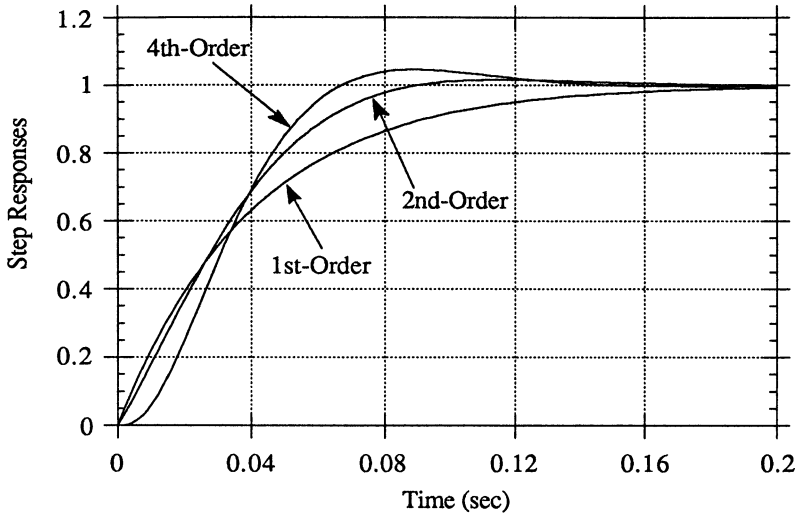


Fig. 8.3-1 Unit-step responses for first-order, second-order, and fourth-order (original) system models used in Examples 8.3-1 and 8.3-2.

8.4 CASE STUDY: REDUCED-ORDER MODELS FOR A MOTOR DRIVE

Reduced-order models, whether obtained analytically or experimentally, are practical and can be helpful when designing control algorithms. This is particularly useful in the case of motor drive applications, where simplified plant models are not readily available.

For the DC motor drive system analyzed in Section 4.7 (see Case Study, Part I), the velocity-to-current (analog) transfer function $\Omega(s)/I_t(s)$ is given by

$$G_{\omega}(s) = \frac{\frac{1}{J} \left(\frac{K_1 K_t}{L} s + \frac{K_2 K_t}{L} \right)}{s^3 + \left(\frac{F}{J} + \frac{R + K_1 K_c}{L} \right) s^2 + \frac{FR + FK_1 K_c + JK_2 K_c + K_b K_t}{JL} s + \frac{FK_2 K_c}{JL}} \quad (8.4-1)$$

Using the simplification method presented in Section 8.3, the reduced first-order and second-order models are

$$G_{\omega-1}(s) = \frac{\delta_1}{s + \gamma_1} = \frac{K_2 K_t}{(FR + K_t K_b + JK_2 K_c + FK_1 K_c)s + FK_2 K_c} \quad (8.4-2)$$

and

$$G_{\omega-2}(s) = \frac{\delta_2 s + \gamma_2 \delta_1}{s^2 + \gamma_2 s + \gamma_1 \gamma_2} \quad (8.4-3)$$

where the γ and δ values are calculated from the relations

$$\gamma_1 = \frac{FK_2K_c}{FR+K_tK_b+JK_2K_c+FK_1K_c} \quad (8.4-4)$$

$$\gamma_2 = \frac{(FR+K_tK_b+JK_2K_c+FK_1K_c)^2}{(JR+FL+JK_1K_c)(FR+K_tK_b+JK_2K_c+FK_1K_c)-FK_2K_cJL} \quad (8.4-5)$$

$$\delta_1 = \frac{K_2K_t}{FR+K_tK_b+JK_2K_c+FK_1K_c} \quad (8.4-6)$$

and

$$\delta_2 = \frac{K_2K_t}{FR+K_tK_b+JK_2K_c+FK_1K_c} \quad (8.4-7)$$

For the frictionless ($F=0$) case, the reduced-order analog velocity plant models become

$$G_{\omega-1}(s) = \frac{K_2K_t}{(K_tK_b+JK_2K_c)} \frac{1}{s} \quad (8.4-8)$$

and

$$G_{\omega-2}(s) = \frac{K_1K_t}{(JR+JK_1K_c)} \frac{s + \frac{K_2}{K_1}}{s(s + \frac{K_tK_b+JK_2K_c}{JR+JK_1K_c})} \quad (8.4-9)$$

If analog position, rather than velocity, plant models are desired, the velocity plant transfer functions are simply divided by s .

To obtain digital, rather than analog, position plant models, substitute the parameter values given in Part I of the Case Study and multiply by the interface gains K_{da} and K_{dt} . This results in the following (digital) position plant transfer functions:

$$G_{\theta-1}(s) = \frac{K_{\theta}}{s^2} = \frac{1069.63}{s^2} \quad (8.4-10)$$

$$G_{\theta-2}(s) = \frac{1405.264(s+166.667)}{s^2(s+218.964)} = \frac{1405.264s+234210}{s^2(s+218.964)} \quad (8.4-11)$$

From Eq. (4.7-16), the "unreduced" position plant model is

$$G_{\theta}(s) = \frac{172333.2(s+166.667)}{s^2(s^2+269.634s+26852.46)} = \frac{172333.2s+28722000}{s^2(s^2+269.634s+26852.46)} \quad (8.4-12)$$

If a simplified position plant model is not readily available, it is possible to obtain the position plant gain K_{θ} experimentally. For example, consider the simplified position plant transfer function

$$G_{\theta}(s) = \frac{\Theta(s)}{M(s)} = \frac{K_{\theta}}{s^2} \quad (8.4-13)$$

where $M(s)$ represents the input/manipulation and K_{θ} the position plant gain. Then, from Eq. (8.4-8), the all-analog plant position gain, call it K_{m} , is given by

$$K_m = \frac{K_2 K_t}{(K_t K_b + J K_2 K_c)} \quad (8.4-14)$$

When the DAC and PG interfaces are also considered, the position plant gain is written in the form

$$K_\theta = (K_{da})(K_m)(K_{dt}) = \frac{K_{da} K_{dt} K_2 K_t}{(K_t K_b + J K_2 K_c)} \quad (8.4-15)$$

From Eq. (4.7-20), the relationship between the velocity and position plant gains is

$$K_\omega = T K_\theta \quad (8.4-16)$$

where K_ω represents the slope of the step response of the (digital) velocity plant. Thus, K_ω can be calculated from the experimental response data (column 2 in Table 8.4-1) by evaluating the slope of the motor drive (PG) sample values to a unit-step input. Using the sample values corresponding to the times of 0.2 sec and 0.1 sec, for instance, we obtain the velocity plant gain as

$$K_\omega = \frac{(215-105)}{(0.2-0.1)204} = 5.392 \quad (8.4-17)$$

From Eq. (8.4-15), the corresponding position plant gain becomes

$$K_\theta = \frac{K_\omega}{T} = \frac{5.392}{0.005} = 1078 \quad (8.4-18)$$

This value compares favorably with the value of 1069.6 calculated from the motor drive parameter values given in Section 4.7.

The position plant transfer functions given by Eq. (8.4-10)-(8.4-12) can be used to obtain corresponding (digital) velocity plant transfer functions. To accomplish this, simply multiply the position plant transfer functions by s and by T , the sampling time. The multiplication by T is required to convert the plant gains from K_θ to K_ω .

The responses of the velocity plant transfer functions to a step input of size 204 (pulses/ T) are shown as the rightmost three columns of Table 8.4-1. The step response of the actual DC motor drive system was obtained experimentally and is shown as the second column of the Table. Specifically, a computer applied a step of 204 pulses (equivalent to 1 volt) at the current command input of the plant via a DAC interface and the output of the digital tach (PG) was monitored every 5 msec. If desired, the tach output (velocity) can be integrated to obtain position values in pulses.

The velocity plant gain K_ω can also be estimated from experimental data by yet another approach. If the size of the step input applied by the computer is denoted by M_o , then, since $M(s)=M_o/s$, the output transform of the velocity plant can be written as

$$\Omega(s) = \left(\frac{K_\omega}{s}\right)\left(\frac{M_o}{s}\right) = \frac{K_\omega M_o}{s^2} \quad (8.4-19)$$

Taking the inverse Laplace transform of Eq. (8.4-19), gives

$$\omega(t) = K_\omega M_o t \quad (8.4-20)$$

Table 8.4-1. Velocity step responses for motor drive plant (step=204 pulses, $T=0.005$ sec)

Time(sec)	Input	Drive (PG)	1st-order	2nd-order	3rd-order
0.000	204	0	0	0	0
0.005	204	0	5	6	2
0.010	204	3	11	12	6
0.015	204	8	16	18	11
0.020	204	15	22	23	17
0.025	204	21	27	29	23
0.030	204	26	32	34	28
0.035	204	33	38	40	34
0.040	204	37	43	45	39
0.045	204	43	49	51	45
0.050	204	51	54	56	50
0.055	204	54	59	62	56
0.060	204	61	65	67	61
0.065	204	67	70	72	67
0.070	204	71	76	78	72
0.075	204	79	81	83	77
0.080	204	84	86	89	83
0.085	204	89	92	94	88
0.090	204	94	97	100	94
0.095	204	101	103	105	99
0.100	204	105	108	111	105
0.105	204	112	113	116	110
0.110	204	117	119	122	116
0.115	204	123	124	127	121
0.120	204	128	130	132	127
0.125	204	134	135	138	132
0.130	204	138	140	143	137
0.135	204	145	146	149	143
0.140	204	151	151	154	148
0.145	204	155	157	160	154
0.150	204	161	162	165	159
0.155	204	166	167	171	165
0.160	204	172	173	176	170
0.165	204	177	178	182	176
0.170	204	182	184	187	181
0.175	204	188	189	192	187
0.180	204	194	194	198	192
0.185	204	198	200	203	197
0.190	204	204	205	209	203
0.195	204	209	211	214	208
0.200	204	215	216	220	214

Eq. (8.4-20) can be evaluated at some steady-state time t_o to give the plant gain as

$$K_{\omega} = \frac{\omega(t_o)}{M_o t_o} \quad (8.4-21)$$

Using the experimental data of the second column of Table 8.4-1, for instance, we can determine K_{ω} by evaluating Eq. (8.4-21) for, say, $t_o=0.2$ sec. Since the input is $M_o=204$ pulses, the velocity plant gain is

$$K_{\omega} = \frac{215}{204 \cdot 0.2} = 5.27 \quad (8.4-22)$$

This corresponds to a position plant gain of $K_{\theta}=1054$, another good estimate.

One should be careful when using this latter approach for finding plant gains, because a single sample point may lead to inaccurate values, particularly in noisy data situations. A more elaborate but perhaps "safer" method for finding the slope of the velocity response (PG output) is to fit a set of data points selected from the steady-state region with a straight line using, for instance, the method of least squares (linear regression). Then, the slope of the line becomes the value of the velocity plant gain K_{ω} . To illustrate this method, the sample points in the region from 0.1 sec to 0.2 sec are shown plotted in Fig. 8.4-1. The equation of the line (first-degree polynomial fit) determined by the method of least squares is

$$\omega(t) = 1087t - 2.385 \quad (8.4-22)$$

To verify the validity of this equation, evaluated it at, say, $t=0.1$. This gives $\omega(0.1)=106$ pulses/T, a close approximation to the actual value of 105 pulses/T shown in Table 8.4-1.

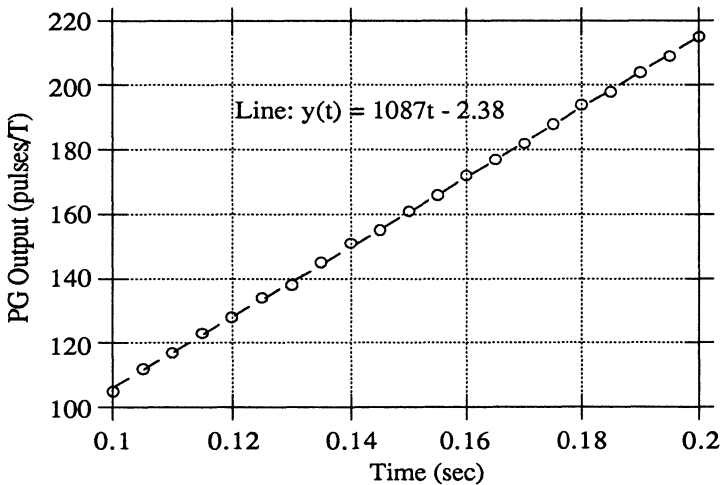


Fig. 8.4-1 Plot of experimental motor drive output (PG) values vs. time, and linear approximation of data. Note slope of line implying $K_{\omega}=1087$.

PROBLEMS

- 8-1.** Write a computer program to implement the system identification scheme proposed in this section. At a minimum, implement the system of equations given by Eq. (8.6-2).
- 8-2.** Identify the system of Example 8.2-1 from its step, rather than impulse, response data.
- 8-3.** The unit-impulse response of a system is represented by the sampled points shown in the following table. Identify (a) a second-order, and (b) a third-order approximation of the system. Find transfer functions, continuous and discrete.

<u>Time (sec)</u>	<u>Impulse Response</u>
0.00	0.00000
0.05	0.23932
0.10	0.73229
0.15	1.25742
0.20	1.69976
0.25	2.00946
0.30	2.17541
0.35	2.20846
0.40	2.13068
0.45	1.96852
0.50	1.74867
0.55	1.49552
0.60	1.22986
0.65	0.96833
0.70	0.72339
0.75	0.50357
0.80	0.31397
0.85	0.15683
0.90	0.03210
0.95	-0.06196
1.00	-0.12827

- 8-4.** Determine reduced-order models for the motor position plant of Problem 7-16. Find both transfer function and state space models. See Section 8.4 for ideas.
- 8-5.** Determine reduced-order models for the load position plant of Problem 7-16. Find both transfer function and state space models.
- 8-6.** Determine a second-order model that approximates the transfer function

$$G(s) = \frac{250}{(s+10)(s^2+6s+25)}$$

APPENDIX A: REFERENCES

- Ahmed, N., Natarajan, T.,** *Discrete-Time Signals and Systems*.
Reston, Va: Prentice-Hall, Inc., 1983.
- Astrom, K.J., Wittenmark, B.,** *Computer Controlled Systems Theory and Design*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1984.
- Bibbero, R.J.,** *Microprocessors in Instruments and Control*.
New York: John Wiley & Sons, Inc., 1977.
- Bishop, A.B.,** *Introduction to Linear Controls: Theory and Application*.
New York: Academic Press Inc., 1975.
- Bollinger, J.G., Duffie, N.A.,** *Computer Control of Machine and Processes*.
Reading, Ma: Addison-Wesley Publishing Company, 1988.
- Cadzow, J.A.,** *Discrete-Time Systems : An Introduction With Intredisciplinary Applications*. Englewood Cliffs, N.J. : Prentice-Hall, Inc., 1973.
- Cadzow, J.A., Martens, H.R.,** *Discrete Time and Computer Control Systems*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1970.
- Cassell, D.A.,** *Microcomputers & Modern Control Engineering*.
Reston, Va.: Reston Publishing Inc., 1983.
- Chirlian, P.M.,** *Signals, Systems and the Computer*.
New York: Intext Press, Inc., 1973.
- Clarke, D.W.,** *"PID algorithms and their computer implementation"*, University of Oxford, O.U.E.L. Report No. 1482/83.
- D'Souza, A.F.,** *Design of Control Systems*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1988.
- Derusso, P.M., Roy, R.J., Close, C.M.,** *State Variables for Engineers*.
New York: John Wiley & Sons, Inc., 1967.
- Dorf, R.C.,** *Modern Control Systems*.
Reading, Ma: Addison-Wesley Publishing Company, 1989.
- Franklin, G.F., Powell, J.D., Workman, M.L.,** *Digital Control of Dynamic Systems*. Reading, Ma: Addison-Wesley Publishing Co., 1990.
- Gabel, R.A., Roberts, R.A.,** *Signals and Linear Systems*.
New York: John Wiley & Sons, Inc., 1973.
- Graham, D., Lathrop, R.C.,** *"The synthesis of optimum transient response: Criteria and Standard forms,"* Trans. AIEE, vol. 72, pp. 273-288, Nov. 1953.
- Gupta, S.C., Hasdorff, L.,** *Fundamentals of Automatic Control*.
New York: John Wiley & Sons, Inc., 1970.
- Hostetter, G.H.,** *Digital Control System Design*.
Holt, Rinehart and Winston Inc., 1988.
- Houpis, C.H., Lamont, G.B.,** *Digital Control Systems Theory, Hardware, Software*.
New York: McGraw-Hill Book Company, 1985.
- Isermann, R.,** *Digital Control Systems*.
Heidelberg: Springer-Verlag Heidelberg, 1981.
- Jacquot, R.G.,** *Modern Digital Control Systems*.
New York: Marcel Dekker, Inc., 1981.
- Johnson, C.D.,** *Process Control Instrumentation Technology*.
New York: John Wiley & Sons, Inc., 1982.
- Jury, E.I.,** *Theory and Application of the Z - Transform Method*.
New York: John Wiley & Sons, Inc., 1964.

- Katz, P.**, *Digital Control Using Microprocessors*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1981.
- Kuo, B.C.**, *Automatic Control Systems*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1982.
- Kuo, B.C.**, *Digital Control Systems*.
New York: Holt, Rinehart and Winston, Inc., 1991.
- Lago, G.V., Benningfield, L.M.**, *Circuit and System Theory*.
New York: John Wiley & Sons, Inc., 1979.
- Lewis, L.J., Reynolds, D.K., Bergseth, F.R., Alexandro, F.J. Jr.**, *Linear Systems Analysis*. New York: McGraw-Hill Book Company, 1969.
- Liu, C.L., Liu, Jane W.S.**, *Linear Systems Analysis*.
New York: McGraw-Hill Book Company, 1975.
- McGillem, C.D, Cooper, G.R.**, *Continuous and Discrete Signal and System Analysis*. New York : CBS College Publishing, 1984.
- Middleton, R.H., Goodwin, G.C.**, *Digital Control and Estimation :A Unified Approach*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1990.
- Monroe, A.J.**, *Digital Processes For Sampled Data Systems*.
New York: John Wiley & Sons, Inc., 1962.
- Murrill, P.W.**, *Fundamentals of Process Control Theory*.
Research Triangle Park, NC: Instrument Society of America, 1981.
- Neff, H.P., Jr.**, *Continuous and Discrete Linear Systems*.
New York: Harper & Row Publishers, 1984.
- Ogata, K.**, *Discrete -Time Control Systems*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1987.
- Phelan, R.M.**, *Automatic Control Systems*.
London: Cornell University Press Ltd., 1977
- Phillips, C.L., Nagle, H.T.**, *Digital Control System Analysis and Design*.
Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1984.
- Roa, A.S., Lamba, S.S, Roa, S.V.**, "Routh approximant time domain reduced order models for SISO systems", Proc. IEE, London, England, pp 1059-1063, Oct, 1978.
- Sinha, N.K.**, *Control Systems*. New York: Holt, Rinehart and Winston, 1986.
- Smith, C.L.**, *Digital Computer Process Control*.
Scranton, Pa: Intext Educational Publishers, 1972.
- Smith, J.M.**, *Mathematical Modeling & Digital Simulation for Engineers & Scientists*. New York: John Wiley & Sons, 1977.
- Smith, O.J.M.**, *Feedback Control Systems*.
New York: McGraw-Hill Book Company, 1958.
- Stanley, W.D., Dougherty, G.R., Dougherty, R.**, *Digital Signal Processing*.
Reston, Va: Prentice-Hall, Inc., 1984.
- Steiglitz, K.**, *An Introduction to Discrete Systems*.
New York: John Wiley & Sons, Inc., 1974.
- Tretter, S.A.**, *Introduction to Discrete-Time Signal Processing*.
New York: John Wiley & Sons, Inc., 1976.
- Truxal, J.G.**, *Automatic Feedback Control System Synthesis*.
McGraw-Hill Book Company, 1955.
- Tzafestas, S.G.**, *Applied Digital Control :Volume 7*.
Amsterdam: Elsevier Science Publishers B.V., 1985.
- Tzafestas, S.G.**, *Microprocessors on Signal Processing, Measurement and Control* .
Dordrecht: D. Reidel publishing Company, 1983.
- VanLandingham, H.F.**, *Introduction to Digital Control Systems*.
New York: Macmillan Publishing Company, 1985.

Wolovich, W.A., *Robotics : Basic Analysis and Design*.

New York: Holt, Rinehart and Winston, 1987.

Yasuhiko, D., *Servo Motor and Motion Control using Digital Signal Processors*.

Reston, Va: Prentice-Hall, Inc., 1990.

Ziegler, J.G., Nichols, N.B., "*Optimal settings for Automatic Controllers*", ASME Transactions, vol. 64, pp. 759-765, 1942.

Ziemer, R.E., Tranter, W.H., Fannin, D.R., *Signals and Systems Continuous and Discrete*. New York: Macmillan Publishing Co., Inc., 1983.

APPENDIX B: REAL-TIME PROGRAMS

```

/*****
*   filter_float.c
*
*       Floating point implementation of a 2nd order real-time iir filter
*
*****/

#define FALSE          0
#define TRUE           !FALSE
#define DAC_LIM_POS    2047      /* Positive limit of a 12 bit DAC */
#define DAC_LIM_NEG    -2048     /* Negative limit of a 12 bit DAC */

/* Global variables */

float   a0, a1, a2, b1, b2;      /* Filter constants */
float   xk_m1, xk_m2;            /* Filter delayed terms */
float   yk_m1, yk_m2;
short   clock_rate;              /* Interrupt sampling rate in msec. */

extern char   toupper();          /* Force a char to upper case */
extern int    getchar();          /* Get a char from the keyboard */
extern short  adc();              /* Analog to digital conversion */
extern void   dac();              /* Digital to analog conversion */
extern void   disable_interrupt(); /* Disable real time clock interrupts */
extern void   enable_interrupt(); /* Enable real time clock interrupts */
extern void   foreground();       /* Interrupt service routine */
extern void   set_clock();        /* Set real time interrupt clock */
extern void   zero_dacs();        /* Zero digital to analog converters */

main()
{
    zero_dacs();

    while ( TRUE )                /* Background loop */
    {
        puts( "\nCOMMAND ? " );
    }
}

```



```

switch ( toupper( getchar() ) )
{
    case 'S':                                /* Stop the filter                */
        disable_interrupt();
        zero_dacs();
        break;

    case 'G':                                /* Start the filter                */
        xk_m2 = 0.0;                         /* Zero all delayed terms          */
        xk_m1 = 0.0;
        yk_m2 = 0.0;
        yk_m1 = 0.0;
        enable_interrupt();
        break;

    case 'F':                                /* Enter filter constants          */
        puts( "\nEnter all filter parameters" );
        printf( "\nInterrupt clock rate %d (msec.)", clock_rate );
        puts( "\nEnter new interrupt clock rate" );
        scanf( "%d", &clock_rate );
        set_clock( clock_rate );

        printf( "\nA0 = %f", a0 );
        puts( "\nNew value for A0 ?" );
        scanf( "%f", &a0 );

        printf( "\nA1 = %f", a1 );
        puts( "\nNew value for A1 ?" );
        scanf( "%f", &a1 );

        printf( "\nA2 = %f", a2 );
        puts( "\nNew value for A2 ?" );
        scanf( "%f", &a2 );

        printf( "\nB1 = %f", b1 );
        puts( "\nNew value for B1 ?" );
        scanf( "%f", &b1 );

        printf( "\nB2 = %f", b2 );
        puts( "\nNew value for B2 ?" );
        scanf( "%f", &b2 );
        break;
}
}
}

```

```

/*****
*
* foreground()
*
* Returns
*           Not applicable
*
* Description
*           Interrupt service routine
*
*****/

void foreground()
{
    float  non_filtered;           /* Raw input to filter      */
    float  filter_out;            /* Filtered output          */
    short  dac_out;               /* Filtered output as integer */

    non_filtered = (float)adc();

    filter_out = (a0 * non_filtered) + ( a1 * xk_m1) + (a2 * xk_m2) -
                (b1 * yk_m1) - (b2 * yk_m2);

    dac_out = (short)(filter_out);

    if ( dac_out > DAC_LIM_POS )    /* Limit the output of the DAC */
    {
        dac_out = DAC_LIM_POS ;
    }
    else if ( dac_out < DAC_LIM_NEG )
    {
        dac_out = DAC_LIM_NEG;
    }

    dac( dac_out );

    xk_m2 = xk_m1;                /* Update all delayed terms  */
    xk_m1 = filter_out;
    yk_m2 = yk_m1;
    yk_m1 = non_filtered;
}

```

```

/*****
* pdfiff_float.c
*
*           Floating point implementation of a real time pdfiff motor control
*
*****/

#define FALSE          0
#define TRUE           !FALSE

#define DAC_LIM_POS    2047      /* Positive limit of a 12 bit DAC */
#define DAC_LIM_NEG    -2048     /* Negative limit of a 12 bit DAC */

/* Global variables */

float  kp, ki, kd, ka, kv, kr;    /* PDFF control gains */
float  pe;                        /* Control position error */
float  ref_array[ 1000 ];         /* Velocity reference storage */
float  sout;                      /* Software integrator */
float  sout_m1;                  /* Previous software integrator value */
short  ramp_len;                 /* Ramp length in pulses */
short  ref_end;                  /* End of velocity reference */
short  ref_index;                /* Start of velocity reference */
short  step_height;              /* Step height in pulses */
short  step_len;                 /* Step length in pulses */
short  clock_rate;               /* Interrupt sampling rate in msec. */

/* Function declarations */

extern char toupper();            /* Force a char to upper case */
extern int  getchar();            /* Get a char from the keyboard */
extern short mpg_feedback();      /* Read velocity feedback device */
extern void dac();                /* Digital to analog conversion */
extern void disable_interrupt();  /* Disable real time clock interrupts */
extern void enable_interrupt();   /* Enable real time clock interrupts */
extern void set_clock();          /* Set real time interrupt clock */
extern void zero_dacs();          /* Zero digital to analog converters */
short ref_generator();            /* Generate velocity reference */
void foreground();                /* Interrupt service routine */

```

```

main()
{
    zero_dacs();
    while ( TRUE )                                /* Background loop          */
    {
        puts( "\nCOMMAND ? " );

        switch ( toupper( getchar() ) )
        {
            case 'S':                                /* Stop PDFF control        */
                disable_interrupt();
                zero_dacs();
                break;

            case 'G':                                /* Start PDFF control       */
                pe = 0.0;                             /* Zero PDFF control integrators */
                sout_m1 = 0.0;
                ref_index = 0;                         /* Point to first velocity value */
                enable_interrupt();
                break;

            case 'P':                                /* Enter new KP term        */
                printf( "\nKP = %7.3f", kp );
                puts( "\nKP new value ? " );
                scanf( "%f", &kp );
                break;

            case 'I':                                /* Enter new KI term        */
                printf( "\nKI = %7.3f", ki );
                puts( "\nKI new value ? " );
                scanf( "%f", &ki );
                break;

            case 'D':                                /* Enter new KD term        */
                printf( "\nKD = %7.3f", kd );
                puts( "\nKD new value ? " );
                scanf( "%f", &kd );
                break;

            case 'V':                                /* Enter new KV term        */
                printf( "\nKV = %7.3f", kv );
                puts( "\nKV new value ? " );
                scanf( "%f", &kv );
                break;

            case 'A':                                /* Enter new KA term        */
                printf( "\nKA = %7.3f", ka );
                puts( "\nKA new value ? " );
                scanf( "%f", &ka );
                break;
        }
    }
}

```

```
case 'R':                                /* Enter new KR term          */
    printf( "\nKR = %7.3f", kr );
    puts( "\nKR new value ? " );
    scanf( "%f", &kr );
    break;

case 'T':                                /* Enter new clock rate        */
    Printf( "\nCurrent interrupt clock rate %d (msec.)", clock_rate );
    puts( "\nEnter new interrupt clock rate" );
    scanf( "%d", &clock_rate );
    set_clock( clock_rate );
    break;

case 'W':                                /* Set control velocity reference */
    puts( "\nNumber of ramp samples ?" );
    scanf( "%d", &ramp_len );
    puts( "\nStep height in pulses ?" );
    scanf( "%d", &step_height );
    puts( "\nLength of step in pulses ?" );
    scanf( "%d", &step_len );
    ref_end = ref_generator( ref_array, ramp_len, step_height, step_len );
    break;
}
}
}
```



```

/*****
*
* foreground()
*
* Returns          Not applicable
*
* Description
*          Interrupt service routine
*
*****/

void foreground()
{
    float  dac_out;          /* PDFF control output to DAC      */
    float  ref;              /* PDFF control velocity reference */
    float  vel_error;        /* PDFF control velocity error     */
    float  vel_feedback;     /* Plant velocity feedback         */

    vel_feedback = (float)mpg_feedback();

    if ( ref_index == ref_end )      /* Test if end of ref_array reached */
        ref_index = 0;

    ref = ref_array[ ref_index++ ];  /* Get a reference value            */

    vel_error = ref - vel_feedback;

    pe += vel_error;

    sout = ( sout_m1 * kr ) + ( pe * ki ) + ( kv * ref ) - ( kp * vel_feedback );
    sout_m1 = sout;

    dac_out = sout + ( ka * ref ) - ( kd * vel_feedback );

    if ( dac_out > DAC_LIM_POS )      /* Limit the output of the DAC      */
    {
        dac_out = DAC_LIM_POS ;
    }
    else if ( dac_out < DAC_LIM_NEG )
    {
        dac_out = DAC_LIM_NEG;
    }

    dac( (short)dac_out );           /* PDFF control output to plant     */
}

```

Systems Analysis & Design Programs

(for availability contact author)

Note: Where appropriate, programs download to files for spreadsheet plotting .

ROOTS: ROOTS of Polynomials

Purpose: Calculates the real and/or complex roots of a polynomial with real coefficients.

Remarks: The program requires the order and coefficients of the polynomial. Floating-point overflow may be indicated for high-order polynomials but it should not affect the accuracy.

SIMULT: SIMULTaneous Equations

Purpose: Solves a system of linear algebraic equations given in matrix form. Uses the augmented matrix for input and outputs the solution vector.

Remarks: The program requires the number of equations and the augmented matrix.

SYSTEM: SYSTEM simulation

Purpose: Determines response characteristics of a system for which a mathematical model exists in one of the following forms:

- a) Continuous dynamic equations:
- b) Discrete dynamic equations:
- c) Transfer Function: $G(s)$

The program, also, determines state equations from the transfer function, the transfer function from state equations, and the discrete pulse transfer function.

Remarks: In addition to the parameter values specifying the appropriate system model, the program requires the sampling time (T) for spacing the output points, and the number of points (NO) needed to describe the response. The input function may be a step of arbitrary size, trapezoidal, sin/cos, or any other type of signal available from the keyboard or a file.

FREQC: FREQUENCY analysis - Continuous

Purpose: Calculates the magnitude and phase angle of a continuous transfer function for a specified frequency range and arbitrary number of points per decade. The transfer function must be in rational polynomial form. **Remarks:** The program requires the order and coefficients of the numerator and denominator polynomials, the frequency range in rad/sec, and the number of points per decade.

FREQD: FREQUENCY analysis - Discrete

Purpose: Calculates the magnitude and phase angle of a discrete transfer function for a specified frequency range and arbitrary number of points per decade. The transfer function must be in rational polynomial form. **Remarks:** The program requires the order and coefficients of the numerator and denominator polynomials, the frequency range in rad/sec, the number of points per decade, and the sampling time T.

LONGDI: LONG Division

Purpose: Performs long division between two polynomials.

Remarks: The program requires the order and coefficients of the two polynomials as well as the number of points it must calculate. It can be used, for instance, to calculate time responses from discrete (z) transfer functions.

MATMAN: MATrix MANipulation

Purpose: Manipulates, that is, it can add, subtract, multiply, or invert matrices.

Remarks: The program requires the order and elements of the matrice(s).

POLMAN: POLynomial MANipulation

Purpose: It adds, subtracts, and multiplies rational polynomials.

Remarks: The program requires the order and coefficient of the polynomials involved as well as the operation to be performed (i.e., + or *). **Note:** to subtract polynomials, simply multiply the second polynomial by -1 and add them.

EIGEN: EIGEN Values

Purpose: Calculates the characteristic polynomial and eigen values of a real symmetric matrix.

Remarks: The program requires the order and elements of the matrix.

POLFIT: POLynomial curve FITting

Purpose: Fits a polynomial to a set of points using the method of least squares.

Remarks: The program requires the number of points, their values, and spacing. It determines the coefficients for a specific polynomial and displays the approximation results in tabular form or it can store them in a file for plotting.

SYSID: SYStem IDentification

Purpose: Identifies discrete and/or continuous mathematical system models from a system response described by a set of equally-spaced amplitudes. That is, it determines the discrete transfer function, $G(z)$, and the corresponding continuous transfer function, $G(s)$, that best represent the given points. Results for a selected-order approximation are available in tabular/file form.

Remarks: The program may be used, for instance, to identify a system from its sampled impulse or step response. It requires the number of points, their spacing, and values. It also determines corresponding continuous models. Calculates the continuous transfer function and its equivalent time representation.

DB: DeadBeat Controller Design

Purpose: Designs deadbeat algorithms for open-loop and closed-loop computer control systems. Displays simulation results in tabular form or it stores them for plotting.

Remarks: The program input is similar to that of the program SYSTEM. The coefficients of the algorithm for open-loop control are the elements in the first column of P-INVERSE .

PDFF: PDFF=PID with (P) and (D) control actions in the Feedback + Feedforward Gains.

Purpose: Simulates a computer control loop when the algorithm is of the type where the (I) action is in the forward path, the (P) and (D) actions are in the Feedback, and the (V), (A), and (J) are in the Feedforward. Displays results in tabular/file form.

Remarks: The program is used to tune PDFF/PID control loops. It requires the pulse transfer function of the plant/process which, if not available, can be obtained from the program SYSTEM.

Note: For PID control, set $K_v=K_p$, $K_a=K_d$, and let $K_j=0$.

PDFPG: Similar to PDFF, above, but with Motor Pulse Generator (MPG) for feedback interface and considerably more versatile. It includes graphical output, also.

PFEXP: Partial Fraction EXPansion

Purpose: Expands a transfer function into partial fractions. The denominator of the transfer function must be in factored form. Output consists of the residue for each root.

Remarks: The program requires the order of the numerator, the number of real roots and the number of complex pairs of roots of the denominator.

FILTER: Digital FILTER Design/Simulation

Purpose: To design a digital filter from its analog prototype and to simulate its performance. The filter input can be the combination of a signal plus noise. Coefficient and interphase (ADC/DAC) wordlengths can be varied and the results observed in tabular form or stored in a file for plotting.

Remarks: The coefficient wordlength (n) determines the scale by which decimal coefficients are multiplied and divided. For instance, if $n=10$, then $SCALE=2^{10}=1024$, etc.

A

ACSL 134, 383, 386, 422
 ADC 6, 292, 417
 Aliasing 17, 163
 guard band 164
 sampling rate 163
 Analog 2
 Analog-to-digital
 gain 10
 quantization
 error 9
 noise 9
 scale 10
 two's complement representation 10
 Analog-to-digital converter 6, 9, 316

B

Bandwidth 120
 Bilinear transformation 205
 designing digital filters 305, 418
 primary strip 206
 Block diagram 2, 81
 canonical 81
 standard 81
 Bode plots 115
 Bounded-Input-Bounded-Output 201

C

Caley-Hamilton Theorem 248
 Closed-loop 3, 4
 Computational delay 18
 Computer controlled system 6
 Computer integrated manufacturing 1
 Computer wordlength
 arithmetic overflow 18
 limit cycle 18
 Control
 algorithm 3
 bandwidth 16
 derivative or rate control 394
 disturbances 3
 error 4
 feedback 4
 gains
 derivative 4, 394
 integral 4, 393
 proportional 4, 393
 integral or reset control action 393
 law 3
 manipulated variable 3
 on-off action 393
 phase-lag 428
 phase-lead 428

 proportional band 393
 proportional control action 393
 proportional-integral control action 393
 proportional-integral-derivative 394
 regulators 8
 robust 392
 servomechanisms 8
 single-loop 4
 three-mode controller 394
 track 16
 Control sequence 276
 Controllability 275
 Controlled variable 3, 30
 Controller 3
 Critically-damped 37

D

DAC 6, 292, 323, 408
 Damped natural frequency 35
 Damping ratio 35
 DC gain 214
 DC motor
 positional plant model 403
 velocity plant model 401
 Dead time 120
 Deadbeat control
 closed-loop 348
 state variable design method 356
 controller 322
 disturbance control 365
 design
 state variable method 370
 Z transform method 366
 open-loop 323
 design
 Laplace transform 324
 state variable method 340
 Z transform method 337
 posicast 334
 positive cast 334
 ramp input algorithms 358
 reference and/or disturbance inputs 374
 ramp inputs 381
 PDF control 378
 PI control 381
 step inputs 378, 379
 response step input 325
 Deadbeat response 322
 Deadtime
 plants 430
 Decade 117
 Decibel 117
 Decomposition
 cascade 251, 266
 coupled 268
 complex poles 270

direct 250, 253, 265
 parallel 252, 254, 267
 multiple poles 267
 series 254
 Delay time 47
 Delta table 449
 Derivative time 394
 Difference equation
 characteristic equation 148
 characteristic zeros 148
 digital filter 292
 eigenvalues 148
 real and distinct roots 150
 real and equal roots 151
 Difference equations 144
 Differential equation
 characteristic equation 32
 complementary 32
 degree 25
 forcing function 25
 general solution 33
 homogeneous 25, 32
 independent variable 25
 initial condition 33
 linear 25
 nonhomogeneous 25
 nonlinear 25
 numerical
 differentiation 227
 integration 227
 order 25
 ordinary 25
 partial 25
 settling time 34
 time constant 34
 time-invariant 25
 time-variant 25
 Digital 2
 Digital signal 139
 Digital simulation sampling time selection 273
 Digital-to-analog
 ZOH 12
 Digital-to-analog converter 6, 12
 gain 12
 scale 12
 Direct digital control 6
 Discrete
 difference equation 145
 discrete signals 139
 pulse transfer function 188
 sampling period 139
 sampling time 139
 state transition equation 261
 system 139
 absolute stability 203
 asymptotic stability 203

causal 140
 constant-parameter 140
 deterministic 140
 homogeneous 140
 linear 140
 minimum-phase 202
 poles 201
 realizable 140
 relative stability 203
 single-input-single-output 144
 stable 201
 transfer function 186
 zero-input response 203
 zero-input stable 203
 zeros 201
 transfer function 281
 Discrete-time 139
 Distributed control system 15
 Disturbance control 365
 Dynamic equations 240
 coefficient matrix 240
 control matrix 240
 input vector 240
 output matrix 240
 output vector 240
 state variable vector 240
 system matrix 240
 transmission matrix 240

E

Encoder
 absolute 14
 incremental 14
 pulse generator 14

F

Feedforward 5
 Filter 2, 291
 analog prototype 295
 band-pass filter 293, 301
 center frequency 301
 band-reject filter 292, 301
 cutoff frequency 292
 digital 6
 data window 293
 finite impulse response 293, 311
 design
 lagrangian multipliers
 method 312
 least squares method 314
 prediction, smoothing, and
 differentiation 315
 folding frequency 293
 infinite impulse response 293

- design
 - bilinear transformation method 305
 - frequency warping 305
 - impulse invariant method 302
 - pole-zero matching method 304
 - step invariance 303
- limit cycle 318
- nonrecursive type 293
- Nyquist
 - strip band 305
- recursive type 293
- sampling frequency 293
- form
 - Bessel 299
 - Binomial 299
 - Butterworth 296
 - ITAE 299
- high-pass filter 292, 300
- low pass forms
 - Bessel 300
 - Binomial 299
 - Butterworth 299
 - ITAE 300
- low-pass filter 292
- notch 80, 301
- response comparisons 300
- software advantages 291
- twin-T 80, 301
- Fixed-point (integer) arithmetic 17
- Frequency
 - break 117
 - corner 117
 - cutoff 122
 - cutoff Rate 122
 - gain crossover 125
 - natural 35
 - passband 120
 - phase crossover 125
- Frequency response 114
 - discrete 213
 - logarithmic 115
 - rectangular plots 116
- Frequency spectrum 160
 - band-limited 161
 - complementary 161
 - primary component 161
 - sidebands 161

G

- Gain margin 125
- Gamma Table 449

J

- Jury stability criterion 208

L

- Lagrangian multipliers 312
- Laplace transform
 - complex differentiation 56
 - complex integration 56
 - convolution 57
 - convolution integral 57
 - delay operator 55
 - differentiation 54
 - exponential order 53
 - final value theorem 57
 - initial value theorem 56
 - integration 55
 - linearity 54
 - periodic functions 57
 - transform pair 53
 - translation in frequency 56
 - translation in time 55
 - uniqueness property 54
- Leverrier's algorithm 246
- Limit cycle 318

M

- Maclaurin series 224
- Mason's gain formula 94, 197
- Matrix
 - controllability test 256
 - discrete state transition 261
 - exponential 247
 - fundamental 242
 - modal 257
 - observability 278
 - state transition 242, 247
 - trace 246
- Method of undetermined coefficients 32, 148
- Modern control theory
 - controllability 275
 - observability 275
- Modern system theory 235
- Motor drive system 280
 - positional plant model
 - frequency response 422
- digital
 - control of velocity and positional plants 410
 - positional plant gain 408
 - velocity plant gain 409
 - positional plant gain 288
 - positional plant model 403, 456
 - least squares method 459

- standard filter form tuning
 - zero-acceleration error (ZAE) 404
 - zero-displacement error (ZDE) 404
 - zero-velocity error (ZVE) 404
- velocity plant gain 286
- velocity plant model 401, 457
- Motor pulse generator 14

N

- Negatively damped 36
- Nondeterministic 140
- Nonrealizable (noncausal) 140
- Numerical differentiation
 - backward difference 223
 - forward difference 218
- Numerical integration 221
 - Euler's 222
 - one-fourth (1/4) integration rule, 227
 - quadratic 226
 - rectangular 222, 396
 - Simpson's 226
 - trapezoidal 224, 396
 - Tustin's 224
- Nyquist
 - folding frequency 163
 - frequency 162
 - plot 128
 - rate 162

O

- Observability 275
- Observable 277
- Octave 117
- Offset error 393
- Open-loop 3
- Overdamped 36
- Overshoot 47

P

- Partial fraction expansion
 - complex roots 67
 - Heaviside expansion 65
 - proper function 64
 - real and distinct roots 65
 - real and repeated roots 66
 - residues 65

PDF

- analog 89, 397
 - error constants 93
 - frequency response 123
 - phase margin 126
 - root locus plot 112

- digital 193, 397
 - deadbeat gains 379
 - disturbance control 378
 - frequency response 218
 - root locus plot 211, 213
 - stability 210
 - steady state error 195
 - time-optimal or deadbeat 378, 379
 - tuning plain integrator plant 405

PDFF

- analog 4, 98, 100, 397
 - signal flow diagram 99
 - steady state error 101
 - unit step response 101
- digital 198, 201, 397
 - control and steady state errors
 - positional plant 426
 - velocity plant 426
 - derivative path filter 410, 417
 - frequency response 424
 - second-order feedforward derivative 411
 - second-order feedforward derivative (KJ) 419
 - signal flow graph 196
 - "inside-out" tuning 414
 - software integrator 411
 - tuning
 - double integrator plant 407
 - plain integrator plant 405
 - unit step response 199

PDFF advantages over PID 401

PDFF and PID relationships

- positional plant 404
- velocity plant 402

Phase margin 125

Phase plot 128

Phase-lead design technique 429

Phase-variable canonical form 255

PI

- analog 4, 132
 - digital simulation 134
- digital 360, 369, 407

PID 4

- analog 88, 395
 - block diagram 5
 - error constants 93
 - frequency response 123
 - phase margin 126
 - transfer function 89

digital 388, 395

- block diagram 8
- control and steady state errors
 - positional plant 426
 - velocity plant 426
- gains 396
- optimally-tuned 382

- ideal
 - interacting controller 394
 - noninteracting controller 394
 - parallel controller 394
 - Ziegler-Nichols tuning method 433
 - Plant 3
 - Polar plot 128
 - Pole-zero matching method 429
 - Poles 102
 - pole-zero cancellation 279
 - ringing 205
 - Principle of superposition 20, 140
 - Process 3
 - Profile 8
 - Pulse generator 14, 408, 417
 - noise 417
 - Pulse transfer function 187
- Q**
- Quantization 9
 - error 9, 316
 - addition/multiplication 316
 - number of bits relationship 13, 316
 - level 9
 - step size 9
- R**
- Real-time 18
 - Reference 3
 - Regulator 8
 - Reset time 394
 - Resonance frequency 122
 - Resonance peak 122
 - Rise time 47
 - Robust control 392
 - Root locus diagram 110
 - Root locus discrete systems 211
 - Roots
 - characteristic 32
 - complex roots 38
 - dominant pair 73
 - eigenvalues 32
 - real and distinct roots 36
 - real and repeated roots 37
 - Routh's criterion
 - bilinear transformation 206
 - stability 108
- S**
- Sampled-data signal 139
 - Sampling instants 139
 - Sampling rate 16
 - aliasing 163
 - bandwidth 16
 - theorem 16
 - Sampling theory
 - ideal impulse sampling 158
 - nonideal 158
 - uniform sampling 157
 - Sampling time 8
 - Sampling time selection 273
 - Sensor 4
 - Servomechanism 8
 - Setpoint 3
 - Settling time 36
 - Shannon's sampling theorem 161, 162
 - Signal 20
 - Signal flow graphs
 - forward path 95
 - input nodes 95
 - loop 95
 - loop gain 95
 - mixed nodes 95
 - nodes 94
 - nontouching loops 95
 - output nodes 95
 - path 95
 - path gain 95
 - transmittance 94
 - SIMULAB 135, 231, 383, 422
 - analog PI 135
 - digital PDFF 232, 432
 - heating plant example 435
 - Singularity function 21
 - analog
 - delayed unit-step 21
 - pulse 21
 - unit doublet 22
 - unit triplet 23
 - unit-impulse 22
 - unit-ramp 24
 - unit-step 21
 - discrete
 - delta 142
 - Kronecker delta, 142
 - unit-impulse 142
 - unit-impulse train, 143
 - unit-ramp 143
 - unit-step 141
 - Stability 102
 - absolute stability 104
 - asymptotic stability 104
 - conditionally stable 110
 - critically stable 103
 - discrete
 - bilinear transformation, 206
 - marginally 202
 - unstable 201
 - marginally stable 102

relative stability 104
 unstable 102
 zero-input stable 104
 State controllable 275
 State diagram 239
 State space 235, 236
 digital simulation
 continuous system 273
 state 236
 variables 236
 vector 236
 vector trajectory 236
 State transition 242
 equation 242
 matrix 242
 State variables
 deadbeat control design 340, 356
 discrete 260
 Steady-state 40
 error 92
 acceleration error constant 93, 425
 displacement error constant 92, 425
 velocity error constant 425
 gain 72
 System 2, 19
 analog 5
 analysis 24
 batch 323
 causal 20
 continuous 2
 design 24
 deterministic 20
 discrete 2
 dynamic 2
 homogeneous 20
 hybrid 140
 inherently stable 126
 input 2
 linear 20
 minimum-phase 105
 multivariable 235
 nondeterministic 20
 nondynamic 2
 nonminimum-phase 105
 nonrealizable 21
 observable 277
 output 2
 positional 286
 quadratic mode 73
 realizable 20
 time-invariant 21
 System identification
 discrete transfer function 441
 impulse input method 443
 least-squares approximation 441
 pulse response 448
 pulse transfer function 447

time domain approach 440
 System model reduction
 delta table 449
 gamma table 449
 Rao's model-reduction method 448
 state space model 451
 gamma-delta representations 452

T

Time constant dominant 17
 Time response function
 unit-impulse 71
 unit-pulse 72
 unit-step 71
 Time-invariant 140
 Time-optimal 322
 Transducer 4
 Transfer function 77
 position-to-manipulation 285
 velocity-to-current 286
 Transportation lag 120
 Two's complement 10

U

Ultimate gain 433
 Ultimate method 433
 Ultimate period 433
 Uncontrollable 277
 Undamped 36
 Underdamped 36, 37

W

Wordlength 17

Z

Z transform
 advance (shift Left) 169
 advance operator 147
 bilinear transformation 205
 complex differentiation 170
 complex integration 170
 computer simulations 218
 convolution summation 171
 delay (shift right) 169
 delay operator 147
 final value theorem 171
 initial value theorem 171
 linearity 168
 long division method 176
 multiplication by e^{-sT} (Damping) 170
 one-sided 167

- partial fraction expansion 179
- periodic functions 172
- primary strip 168
- sampling frequency 168
- solution of difference equations 218
- summation 169
- table 173
- unilateral 167
- weighting sequence 171
- Zero-order-hold 12, 164
 - boxcar 164
 - staircase generator 164
- Zeros 32, 102
- Ziegler-Nichols tuning method 433
 - ultimate gain 433
 - ultimate method 433
 - ultimate period 433

International Series on

MICROPROCESSOR-BASED SYSTEMS ENGINEERING

Editor: Professor S. G. Tzafestas, National Technical University, Athens, Greece

1. S.G. Tzafestas (ed.): *Microprocessors in Signal Processing, Measurement and Control*. 1983 ISBN 90-277-1497-5
2. G. Conte and D. Del Corso (eds.): *Multi-Microprocessor Systems for Real-Time Applications*. 1985 ISBN 90-277-2054-1
3. C.J. Georgopoulos: *Interface Fundamentals in Microprocessor-Controlled Systems*. 1985 ISBN 90-277-2127-0
4. N.K. Sinha (ed.): *Microprocessor-Based Control Systems*. 1986 ISBN 90-277-2287-0
5. S.G. Tzafestas and J.K. Pal (eds.): *Real Time Microcomputer Control of Industrial Processes*. 1990 ISBN 0-7923-0779-8
6. S.G. Tzafestas (ed.): *Microprocessors in Robotic and Manufacturing Systems*. ISBN 0-7923-0780-1
7. N.K. Sinha and G.P. Rao (eds.): *Identification of Continuous-Time Systems*. Methodology and Computer Implementation. 1991 ISBN 0-7923-1336-4