DISCRETE SYSTEMS and DIGITAL SIGNAL PROCESSING with MATLAB

Taan S. ElAli

CRC PRESS

Boca Raton London New York Washington, D.C.

DISCRETE SYSTEMS and DIGITAL SIGNAL PROCESSING with MATLAB®

Electrical Engineering Textbook Series

Richard C. Dorf, Series Editor
University of California, Davis

Forthcoming and Published Titles

Applied Vector Analysis
Matiur Rahman and Isaac Mulolani

Continuous Signals and Systems with MATLAB®
Taan ElAli and Mohammad A. Karim

Discrete Systems and Digital Signal Processing with MATLAB®
Taan ElAli

Electromagnetics
Edward J. Rothwell and Michael J. Cloud

Optimal Control Systems
Desineni Subbaram Naidu

DISCRETE SYSTEMS and DIGITAL SIGNAL PROCESSING with MATLAB®

Taan S. ElAli

CRC PRESS

Boca Raton London New York Washington, D.C.

This edition published in the Taylor & Francis e-Library, 2005.

Library of Congress Cataloging-in-Publication Data

Elali, Taan S.
Discrete systems and digital signal processing with MATLAB / Taan S. Elali.
p. cm. (Electrical engineering textbook series)
Includes bibliographical references and index.
ISBN 0-8493-1093-8 (alk. paper)
1. Signal processing--Digital techniques--Mathematics. 2. MATLAB. I. Title II. Series.
TK5102.9.E35 2003
621.382'2—dc21 2003053184
Catalog record is available from the Library of Congress

International Standard Book Number 0-8493-1093-8
Library of Congress Card Number 2003053184

ISBN 0-203-48711-7 Master e-book ISBN

ISBN 0-203-58523-2 (Adobe eReader Format)

Preface

All books on linear systems for undergraduates cover both the discrete and the continuous systems material together in one book. In addition, they also include topics in discrete and continuous filter design, and discrete and continuous state-space representations. However, with this magnitude of coverage, although students typically get a little of both continuous and discrete linear systems, they do not get enough of either. A minimal coverage of continuous linear systems material is acceptable provided there is ample coverage of discrete linear systems. On the other hand, minimal coverage of discrete linear systems does not suffice for either of these two areas. Under the best of circumstances, a student needs solid background in both of these subjects. No wonder these two areas are now being taught separately in so many institutions.

Discrete linear systems is a big area by itself and deserves a single book devoted to it. The objective of this book is to present all the required material that an undergraduate student will need to master this subject matter and to master the use of MATLAB®[1] in solving problems in this subject.

This book is primarily intended for electrical and computer engineering students, and especially for the use of juniors or seniors in these undergraduate engineering disciplines. It can also be very useful to practicing engineers. It is detailed, broad, based on mathematical basic principles and focused, and contains many solved problems using analytical tools as well as MATLAB.

The book is ideal for a one-semester course in the area of discrete linear systems or digital signal processing where the instructor can cover all chapters with ease. Numerous examples are presented within each chapter to illustrate each concept when and where it is presented. In addition, there are end-of-chapter examples that demonstrate the theory presented. Most of the worked-out examples are first solved analytically and then solved using MATLAB in a clear and understandable fashion.

The book concentrates on understanding the subject matter with an easy-to-follow mathematical development and many solved examples. It covers all traditional topics plus stand-alone chapters on transformations and continuous filter design, which should be covered before attempting the IIR digital filter design. These chapters (transformation and continuous filter design) plus the two comprehensive chapters on IIR and FIR digital filter design make this book unique in terms of its thorough and comprehensive

[1] MATLAB is a registered trademark of The Mathworks, Inc. For product information, please contact: The Mathworks, Inc., 3 Apple Hill Drive, Natick, MA 01760-2098. Tel: 508-647-7000. www.mathworks.com.

treatment. A complete chapter on state-space is presented. Another chapter summarizes all representations used in describing discrete linear systems with many examples and illustrations. A very comprehensive chapter on the DFT and FFT is also unique in terms of the FFT applications.

In working with the examples that are solved with MATLAB, the reader will not need to be fluent in this powerful programming language, because they are presented in a self-explanatory way.

To the Instructor: All chapters can be covered in one semester. In a quarter system, Chapters 8 and 9 can be skipped. The MATLAB m-files used with this book can be obtained from the publisher.

To the Student: Familiarity with calculus, differential equations and programming knowledge is desirable. In cases where other background material needs to be presented, that material directly precedes the topic under consideration (just-in-time approach). This unique approach will help the student stay focused on that particular topic. In this book there are three forms of the numerical solutions presented using MATLAB, which allows you to type any command at its prompt and then press the Enter key to get the results. This is one form. Another form is the MATLAB script which is a set of MATLAB commands to be typed and saved in a file. You can run this file by typing its name at the MATLAB prompt and then pressing the Enter key. The third form is the MATLAB function form where it is created and run in the same way as the script file. The only difference is that the name of the MATLAB function file is specific and may not be renamed.

To the Practicing Engineer: The practicing engineer will find this book very useful. The topics of discrete systems and signal processing are of most importance to electrical and computer engineers. The book uses MATLAB, an invaluable tool for the practicing engineer, to solve most of the problems.

Author

Taan S. ElAli, Ph.D., is a full professor of engineering and computer science at Wilberforce University, Wilberforce, Ohio. He received his B.S. degree in electrical engineering in 1987 from The Ohio State University, Columbus, an M.S. degree in systems engineering in 1989 from Wright State University, Dayton, Ohio and an M.S. in applied mathematics and Ph.D. in electrical engineering, with a specialization in systems, from the University of Dayton in 1991 and 1993, respectively. He has more than 12 years teaching and research experience in the areas of discrete and continuous signals and systems. He was listed in "Who's Who Among America's Teachers" for 1998 and 2000. He is also listed in "Who's Who in America" for 2004.

Dr. ElAli has contributed many journal articles and conference presentations in the area of systems. He has been extensively involved in the establishment of the electrical and computer degree programs and curriculum development at Wilberforce University. He is the author of *Introduction to Engineering and Computer Science with C and MATLAB* and *Continuous Signals and Systems with MATLAB*. Dr. ElAli has contributed a chapter to *The Engineering Handbook* published by CRC Press.

Acknowledgments

I would like to thank the CRC Press International team. Special thanks go to Nora Konopka, who encouraged me greatly when I discussed this project with her for the first time. She has reaffirmed my belief that this book is very much needed. Helena Redshaw and Sylvia Wood were also very helpful in the production of the book.

Thanks also to Mr. Dlamini, Ms. Jordan, Ms. Randaka, and Mr. Oluyitan from Wilberforce University, who helped in the typing of the manuscript.

Dedication

This book is dedicated first to the glory of Almighty God. It is dedicated next to my beloved parents, father Saeed and mother Shandokha. May Allah have mercy on their souls. It is dedicated then to my wife Salam; my beloved children, Nusayba, Ali and Zayd; my brothers, Mohammad and Khaled; and my sisters, Sabha, Khulda, Miriam and Fatma. I ask the Almighty to have mercy on us and to bring peace, harmony and justice to all.

Table of Contents

1 Signal Representation 1
1.1 Introduction 1
1.2 Why Do We Discretize Continuous Systems? 2
1.3 Periodic and Nonperiodic Discrete Signals 3
1.4 The Unit Step Discrete Signal 4
1.5 The Impulse Discrete Signal 6
1.6 The Ramp Discrete Signal 6
1.7 The Real Exponential Discrete Signal 7
1.8 The Sinusoidal Discrete Signal 7
1.9 The Exponentially Modulated Sinusoidal Signal 11
1.10 The Complex Periodic Discrete Signal 11
1.11 The Shifting Operation 15
1.12 Representing a Discrete Signal Using Impulses 16
1.13 The Reflection Operation 18
1.14 Time Scaling 19
1.15 Amplitude Scaling 20
1.16 Even and Odd Discrete Signal 21
1.17 Does a Discrete Signal Have a Time Constant? 23
1.18 Basic Operations on Discrete Signals 25
1.18.1 Modulation 25
1.18.2 Addition and Subtraction 25
1.18.3 Scalar Multiplication 25
1.18.4 Combined Operations 26
1.19 Energy and Power Discrete Signals 28
1.20 Bounded and Unbounded Discrete Signals 30
1.21 Some Insights: Signals in the Real World 30
1.21.1 The Step Signal 31
1.21.2 The Impulse Signal 31
1.21.3 The Sinusoidal Signal 31
1.21.4 The Ramp Signal 31
1.21.5 Other Signals 32
1.22 End of Chapter Examples 32
1.23 End of Chapter Problems 50

2 The Discrete System 55
2.1 Definition of a System 55
2.2 Input and Output 55
2.3 Linear Discrete Systems 56
2.4 Time Invariance and Discrete Signals 58

2.5 Systems with Memory 60
2.6 Causal Systems 61
2.7 The Inverse of a System 62
2.8 Stable System 63
2.9 Convolution 64
2.10 Difference Equations of Physical Systems 68
2.11 The Homogeneous Difference Equation and Its Solution 69
2.11.1 Case When Roots Are All Distinct 71
2.11.2 Case When Two Roots Are Real and Equal 72
2.11.3 Case When Two Roots Are Complex 72
2.12 Nonhomogeneous Difference Equations and their Solutions 73
2.12.1 How Do We Find the Particular Solution? 75
2.13 The Stability of Linear Discrete Systems: The Characteristic Equation 75
2.13.1 Stability Depending On the Values of the Poles 75
2.13.2 Stability from the Jury Test 76
2.14 Block Diagram Representation of Linear Discrete Systems 78
2.14.1 The Delay Element 79
2.14.2 The Summing/Subtracting Junction 79
2.14.3 The Multiplier 79
2.15 From the Block Diagram to the Difference Equation 81
2.16 From the Difference Equation to the Block Diagram: A Formal Procedure 82
2.17 The Impulse Response 85
2.18 Correlation 87
2.18.1 Cross-Correlation 87
2.18.2 Auto-Correlation 89
2.19 Some Insights 90
2.19.1 How Can We Find These Eigenvalues? 90
2.19.2 Stability and Eigenvalues 91
2.20 End of Chapter Examples 91
2.21 End of Chapter Problems 134

3 The Fourier Series and the Fourier Transform of Discrete Signals 143
3.1 Introduction 143
3.2 Review of Complex Numbers 143
3.2.1 Definition 145
3.2.2 Addition 145
3.2.3 Subtraction 145
3.2.4 Multiplication 145
3.2.5 Division 146
3.2.6 From Rectangular to Polar 146
3.2.7 From Polar to Rectangular 146

3.3 The Fourier Series of Discrete Periodic Signals 147
3.4 The Discrete System with Periodic Inputs: The Steady-State Response 150
3.4.1 The General Form for $yss(n)$ 153
3.5 The Frequency Response of Discrete Systems 154
3.5.1 Properties of the Frequency Response 157
3.5.1.1 The Periodicity Property 157
3.5.1.2 The Symmetry Property 157
3.6 The Fourier Transform of Discrete Signals 159
3.7 Convergence Conditions 161
3.8 Properties of the Fourier Transform of Discrete Signals 162
3.8.1 The Periodicity Property 162
3.8.2 The Linearity Property 162
3.8.3 The Discrete-Time Shifting Property 163
3.8.4 The Frequency Shifting Property 163
3.8.5 The Reflection Property 163
3.8.6 The Convolution Property 164
3.9 Parseval's Relation and Energy Calculations 167
3.10 Numerical Evaluation of the Fourier Transform of Discrete Signals 168
3.11 Some Insights: Why Is This Fourier Transform? 172
3.11.1 The Ease in Analysis and Design 172
3.11.2 Sinusoidal Analysis 173
3.12 End of Chapter Examples 173
3.13 End of Chapter Problems 189

4 The *z*-Transform and Discrete Systems 195
4.1 Introduction 195
4.2 The Bilateral *z*-Transform 195
4.3 The Unilateral *z*-Transform 197
4.4 Convergence Considerations 200
4.5 The Inverse *z*-Transform 203
4.5.1 Partial Fraction Expansion 203
4.5.2 Long Division 206
4.6 Properties of the *z*-Transform 207
4.6.1 Linearity Property 207
4.6.2 Shifting Property 207
4.6.3 Multiplication by e^{-an} 209
4.6.4 Convolution 210
4.7 Representation of Transfer Functions as Block Diagrams 210
4.8 $x(n)$, $h(n)$, $y(n)$, and the *z*-Transform 212
4.9 Solving Difference Equation using the *z*-Transform 214
4.10 Convergence Revisited 216
4.11 The Final Value Theorem 219
4.12 The Initial-Value Theorem 219

4.13 Some Insights: Poles and Zeroes 220
4.13.1 The Poles of the System 220
4.13.2 The Zeros of the System 221
4.13.3 The Stability of the System 221
4.14 End of Chapter Exercises 221
4.15 End of Chapter Problems 255

5 State-Space and Discrete Systems 265
5.1 Introduction 265
5.2 A Review on Matrix Algebra 266
5.2.1 Definition, General Terms and Notations 266
5.2.2 The Identity Matrix 266
5.2.3 Adding Two Matrices 267
5.2.4 Subtracting Two Matrices 267
5.2.5 Multiplying A Matrix by a Constant 267
5.2.6 Determinant of a Two-by-Two Matrix 268
5.2.7 Transpose of A Matrix 268
5.2.8 Inverse of A Matrix 268
5.2.9 Matrix Multiplication 269
5.2.10 Eigenvalues of a Matrix 269
5.2.11 Diagonal Form of a Matrix 269
5.2.12 Eigenvectors of a Matrix 269
5.3 General Representation of Systems in State-Space 270
5.3.1 Recursive Systems 270
5.3.2 Nonrecursive Systems 272
5.3.3 From the Block Diagram to State-Space 273
5.3.4 From the Transfer Function $H(z)$ to State-Space 276
5.4 Solution of the State-Space Equations in the z-Domain 283
5.5 General Solution of the State Equation in Real-Time 284
5.6 Properties of A^n and Its Evaluation 285
5.7 Transformations for State-Space Representations 289
5.8 Some Insights: Poles and Stability 291
5.9 End of Chapter Examples 292
5.10 End of Chapter Problems 322

6 Modeling and Representation of Discrete Linear Systems 329
6.1 Introduction 329
6.2 Five Ways of Representing Discrete Linear Systems 330
6.2.1 From the Difference Equation to the Other Four Representations 330
6.2.1.1 The Difference Equation Representation 330
6.2.1.2 The Impulse Response Representation 331
6.2.1.3 The z-Transform Representation 332
6.2.1.4 The State-Space Representation 333
6.2.1.5 The Block Diagram Representation 334

6.2.2 From the Impulse Response to the Other Four Representations 335
6.2.2.1 The Impulse Response Representation 335
6.2.2.2 The Transfer Function Representation 335
6.2.2.3 The Different Equation Representation 336
6.2.2.4 The State-Space Representation 336
6.2.2.5 The Block Diagram Representation 337
6.2.3 From the Transfer Function to the Other Four Representations 337
6.2.3.1 The Transfer Function Representation 337
6.2.3.2 The Impulse Response Representation 338
6.2.3.3 The Difference Equation Representation 338
6.2.3.4 The State-Space Representation 339
6.2.3.5 The Block Diagram Representation 339
6.2.4 From the State-Space to the Other Four Representations 340
6.2.4.1 The State-Space Representation 340
6.2.4.2 The Transfer Function Representation 340
6.2.4.3 The Impulse Response Representation 341
6.2.4.4 The Difference Equation Representation 341
6.2.4.5 The Block Diagram Representation 342
6.2.5 From the Block Diagram to the Other Four Representations 343
6.2.5.1 The State-Space Representation 343
6.2.5.2 The Transfer Function Representation 344
6.2.5.3 The Impulse Response Representation 345
6.2.5.4 The Difference Equation Representation 345
6.3 Some Insights: The Poles Considering Different Outputs within the Same System 346
6.4 End of Chapter Exercises 346
6.5 End of Chapter Problems 361

7 The Discrete Fourier Transform and Discrete Systems 365
7.1 Introduction 365
7.2 The Discrete Fourier Transform and the Finite-Duration Discrete Signals 366
7.3 Properties of the Discrete Fourier Transform 367
7.3.1 How Does the Defining Equation Work? 367
7.3.2 The DFT Symmetry 368
7.3.3 The DFT Linearity 370
7.3.4 The Magnitude of the DFT 371
7.3.5 What Does k in $X(k)$, the DFT, Mean? 372
7.4 The Relation the DFT Has with the Fourier Transform of Discrete Signals, the z-Transform and the Continuous Fourier Transform 373

7.4.1 The DFT and the Fourier Transform of $x(n)$ 373
7.4.2 The DFT and the z-Transform of $x(n)$ 374
7.4.3 The DFT and the Continuous Fourier Transform of $x(t)$ 376
7.5 Numerical Computation of the DFT 377
7.6 The Fast Fourier Transform: A Faster Way of Computing the DFT 378
7.7 Applications of the DFT 380
7.7.1 Circular Convolution 380
7.7.2 Linear Convolution 384
7.7.3 Approximation to the Continuous Fourier Transform 385
7.7.4 Approximation to the Coefficients of the Fourier Series and the Average Power of the Periodic Signal $x(t)$ 387
7.7.5 Total Energy in the Signal $x(n)$ and $x(t)$ 391
7.7.6 Block Filtering 393
7.7.7 Correlation 395
7.8 Some Insights 395
7.8.1 The DFT Is the Same as the fft 395
7.8.2 The DFT Points Are the Samples of the Fourier Transform of $x(n)$ 395
7.8.3 How Can We Be Certain That Most of the Frequency Contents of $x(t)$ Are in the DFT? 395
7.8.4 Is the Circular Convolution the Same as the Linear Convolution? 396
7.8.5 Is $|\boldsymbol{X}(\boldsymbol{w})| \cong |\boldsymbol{X}(\boldsymbol{k})|$? 396
7.8.6 Frequency Leakage and the DFT 396
7.9 End of Chapter Exercises 396
7.10 End of Chapter Problems 415

8 Analogue Filter Design 421
8.1 Introduction 421
8.2 Analogue Filter Specifications 422
8.3 Butterworth Filter Approximation 425
8.4 Chebyshev Filters 428
8.4.1 Type I Chebyshev Approximation 428
8.4.2 Inverse Chebyshev Filter (Type II Chebyshev Filters) 431
8.5 Elliptic Filter Approximation 433
8.6 Bessel Filters 434
8.7 Analogue Frequency Transformation 437
8.8 Analogue Filter Design using MATLAB 438
8.8.1 Order Estimation Functions 439
8.8.2 Analogue Prototype Design Functions 440
8.8.3 Complete Classical IIR Filter Design 440
8.8.4 Analogue Frequency Transformation 442
8.9 How Do We Find the Cut-Off Frequency Analytically? 443
8.10 Limitations 447

8.11 Comparison between Analogue Filter Types 447
8.12 Some Insights: Filters with High Gain vs. Filters with Low Gain and the Relation between the Time Constant and the Cut-Off Frequency for First-Order Circuits and the Series RLC Circuit 448
8.13 End of Chapter Examples 449
8.14 End of Chapter Problems 479

9 Transformations between Continuous and Discrete Representations 487
9.1 The Need for Converting Continuous Signal to a Discrete Signal 487
9.2 From the Continuous Signal to Its Binary Code Representation 488
9.3 From the Binary Code to the Continuous Signal 490
9.4 The Sampling Operation 490
9.4.1 Ambiguity in Real-Time Domain 490
9.4.2 Ambiguity in the Frequency Domain 492
9.4.3 The Sampling Theorem 493
9.4.4 Filtering before Sampling 494
9.4.5 Sampling and Recovery of the Continuous Signal 496
9.5 How Do We Discretize the Derivative Operation? 500
9.6 Discretization of the State-Space Representation 504
9.7 The Bilinear Transformation and the Relationship between the Laplace-Domain and the z-Domain Representations 506
9.8 Other Transformation Methods 512
9.8.1 Impulse Invariance Method 512
9.8.2 The Step Invariance Method 512
9.8.3 The Forward Difference Method 512
9.8.4 The Backward Difference Method 512
9.8.5 The Bilinear Transformation 512
9.9 Some Insights 515
9.9.1 The Choice of the Sampling Interval T_s 515
9.9.2 The Effect of Choosing T_s on the Dynamics of the System 515
9.9.3 Does Sampling Introduce Additional Zeros for the Transfer Function $H(z)$? 516
9.10 End of Chapter Examples 517
9.11 End of Chapter Problems 534

10 Infinite Impulse Response (IIR) Filter Design 541
10.1 Introduction 541
10.2 The Design Process 542
10.2.1 Design Based on the Impulse Invariance Method 542
10.2.2 Design Based on the Bilinear Transform Method 545
10.3 IIR Filter Design Using MATLAB 548

10.3.1 From the Analogue Prototype to the IIR Digital Filter 548
10.3.2 Direct Design 548
10.4 Some Insights 550
10.4.1 The Difficulty in Designing IIR Digital Filters in the z-Domain 550
10.4.2 Using the Impulse Invariance Method 552
10.4.3 The Choice of the Sampling Interval T_s 552
10.5 End of Chapter Examples 552
10.6 End of Chapter Problems 584

11 Finite Impulse Response (FIR) Digital Filters 591
11.1 Introduction 591
11.1.1 What Is an FIR Digital Filter? 591
11.1.2 A Motivating Example 591
11.2 FIR Filter Design 594
11.2.1 Stability of FIR Filters 596
11.2.2 Linear Phase of FIR Filters 597
11.3 Design Based on the Fourier Series: The Windowing Method 598
11.3.1 Ideal Lowpass FIR Filter Design 599
11.3.2 Other Ideal Digital FIR Filters 601
11.3.3 Windows Used in the Design of the Digital FIR Filter 602
11.3.4 Which Window Gives the Optimal $h(n)$? 604
11.3.5 Design of a Digital FIR Differentiator 605
11.3.6 Design of Comb FIR Filters 607
11.3.7 Design of a Digital Shifter: The Hilbert Transform Filter 609
11.4 From IIR to FIR Digital Filters: An Approximation 610
11.5 Frequency Sampling and FIR Filter Design 610
11.6 FIR Digital Design Using MATLAB 611
11.6.1 Design Using Windows 611
11.6.2 Design Using Least-Squared Error 612
11.6.3 Design Using the Equiripple Linear Phase 612
11.6.4 How to Obtain the Frequency Response 612
11.7 Some Insights 613
11.7.1 Comparison with IIR Filters 613
11.7.2 The Different Methods Used in the FIR Filter Design 613
11.8 End of the Chapter Examples 614
11.9 End of Chapter Problems 644

References 649

Index 651

1

Signal Representation

1.1 Introduction

We experience signals of various types almost on a continual basis in our daily life. The blowing of the wind is an example of a continuous wave. One can plot the strength of the wind wave as a function of time. We can plot the velocity of this same wave and the distance it travels as a function of time as well. When we speak, continuous signals are generated. These spoken word signals travel from one place to another so that another person can hear them. These are our familiar sound waves.

When a radar system detects a certain object in the sky, an electromagnetic signal is sent. This signal leaves the radar system and travels the distance in the air until it hits the target object, which then reflects back to the sending radar to be analyzed, where it is decided whether the target is present. We understand that this electromagnetic signal, whether it is the one being sent or the one being received by the radar, is attenuated (its strength reduced) as it travels away from the radar station. Thus, the attenuation of this electromagnetic signal can be plotted as a function of time. If you vertically attach a certain mass to a spring at one end while the other end is fixed and then pull the mass, oscillations are created such that the spring's length increases and decreases until finally the oscillations stop. The oscillations produced are a signal that also dies out with increasing time. This signal, for example, can represent the length of the spring as a function of time. Signals can also appear as electric waves. Examples are voltages and currents on long transmission lines. Voltage value gets reduced as the impressed voltage travels on transmission lines from one city to another. Therefore we can represent these voltages as signals as well and plot them in terms of time. When we discharge or charge a capacitor, the rate of charging or discharging depends on the time factor (other factors also exist). Charging and discharging the capacitor can be represented thus as voltage across the capacitor terminal as a function of time. These are a few examples of continuous signals that exist in nature that can be modeled mathematically as signals that are functions of various parameters.

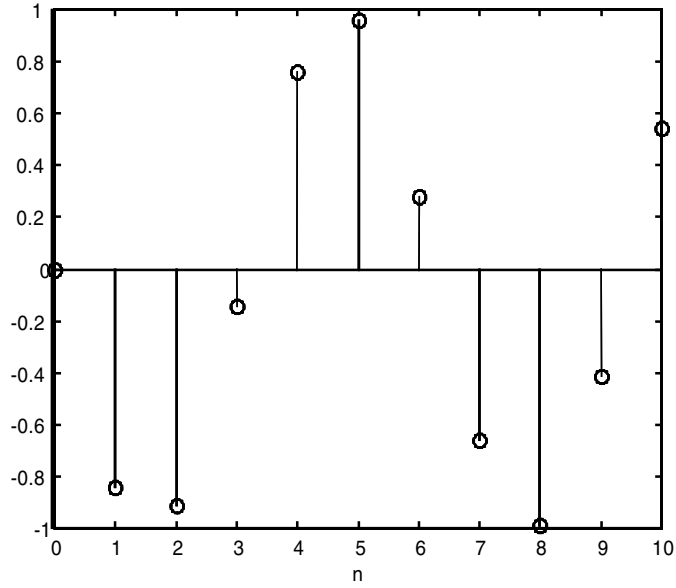

FIGURE 1.1 An example of a discrete signal.

Signals can be continuous or discrete. We will consider only one-dimensional discrete signals in this book. A discrete signal is shown in Figure 1.1. Discrete signals are defined only at discrete instances of time. They can be samples of continuous signals, or they may exist naturally. A discrete signal that is a result of sampling a continuous signal is shown in Figure 1.2. An example of a signal that is inherently discrete is a set of any measurements that are taken physically at discrete instances of time.

In most system operations, we sample a continuous signal, quantize the sample values and finally digitize the values so a computer can operate on them (the computer works only on digital signals).

In this book we will work with discrete signals that are samples of continuous signals. In Figure 1.2, we can see that the continuous signal is defined at all times, while the discrete signal is defined at certain instances of time. The time between sample values is called the sampling period. We will label the time axis for the discrete signal as n, where the sampled values are represented at … –1, 0, 1, 2, 3 … and n is an integer.

1.2 Why Do We Discretize Continuous Systems?

Engineers used to build analogue systems to process a continuous signal. These systems are very expensive, they can wear out very fast as time passes and they are inaccurate most of the time. This is due in part to thermal

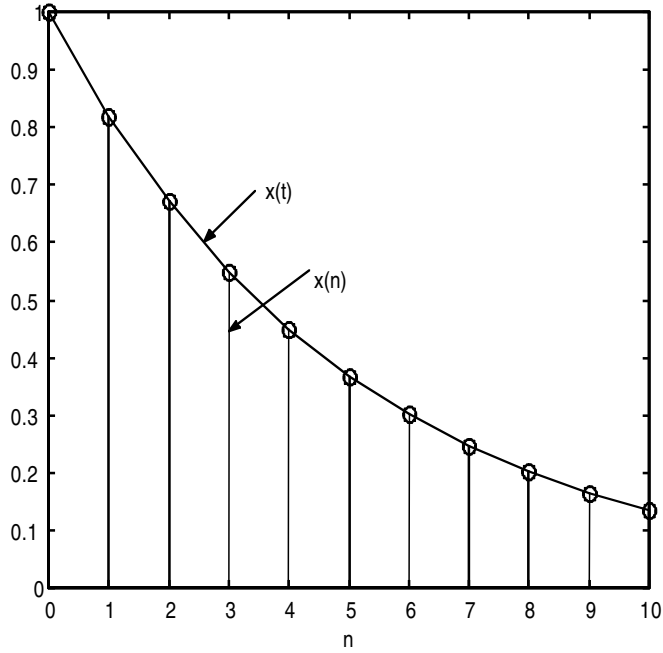

FIGURE 1.2 A sampled continuous signal.

interferences. Also, any time modification of a certain design is desired, it may be necessary to replace whole parts of the overall system.

On the other hand, using discrete signals, which will then be quantized and digitized, to work as inputs to digital systems such as a computer, renders the results more accurate and immune to such thermal interferences that are always present in analogue systems.

Some real-life systems are inherently unstable, and thus we may design a controller to stabilize the unstable physical system. When we implement the designed controller as a digital system that has its inputs and outputs as digital signals, there is a need to sample the continuous inputs to this digital computer. Also, a digital controller can be changed simply by changing a program code.

1.3 Periodic and Nonperiodic Discrete Signals

A discrete signal $x(n)$ is periodic if

$$x(n) = x(n + kN) \tag{1.1}$$

where k is an integer and N is the period which is an integer as well. A periodic discrete signal is shown in Figure 1.3. This signal has a period of 3. This periodic signal repeats every $N = 3$ instances.

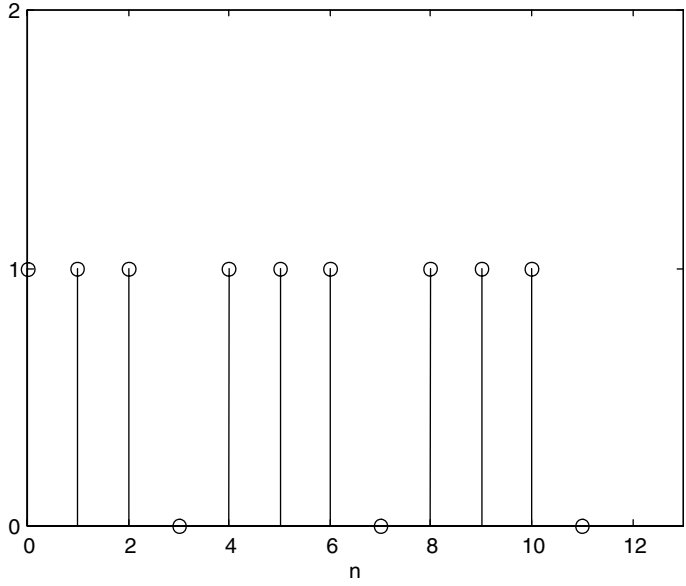

FIGURE 1.3 A periodic discrete signal.

Example 1.1

Consider the two signals in Figure 1.4. Are they periodic?

Solution

The first signal is periodic but the second is not. This can be seen by observing the signals in the figure.

1.4 The Unit Step Discrete Signal

Mathematically, a unit step discrete signal is written as

$$Au(n) = \begin{cases} A & n \geq 0 \\ 0 & n < 0 \end{cases} \tag{1.2}$$

where A is the amplitude of the unit step discrete signal. This signal is shown in Figure 1.5.

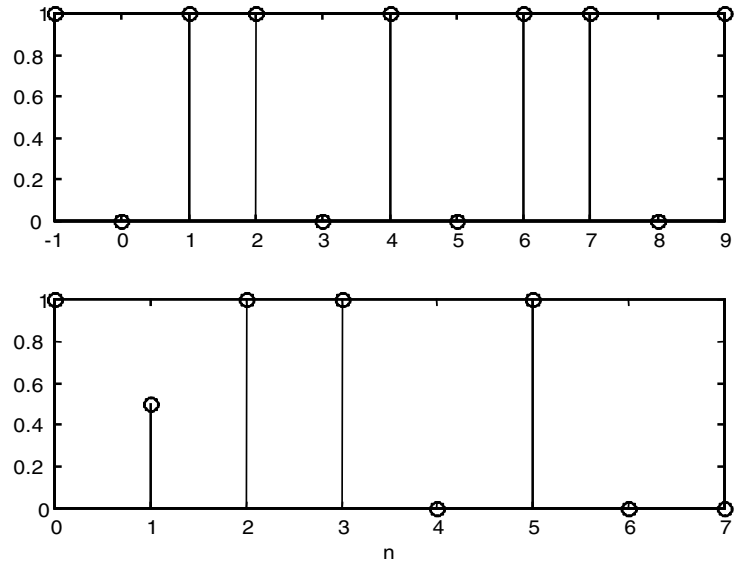

FIGURE 1.4 Signals for Example 1.1.

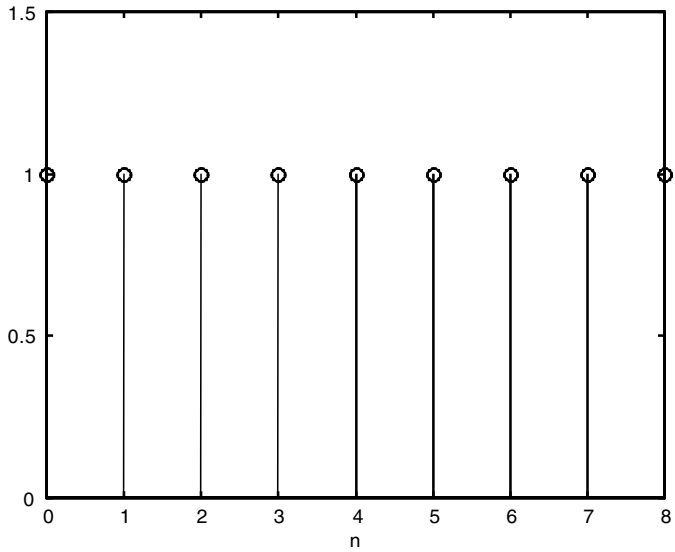

FIGURE 1.5 The step discrete signal.

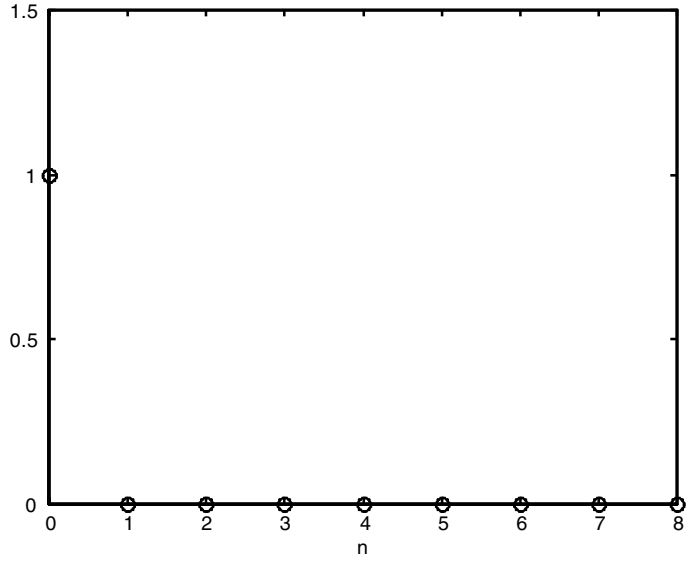

FIGURE 1.6 The impulse discrete signal.

1.5 The Impulse Discrete Signal

Mathematically the impulse discrete signal is written as

$$A\delta(n) = \begin{cases} A & n = 0 \\ 0 & n \neq 0 \end{cases} \tag{1.3}$$

where, again, A is the strength of the impulse discrete signal. This signal is shown in Figure 1.6 with $A = 1$.

1.6 The Ramp Discrete Signal

Mathematically the ramp discrete signal is written as

$$Ar(n) = An \quad -\infty < n < +\infty \tag{1.4}$$

where A is the slope of the ramp discrete signal. This signal is shown in Figure 1.7.

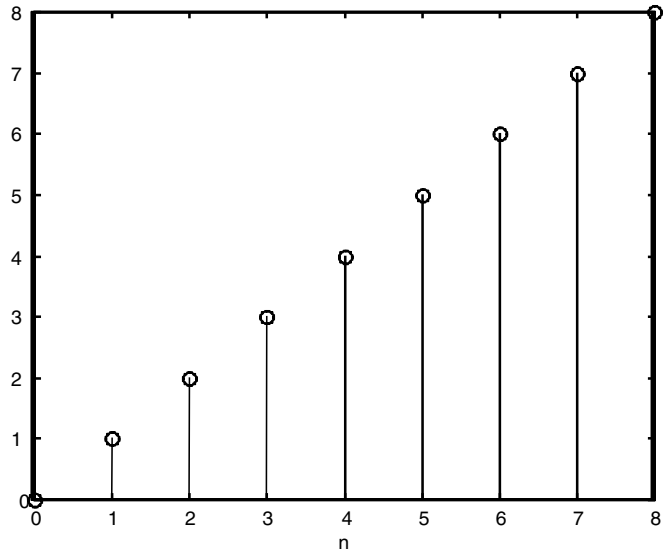

FIGURE 1.7 The ramp discrete signal.

1.7 The Real Exponential Discrete Signal

Mathematically, the real exponential discrete signal is written as

$$x(n) = A\alpha^n \tag{1.5}$$

when α is a real value. If $0 < \alpha < 1$ then the signal $x(n)$ will decay exponentially as shown in Figure 1.8. If $0 > \alpha > 1$ then the signal $x(n)$ will grow without bound as shown in Figure 1.9.

1.8 The Sinusoidal Discrete Signal

Mathematically, the sinusoidal discrete signal is written as

$$x(n) = A\cos\left(\theta_0 n + \phi\right) \quad -\infty < n < +\infty \tag{1.6}$$

where A is the amplitude, θ_0 is the angular frequency and ϕ is the phase. A sinusoidal discrete signal is shown in Figure 1.10. The period of the sinusoidal

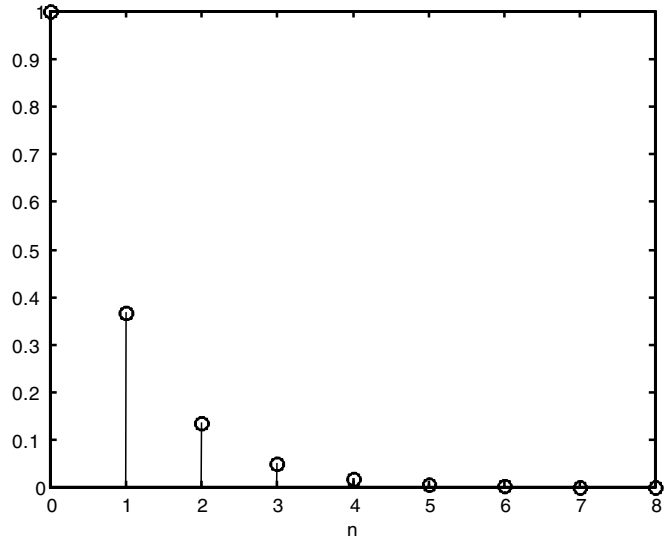

FIGURE 1.8 The decaying exponential discrete signal.

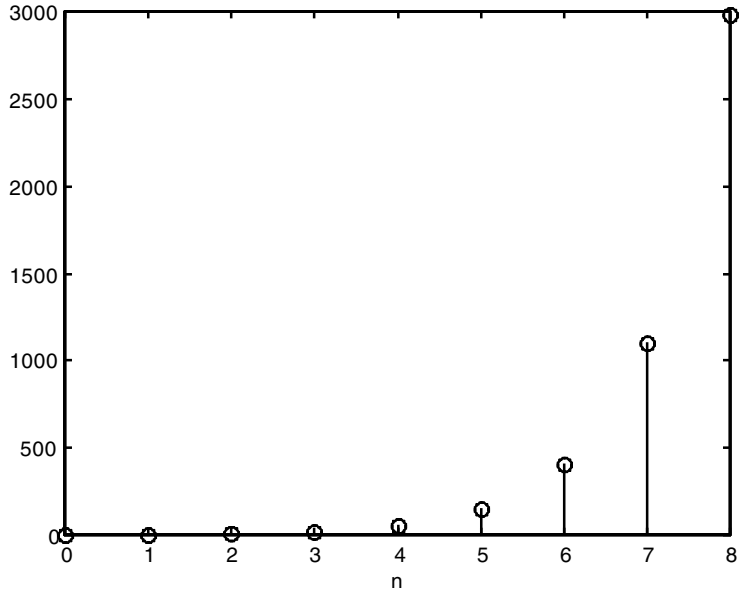

FIGURE 1.9 The growing exponential discrete signal.

discrete signal $x(n)$, if it is periodic, is N. This period can be found as in the following development.

$x(n)$ is the magnitude A times the real part of $e^{j(\theta_0 n+\varphi)}$. But ϕ is the phase and A is the magnitude, and neither has an effect on the period. So if $e^{j\theta_0 n}$ is periodic, then

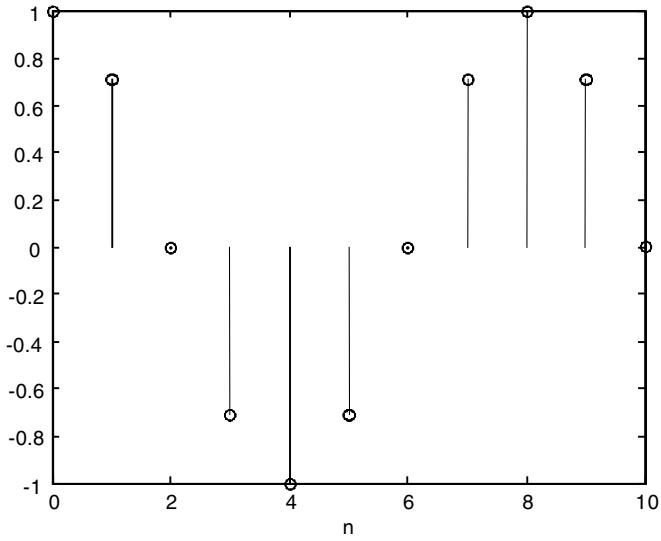

FIGURE 1.10 The sinusoidal discrete signal.

$$e^{j\theta_0 n} = e^{j\theta_0 (n+N)}$$

or

$$e^{j\theta_0 n} = e^{j\theta_0 n} e^{j\theta_0 N}$$

If we divide the above equation by $e^{j\theta_0 n}$ we get

$$1 = e^{j\theta_0 N} = \cos(\theta_0 N) + j\sin(\theta_0 N)$$

For the above equation to be true the following two conditions must be true:

$$\cos(\theta_0 N) = 1$$

and

$$\sin(\theta_0 N) = 0$$

These two conditions can be satisfied only if $\theta_0 N$ is an integer multiple of 2. In other words, $x(n)$ is periodic if

$$\theta_0 N = 2\pi k$$

where k is an integer. This can be written as

$$\frac{2\pi}{\theta_0} = \frac{N}{k}$$

If $\frac{N}{k}$ is a rational number (ratio of two integers) then $x(n)$ is periodic and the period is

$$N = k\left(\frac{2\pi}{\theta_0}\right) \tag{1.7}$$

The smallest value of N that satisfies the above equation is called the fundamental period. If $2\pi/\theta_0$ is not a rational number, then $x(n)$ is not periodic.

Example 1.2

Consider the following continuous signal for the current

$$i(t) = \cos(20\pi t)$$

which is sampled at 12.5 ms. Will the resulting discrete signal be periodic?

Solution

The continuous radian frequency is $w = 20\pi$ radians. Since the sampling interval T_s is 12.5 msec = 0.0125 sec, then

$$x(n) = \cos\left(2\pi(10)(0.0175)n\right) = \cos\left(\frac{2\pi}{8}n\right) = \cos\left(\frac{\pi}{4}n\right)$$

Since for periodicity we must have

$$\frac{2\pi}{\theta_o} = \frac{N}{k}$$

we get

$$\frac{2\pi}{2\pi/8} = \frac{N}{k} = \frac{16\pi}{2\pi} = \frac{8}{1}$$

For $k = 1$ we have $N = 8$, which is the fundamental period.

Example 1.3

Are the following discrete signals periodic? If so, what is the period for each?

1. $x(n) = 2\cos\left(\sqrt{2}\pi n\right)$
2. $x(n) = 20\cos(\pi n)$

Solution

For the first signal $\theta_0 = \sqrt{2}\pi$ and the ratio $\theta_0/2\pi$ must be a rational number.

$$\frac{\theta_0}{2\pi} = \frac{\sqrt{2}\pi}{2\pi} = \frac{\sqrt{2}}{2}$$

This is clearly not a rational number and therefore the signal is not periodic. For the second signal $\theta_0 = \pi$ and the ratio $\theta_0/2\pi = \pi/2\pi = 1/2$ is a rational number. Thus the signal is periodic and the period is calculated by setting

$$\frac{1}{2} = \frac{N}{k}$$

For $k = 2$ we get $N = 1$. This N is the fundamental period.

1.9 The Exponentially Modulated Sinusoidal Signal

The exponentially modulated sinusoidal signal is written mathematically as

$$x(n) = A\alpha^n \cos\left(\theta_0 n + \phi\right) \tag{1.8}$$

If $\cos(\theta_0 n + \phi)$ is periodic and $0 < \alpha < 1$, $x(n)$ is a decaying exponential discrete signal as shown in Figure 1.11. If $\cos(\theta_0 n + \phi)$ is periodic and α is not in the interval $0 < \alpha < 1$, $x(n)$ is a growing exponential discrete signal as shown in Figure 1.12. If $\cos(\theta_0 n + \phi)$ is nonperiodic and $0 < \alpha < 1$, $x(n)$ will not decay exponentially in a regular fashion as in Figure 1.13. If $\cos(\theta_0 n + \phi)$ is nonperiodic and α is not in the interval $0 < \alpha < 1$, $x(n)$ will grow irregularly without bounds as in Figure 1.14.

1.10 The Complex Periodic Discrete Signal

A complex discrete signal is represented mathematically as

$$x(n) = Ae^{j\alpha n} \qquad -\infty < n < +\infty \tag{1.9}$$

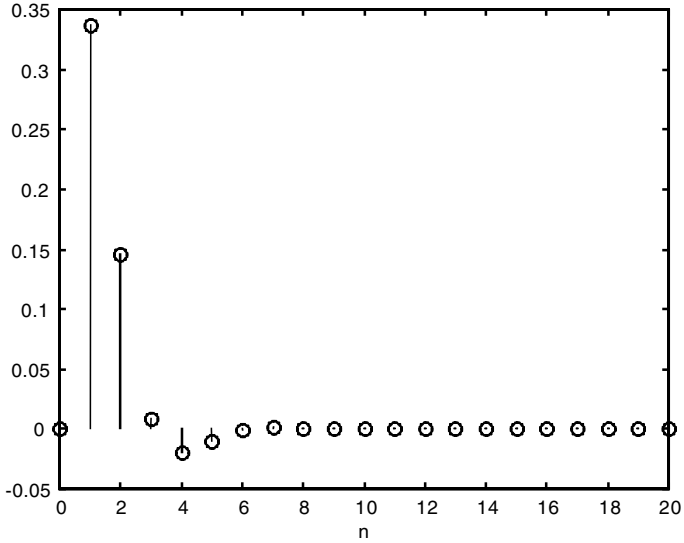

FIGURE 1.11 The decaying sinusoidal discrete signal.

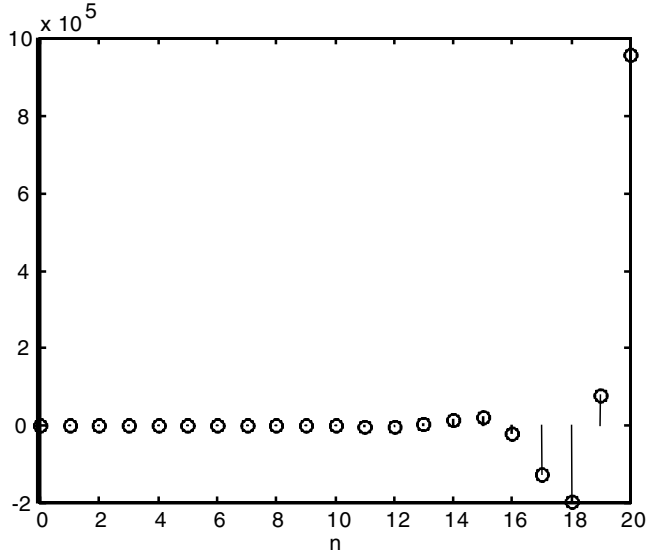

FIGURE 1.12 The growing sinusoidal discrete signal.

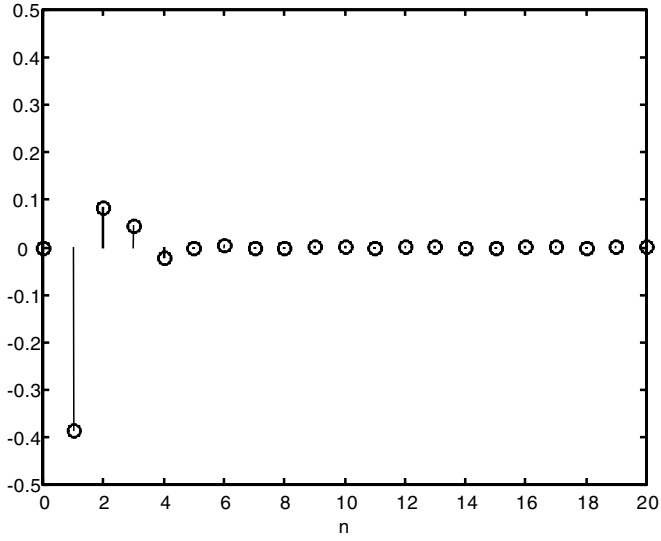

FIGURE 1.13 The irregularly decaying modulated sinusoidal discrete signal.

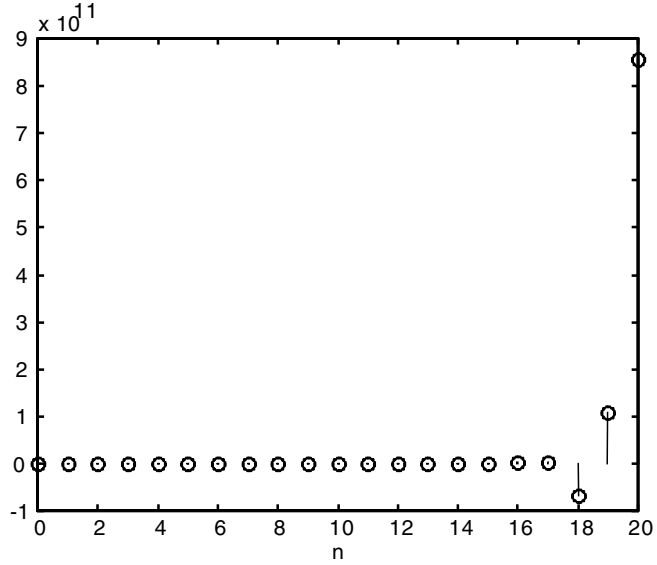

FIGURE 1.14 The irregularly growing modulated sinusoidal discrete signal.

where α is a real number. For $x(n)$ to be periodic we must have

$$x(n) = x(n+N)$$

or

$$Ae^{j\alpha n} = Ae^{j\alpha(n+N)}$$

Simplifying we get

$$Ae^{j\alpha n} = Ae^{j\alpha n}Ae^{j\alpha N}$$

If we divide by $Ae^{j\alpha n}$ we get

$$1 = Ae^{j\alpha N}$$

The above equation is satisfied if

$$\alpha N = 2\pi k$$

or

$$\frac{N}{k} = \frac{2\pi}{\alpha}$$

Again, and similar to what we did in Section 1.8, if $2\pi/\alpha$ is a rational number then $x(n)$ is periodic with period

$$N = \left(\frac{2\pi}{\alpha}\right)k$$

and the smallest N satisfying the above equation is called the fundamental period. If $2\pi/\alpha$ is not rational, then $x(n)$ is not periodic.

Example 1.4

Consider the following complex sinusoidal discrete signals.

1. $2e^{jn}$
2. $e^{jn\pi}$
3. $e^{(j2\pi n+2)}$
4. $e^{jn8\pi/3}$

Are the signals periodic? If so, what are their periods?

Solution

For the first signal, $jn = j\alpha n$ requires that $\alpha = 1$. For periodicity, the ratio $2\pi/\alpha$ must be a rational number. But $2\pi/1$ is not a rational number and this signal is not periodic.

For the second signal, $jn\pi = j\alpha n$ requires $\alpha = \pi$. For periodicity again, $2\pi/\alpha$ must be a rational number. The ratio $2\pi/\alpha = 2\pi/\pi = 2/1$ is a rational number. Thus the signal is periodic.

$$\frac{2\pi}{\alpha} = \frac{N}{k} = \frac{2\pi}{\pi} = \frac{2}{1}$$

For $k = 1$ we get $N = 2$. Thus N is the fundamental period.

For the third signal, $e^{(j2\pi n+)2}$ can be written as $e^2 e^{j2\pi n}$ and $2\pi jn = j\alpha n$ requires $\alpha = 2\pi$. For periodicity, $2\pi/\alpha = 2\pi/2\pi = 1/1$ must be a rational number which is true in this case. Therefore, the signal is periodic.

$$\frac{2\pi}{\alpha} = \frac{2\pi}{2\pi} = \frac{1}{1} = \frac{N}{k}$$

For $k = 1$ we get $N = 1$, which is the fundamental period.

For the fourth signal, $j\alpha n = j\,\frac{8\pi}{3}$ and $\alpha = \frac{8\pi}{3}$. For periodicity, $2\pi/\alpha$ must be a rational number.

$$\frac{2\pi}{\alpha} = \frac{2\pi}{8\pi/3} = \frac{6}{8} = \frac{3}{4}$$

This is a rational number and the signal is periodic with the fundamental period N calculated by setting

$$\frac{2\pi}{\alpha} = \frac{3}{4} = \frac{N}{k}$$

For $k = 4$ we get $N = 3$ as the smallest integer.

1.11 The Shifting Operation

A shifted discrete signal $x(n)$ is $x(n - k)$ where k is an integer. If k is positive, then $x(n)$ is shifted k units to the right and if k is negative, then $x(n)$ is shifted k units to the left.

Consider the discrete impulse signal $x(n) = 5A\delta(n)$. The signal $x(n - 1) = 5\delta(n - 1)$ is the signal $x(n)$ shifted by 1 unit to the right. Also $x(n + 3) = 5\delta(n + 3)$ is $x(n)$ shifted 3 units to the left. The importance of the shift operation

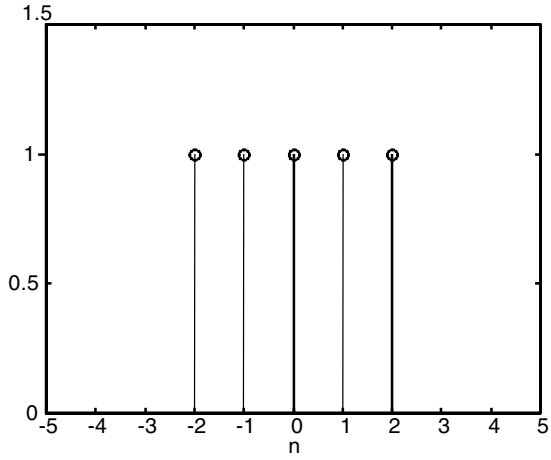

FIGURE 1.15 Pulse signal for Example 1.5.

will be apparent when we get to the next chapters. In the following section we will see one basic importance.

Example 1.5

Consider the discrete pulse in Figure 1.15. Write this pulse as a sum of discrete step signals.

Solution

Consider the shifted step signal in Figure 1.16 and the other shifted step signal in Figure 1.17. Let us subtract Figure 1.16 from Figure 1.17 to get Figure 1.15.

1.12 Representing a Discrete Signal Using Impulses

Any discrete signal can be represented as a sum of shifted impulses. Consider the signal in Figure 1.18 that has values at $n = 0$, 1, 2, and 3.

Each of these values can be thought of as an impulse shifted by some units. The signal at $n = 0$ can be represented as $1\,\delta(n)$, the signal at $n = 1$ as $1.5\delta(n-1)$, the signal at $n = 2$ as $1/2\delta(n-2)$, and the signal at $n = 3$ as $1/4\delta(n-3)$. Therefore, $x(n)$ in Figure 1.18 can be represented mathematically as

$$x(n) = \delta(n) + 1.5\delta(n-1) + \frac{1}{2}\delta(n-2) + \frac{1}{4}\delta(n-3)$$

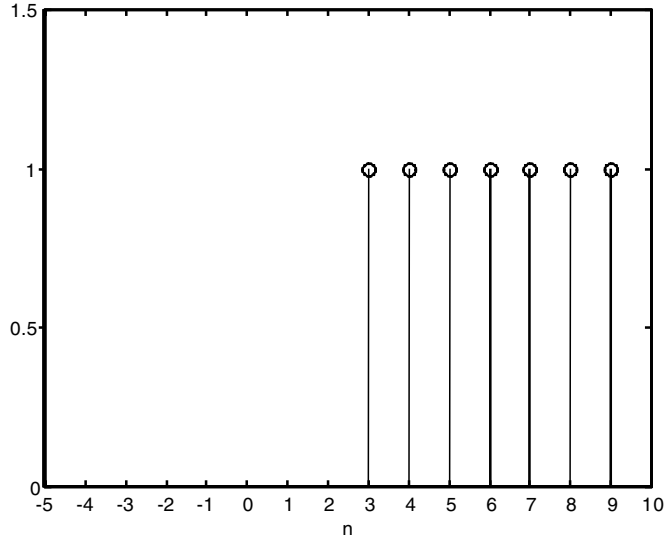

FIGURE 1.16 Signal for Example 1.5.

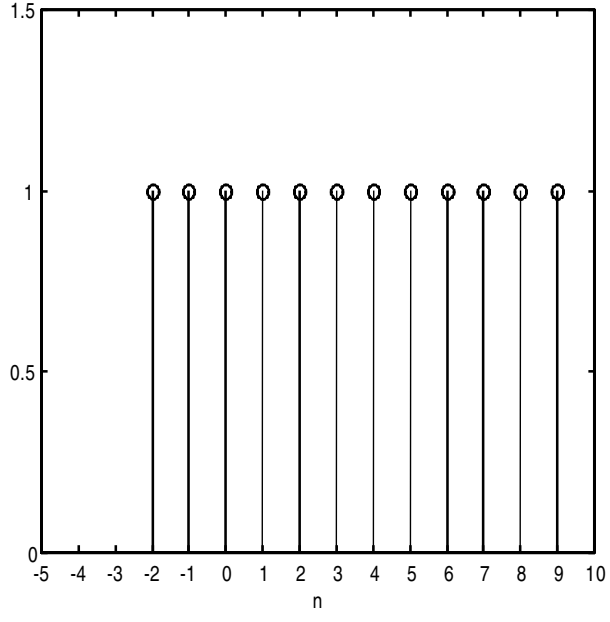

FIGURE 1.17 Signal for Example 1.5.

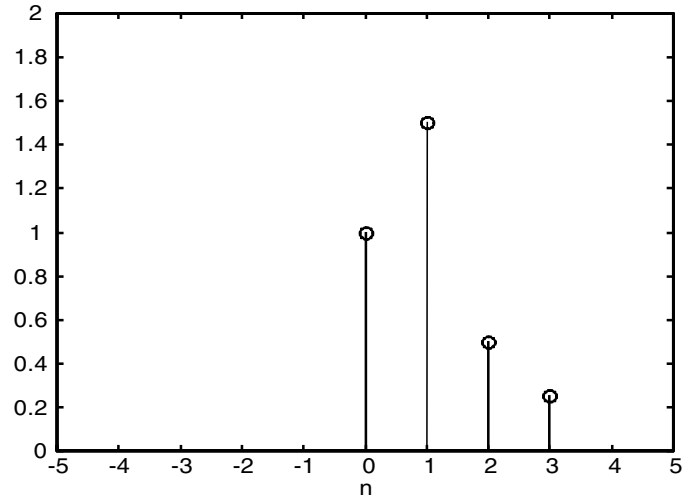

FIGURE 1.18 Representation of a discrete signal using impulses.

Example 1.6

Represent the following discrete signals using impulse signals.

1. $x(n) = \{1, 2, \underset{\uparrow}{3}, 4, -1\}$
2. $x(n) = \{\underset{\uparrow}{0}, 1, 2, -4\}$

where the arrow under the number indicates $n = 0$, where n is the time index where the signal starts.

Solution

Graphically, the first signal is shown in Figure 1.19, and it can be seen as a sum of impulses as

$$x(n) = 1\delta(n+2) + 2\delta(n+1) + 3\delta(n) + 4\delta(n-1) - 1\delta(n-2)$$

The second signal is shown in Figure 1.20 and can be written as the sum of impulses as

$$x(n) = 0\delta(n) + 1\delta(n-1) + 2\delta(n-2) - 4\delta(n-3)$$

1.13 The Reflection Operation

Mathematically, a reflected signal $x(n)$, is written as $x(-n)$ as shown in Figure 1.21.

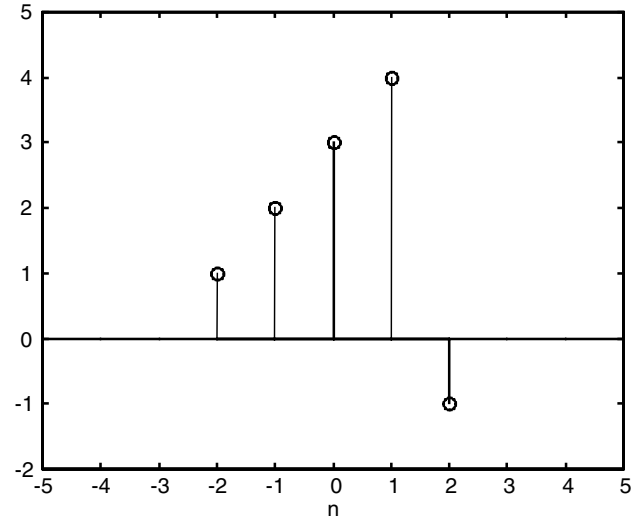

FIGURE 1.19 Signal for Example 1.6.

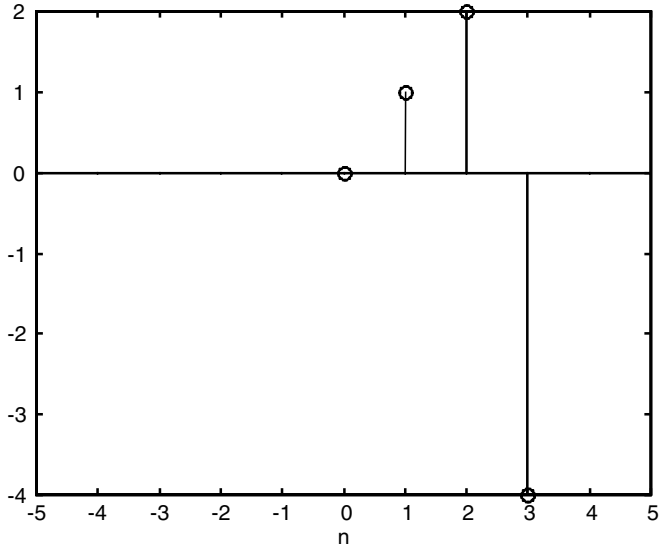

FIGURE 1.20 Signal for Example 1.6.

1.14 Time Scaling

A time-scaled discrete signal $x(an)$ of $x(n)$ is calculated by considering two cases for a. In case $a = k$, we will consider all cases where k is an integer. In

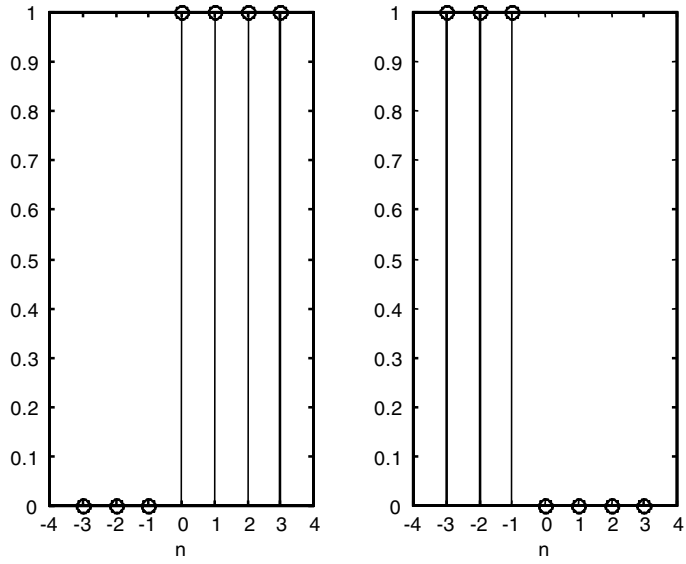

FIGURE 1.21 The reflected signal.

case $a = 1/k$ we will also consider all values for k where k is an integer. An example will be given shortly.

1.15 Amplitude Scaling

An amplitude scaled version of $x(n)$ is $ax(n)$ where, if a is negative, the magnitude of each sample in $x(n)$ is reversed and scaled by the absolute value of a. If a is positive, then each sample of $x(n)$ is scaled by a.

Example 1.7

Consider the following signal

$$x(n) = \left\{-1, 2, \underset{\uparrow}{0}, 3\right\}$$

Find

1. $x(-n)$
2. $x(-n + 1)$
3. $2x(-n + 1)$
4. $x(-n) + x(-n + 1)$

Solution

1. $x(n) = \{-1, 2, \underset{\uparrow}{0}, 3\}$
   $x(-n)$ is $x(n)$ shifted about the zero position which is
   $x(-n) = \{3, \underset{\uparrow}{0}, 2, -1\}$
2. $x(-n + 1)$ is $x(-n)$ shifted to the left 1 unit and it is
   $x(-n+1) = \{3, 0, \underset{\uparrow}{2}, -1\}$
3. $2x(-n + 1)$ is $x(-n + 1)$ scaled by 2 and is
   $2x(-n+1) = \{6, 0, \underset{\uparrow}{4}, -2\}$
4. $x(-n) + x(-n + 1) = \{3, \underset{\uparrow}{0}, 2, -1\} + \{3, 0, \underset{\uparrow}{2}, -1\}$

We add these two signals at the $n = 0$ index to get

$$x(-n)+x(-n+1)=\left\{0, 3, \underset{\uparrow}{0}, 2, -1\right\}+\left\{3, 0, \underset{\uparrow}{2}, -1, 0\right\}=\left\{3, 3, \underset{\uparrow}{2}, 1, -1\right\}$$

1.16 Even and Odd Discrete Signal

Every discrete signal $x(n)$ can be represented as a sum of an odd and an even discrete signal. The discrete signal $x(n)$ is odd if

$$x(n) = -x(-n) \tag{1.10}$$

The discrete signal $x(n)$ is even if

$$x(n) = x(-n) \tag{1.11}$$

Therefore, any discrete signal, $x(n)$, can be written as the sum of an even and an odd discrete signal. We write

$$x(n) = x_{even}(n) + x_{odd}(n)$$

where

$$x_{even}(n) = \frac{1}{2}\left(x(n) + x(-n)\right) \tag{1.12}$$

and

$$x_{odd}(n) = \frac{1}{2}\left(x(n) - x(-n)\right) \tag{1.13}$$

Example 1.8

The signals in Figures 1.22 and 1.23 are not odd or even. Write them as the sum of even and odd signals.

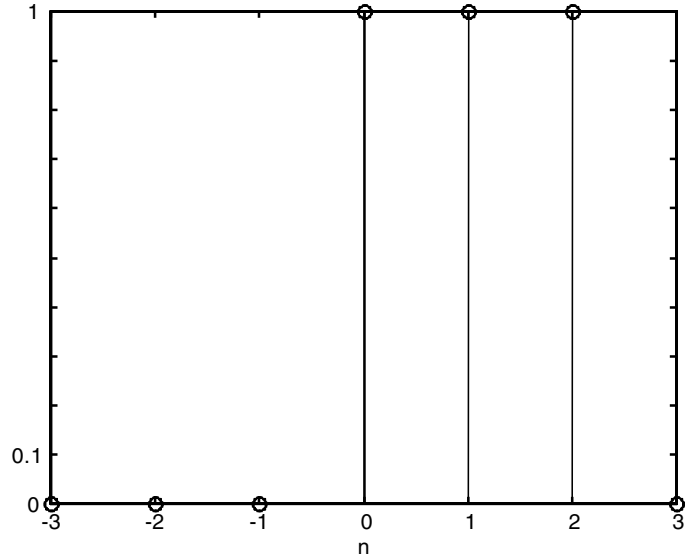

FIGURE 1.22 Signal for Example 1.8.

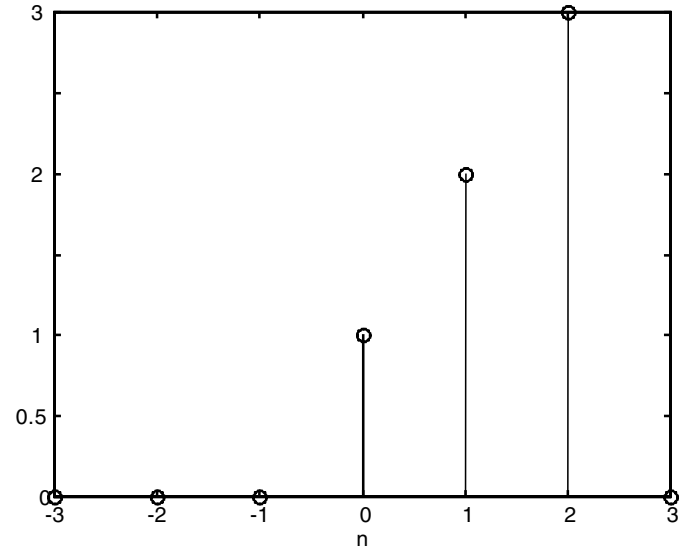

FIGURE 1.23 Signal for Example 1.8.

Solution

For $x(n) = \{\underset{\uparrow}{1}, 1, 1\}$ we have

$$x_{odd}(n) = \frac{1}{2}[x(n) - x(-n)] = \frac{1}{2}\left[\left\{\underset{\uparrow}{1}, 1, 1\right\} - \left\{1, 1, \underset{\uparrow}{1}\right\}\right] = \left\{-1/2, -1/2, \underset{\uparrow}{0}, 1/2, 1/2\right\}$$

$$x_{even}(n) = \frac{1}{2}[x(n) + x(-n)] = \frac{1}{2}\left[\left\{\underset{\uparrow}{1}, 1, 1\right\} + \left\{1, 1, \underset{\uparrow}{1}\right\}\right] = \left\{1/2, 1/2, \underset{\uparrow}{1}, 1/2, 1/2\right\}$$

Now if we add x_{odd} and x_{even} we will get

$$\left\{-1/2, -1/2, \underset{\uparrow}{0}, 1/2, 1/2\right\} + \left\{1/2, 1/2, \underset{\uparrow}{1}, 1/2, 1/2\right\} = \left\{0, 0, \underset{\uparrow}{1}, 1, 1\right\}$$

For $x(n) = \{\underset{\uparrow}{1}, 2, 3\}$ we have

$$x_{odd}(n) = \frac{1}{2}[x(n) - x(-n)] = \frac{1}{2}\left[\left\{\underset{\uparrow}{1}, 2, 3\right\} - \left\{3, 2, \underset{\uparrow}{1}\right\}\right] = \left\{-3/2, -1, \underset{\uparrow}{0}, 1, 3/2\right\}$$

$$x_{even}(n) = \frac{1}{2}[x(n) + x(-n)] = \frac{1}{2}\left[\left\{\underset{\uparrow}{1}, 2, 3\right\} + \left\{3, 2, \underset{\uparrow}{1}\right\}\right] = \left\{3/2, 1, \underset{\uparrow}{0}, 1, 3/2\right\}$$

Now if we add x_{odd} and x_{even} we will get

$$\{-3/2, -1, 0, 1, 3/2\} + \{3/2, 1, 0, 3/2\} = \{0, 0, \underset{\uparrow}{0}, 2, 3\}$$

1.17 Does a Discrete Signal Have a Time Constant?

Consider the continuous real exponential signal

$$x(t) = e^{-\alpha t}$$

Let us sample $x(t)$ every $t = nT_s$ sec with $\alpha > 0$. Then we write

$$x(n) = e^{-\alpha n T_s} = \left(e^{-\alpha T_s}\right)^n = a^n$$

where $a = e^{-\alpha T_s}$ is a real number. The continuous exponential signal has a time constant τ where $e^{-\alpha t} = e^{-t/\tau}$. This implies that $\tau = 1/\alpha$. Therefore

$$x(n) = \left(e^{-\frac{T_s}{\tau}}\right)^n = a^n$$

But $e^{-T_s/\tau} = a$ and by taking the logarithm on both sides, we get

$$\frac{\tau}{T_s} = -\frac{1}{\ln a} \text{ or } \tau = \frac{-T_s}{\ln a}$$

So, a discrete time signal that was obtained by sampling a continuous exponential signal has a time constant as given above.

Example 1.9

Consider the following two continuous signals

$$x(t) = e^{-t}$$

$$x(t) = \mathrm{e}^{-100t}$$

1. Find the time constant for both continuous signals.
2. Discretize both signals for $T_s = 1$ sec and then find the time constant for the resulting discrete signals.

Solution

For $x(t) = \mathrm{e}^{-t}$ and by setting $\mathrm{e}^{-t} = e^{-t/\tau}$ in order to find the time constant τ we see that $\tau = 1$ sec. By letting $t = nT_s$ we get the discretized signal

$$x(n) = \mathrm{e}^{-(nT_s)} = \mathrm{e}^{-n} = \left(e^{-1}\right)^n$$

As discussed earlier, we set

$$\mathrm{e}^{-1} = e^{-\frac{T_s}{\tau}}$$

to get the discrete time constant

$$\tau = \frac{-T_s}{\ln\left(e^{-1}\right)} = \frac{-1}{\ln\left(e^{-1}\right)}$$

For $x(t) = e^{-100t}$, we set

$$\mathrm{e}^{-100t} = e^{-\frac{T_s}{\tau}}$$

to get $\tau = \frac{1}{100}$ sec. Similarly, the second discretized signal is

$$x(n) = \mathrm{e}^{-100T_s n} = \left(e^{-100}\right)^n$$

and the discrete time constant is

$$\tau = \frac{-T_s}{\ln\left(e^{-100}\right)} = \frac{-1}{\ln\left(e^{-100}\right)}$$

1.18 Basic Operations on Discrete Signals

A discrete system will operate on discrete inputs. Some basic operations follow. The output of a discrete system is the input operated upon in a certain way.

1.18.1 Modulation

Consider the two discrete signals $x_1(n)$ and $x_2(n)$. The resulting signal $y(n)$ where

$$y(n) = x_1(n)x_2(n)$$

is called the modulation of $x_1(n)$ and $x_2(n)$ and where $y(n)$ in a discrete signal found by multiplying the sample values of $x_1(n)$ and $x_2(n)$ at every instant.

1.18.2 Addition and Subtraction

Consider the two discrete signals $x_1(n)$ and $x_2(n)$. The addition/subtraction of these two signals is $y(n)$ where

$$y(n) = x_1(n) \pm x_2(n)$$

1.18.3 Scalar Multiplication

A scalar multiplication of the discrete signal $x(n)$ is the signal $y(n)$ where

$$y(n) = Ax(n)$$

where A is the scaling factor.

1.18.4 Combined Operations

We may have multiple operations among input discrete signals. One operation may be represented as

$$y(n) = 2x_1(n) + 1.5x_2(n-1)$$

where we have scaling, shifting and addition operations combined. In other operations you may have the output $y(n)$ presented as

$$y(n) = 2x_1(n) - x_2(n)x_3(n)$$

where you have $x_1(n)$ scaled, then the modulated signal resulting from $x_2(n)$ and $x_3(n)$ is subtracted from $2x_1(n)$.

Example 1.10

Consider the discrete signal as shown in Figure 1.24. Find

1. $x(-n)$
2. $x(n + 2)$
3. $x(2n)$
4. $x(n/2)$
5. $x(2n - 1)$
6. $x(n)x(n)$

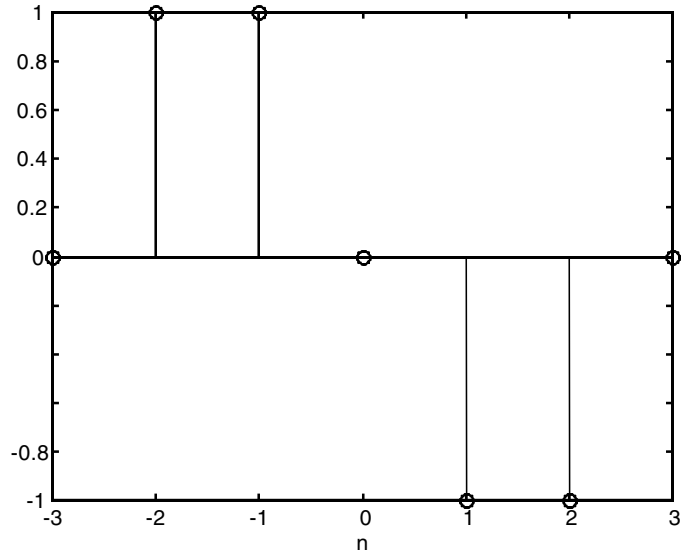

FIGURE 1.24 Signal for Example 1.10.

Solution

From Figure 1.24 we see that

$$x(n) = \left\{1,\ 1,\ \underset{\uparrow}{0},\ -1,\ -1\right\}$$

1. $x(-n)$ is $x(n)$ reflected and is

$$x(-n) = \left\{-1, -1,\ \underset{\uparrow}{0},\ 1,\ 1\right\}$$

2. $x(n + 2)$ is $x(n)$ shifted to the left by 2 units and is

$$x(n+2) = \left\{1,\ 1,\ 1,\ 0,\ -1,\ \underset{\uparrow}{-1}\right\}$$

3. The new time axis in this case is $n_{new} = n_{old}/2$ where n_{old} is the index of the given $x(n)$. The indices are arranged as in the following with the help of the equation $n_{new} = n_{old}/2$.

n_{old}	n_{new}
–2	–1
–1	–1/2 No value will appear at $n = -1/2$ since n must be an integer
0	0
1	1/2 No value will appear
2	1
3	No value for $x(n)$ at $n = 3$ so we stop at this point

Therefore, the new n axis for $x(2n)$ will start at –1 and will contain the indices $n = -1$, 0 and 1. The value at –1 for the scaled signal $x(2n)$ will be the value of $x(n)$ at $n = -2$. Similarly, the value at $n_{new} = 0$ will be the value at $n_{old} = 0$, and the value at $n_{new} = 1$ will be the value at $n_{old} = 2$. Therefore, $x(2n)$ is

$$x(2n) = \left\{0,\ 1,\ \underset{\uparrow}{0},\ -1,\ 0\right\}$$

4. This is also a scaling case. For this case, the new time axis is

$$n_{new} = 2n_{old}$$

These indices are arranged in the following table.

n_{old}	n_{new}
–2	–4
–1	–2
0	0
1	2
2	4

The scaled signal $x(\frac{n}{2})$ therefore is

$$x(n/2) = \left\{1,\ 0,\ 1,\ 0,\ \underset{\uparrow}{0},\ 0,\ -1,\ 0,\ -1\right\}$$

5. $x(2n+1)$ is $x(2n)$ shifted left by 1 unit and is

$$x(2n+1) = \left\{0,\ 1,\ 0,\ \underset{\uparrow}{-1},\ 0\right\}$$

6. $x(n)$ multiplied by $x(n)$ is called the element-by-element multiplication and is

$$x(n)x(n) = \left\{1,\ 1,\ \underset{\uparrow}{0},\ -1,\ -1\right\}\left\{1,\ 1,\ \underset{\uparrow}{0},\ -1,\ -1\right\} = \left\{1,\ 1,\ \underset{\uparrow}{0},\ 1,\ 1\right\}$$

1.19 Energy and Power Discrete Signals

The total energy in the discrete signal $x(n)$ is $E(n)$ and mathematically written as

$$E(n) = \sum_{-\infty}^{+\infty} |x(n)|^2 \tag{1.14}$$

The average power in the discrete signal $x(n)$ is $P(n)$ and is written mathematically as

$$P(n) = \lim_{k\to\infty} \frac{1}{2k+1} \sum_{n=-k}^{k} |x(n)|^2 \tag{1.15}$$

where

$$\sum_{n=-k}^{n=k} |x(n)|^2$$

is the energy in $x(n)$ in the interval $-k < n < k$. The average power of a discrete periodic signal is written mathematically as

$$P(n) = \frac{1}{N} \sum_{n=0}^{N-1} |x(n)|^2 \tag{1.16}$$

where N is the period of $x(n)$.

Example 1.11

Consider the following finite discrete signals

1. $x(n) = -1\delta(n-0) + 2\delta(n-1) - 2\delta(n-2)$
2. $x(n) = \{1, \underset{\uparrow}{0}, -1\}$

Find the energy in both signals.

Solution

The first signal $x(n)$ can be written as

$$x(n) = \left\{\underset{\uparrow}{1},\ 2, -2\right\}$$

The energy in the signal is then

$$E(n) = \sum_{n=0}^{2} |x(n)|^2 = (-1)^2 + (2)^2 + (-2)^2 = 1 + 4 + 4 = 9$$

This means that $x(n)$ has finite energy.

For the second signal the total energy is given by

$$E(n) = \sum_{n=-1}^{1} |x(n)|^2 = (1)^2 + (0)^2 + (-1)^2 = 2$$

Example 1.12

Consider the discrete signals

$$x(n) = 2(-1)^n \qquad n \geq 0$$

Find the energy and the power in $x(n)$

Solution

$$E(n) = \sum_{n=0}^{\infty} \left|2(-1)^n\right|^2 = 4[1 + 1 + \cdots]$$

This sum clearly does not converge to a real number. Hence $x(n)$ has infinite energy. The power in the signal is

$$P(n) = \lim_{k\to\infty} \frac{1}{2k+1}\left(4\sum_{n=0}^{k} 1\right) = \lim_{k\to\infty} \frac{4(k+1)}{2k+1} = 2$$

Therefore, $x(n)$ has finite power and is a power signal.

Example 1.13

Find the energy in the signal:

$$x(n) = \frac{1}{n} \qquad n \geq 1$$

Solution

The energy is given by

$$E(n) = \sum_{n=1}^{\infty} \left|\tfrac{1}{n}\right|^2 = 1 + \left(\tfrac{1}{2}\right)^2 + \left(\tfrac{1}{3}\right)^2 + \ldots$$

which converges to $\pi^2/6$. This means that $x(n)$ has finite energy.

1.20 Bounded and Unbounded Discrete Signals

A discrete signal $x(n)$ is bounded if each sample in the signal has a bounded magnitude. Mathematically, if $x(n)$ is bounded then

$$\left|x(n)\right| \leq \beta < \infty$$

where β is some positive value. If this is not the case then $x(n)$ is said to be unbounded. The step signal is bounded, the ramp is unbounded and the sinusoidal signal is bounded.

1.21 Some Insights: Signals in the Real World

The signals that we have introduced in this chapter were all represented in mathematical form and plotted on graphs. This is how we represent signals for the purpose of analysis, synthesis and design. For better understanding

of these signals we will provide herein real-life situations and see the relation between these mathematical abstractions of signals and get a feel of what they may represent in real life.

1.21.1 The Step Signal

In real-life situations this signal can be viewed as a constant force of magnitude A Newtons applied at time (t) = 0 sec to a certain object for a long time. In another situation, $Au(t)$ can be an applied voltage of constant magnitude to a load resistor R at the time $t = 0$.

1.21.2 The Impulse Signal

Again, in real life, this signal can represent a situation in which a person hits an object with a hammer with a force of A Newtons for a very short period of time (picoseconds). We sometimes refer to this kind of signal as a shock.

In another real-life situation, the impulse signal can be as simple as closing and opening an electrical switch in a very short time. Another situation where a spring-carrying mass is hit upward can be seen as an impulsive force. A sudden oil spill similar to the one that happened during the Gulf war can represent a sudden flow of oil. You may realize that it is impossible to generate a pure impulse signal for zero duration and infinite magnitude. To create an approximation to an impulse we can generate a pulse signal of very short duration where the duration of the pulse signal is very short compared with the response of the system.

1.21.3 The Sinusoidal Signal

This signal can be thought of as a situation where a person is shaking an object regularly. This is like pushing and pulling an object continuously with a period of T sec. Thus, a push and pull forms a complete period of shaking. The distance the object covers during this shaking represents a sinusoidal signal. In the case of electrical signals, an AC voltage source is a sinusoidal signal.

1.21.4 The Ramp Signal

In real-life situations this signal can be viewed as a signal that is increasing linearly with time. An example is when a person applies a force at time $t = 0$ to an object and keeps pressing the object with increasing force for a long time. The rate of the increase in the force applied is constant.

Consider another situation where a radar antenna, an anti-aircraft gun and an incoming jet are in one place. The radar antenna can provide an angular position input. In one case the jet motion forces this angle to change uniformly

with time. This will force a ramp input signal to the anti-aircraft gun since it will have to track the jet.

1.21.5 Other Signals

A train of impulses can be thought of as hitting an object with a hammer continuously and uniformly. In terms of electricity, you may be closing and opening a switch continuously. A rectangular pulse can be likened to applying a constant force to an object for a certain time and instantaneously removing that force. It is also like applying a constant voltage for a certain time and then instantaneously closing the switch of the voltage source. Other signals are the random signals where the magnitude changes randomly as time progresses. These signals can be thought of as shaking an object with variable random force as time progresses, or as a gusting wind.

1.22 End of Chapter Examples

EOCE 1.1

Consider the following discrete periodic and nonperiodic signals in Figure 1.25. For the periodic signal find the period N and also the average power. For the nonperiodic signals, find the total energy.

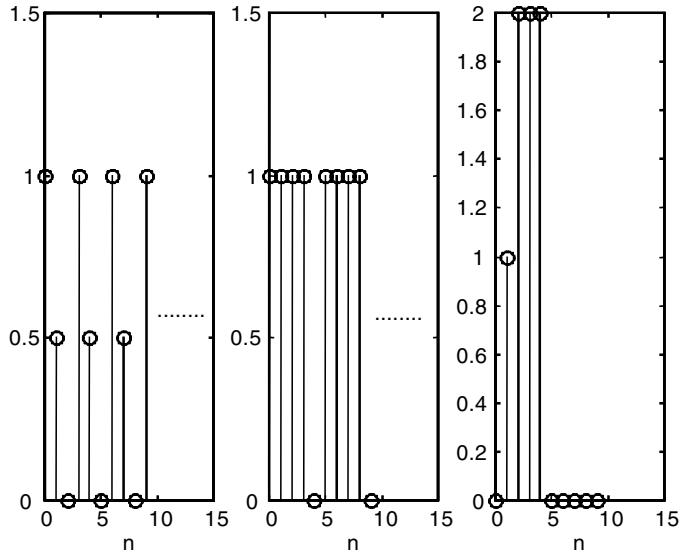

FIGURE 1.25 Signals for EOCE 1.1.

Solution

For the first discrete signal in Figure 1.25, the signal $x(n)$ is periodic with period $N = 3$. The average power is then

$$P(n) = \frac{1}{N}\sum_{n=0}^{N-1}|x(n)|^2 = \tfrac{1}{3}\sum_{n=0}^{2}|x(n)|^2 = \tfrac{1}{3}\left((1)^2 + \left(\tfrac{1}{2}\right)^2 + (0)^2\right)$$

$$P(n) = \tfrac{1}{3}\left(1 + \tfrac{1}{4}\right) = \tfrac{1}{3}\left(\tfrac{4+1}{4}\right) = \tfrac{5}{12}$$

We can use MATLAB to calculate this average power as in the following script.

```
%calculating average power for x(n)
%The signal
x = [1 0.5 0]
N = 3; % the period
% implementing the equation for power
pp = sum(abs(x).^2);
p = pp/N; % the average power
```

The result will be 0.4167.

For the second discrete signal the period is $N = 4$. The average power is calculated as

$$P(n) = \tfrac{1}{N}\sum_{n=\infty}^{N-1}|x(n)|^2 = \tfrac{1}{4}\sum_{n=0}^{3}|x(n)|^2 = \tfrac{1}{4}\left(\left(x(0)\right)^2 + \left(x(1)\right)^2 + \left(x(2)\right)^2 + \left(x(3)\right)^2\right)$$

$$P(n) = \tfrac{1}{5}\left((1)^2 + (1)^2 + (1)^2 + (1)^2 + (0)^2\right) = \tfrac{1}{5}[4] = 0.8$$

We can use MATLAB and write the following script to find the power.

```
%calculating average power for x(n)
%The signal
x = [1 1 1 1 0]
N = 5; % the period
% implementing the equation for power
pp = sum(abs(x).^2);
p = pp/N; % the average power
```

The result is 0.8 for the average power.

The last signal in Figure 1.25 is not periodic and the total energy is

$$E(n) = \sum_{n=0}^{+\infty}|x(n)|^2 = \sum_{n=4}^{n=4}|x(n)|^2 = x(0)^2 + x(1)^2 + x(2)^2 + x(3)^2 + x(4)^2$$

$$E(n) = (0)^2 + (1)^2 + (2)^2 + (2)^2 = 1 + 4 + 4 + 4 = 13$$

We can use MATLAB to find this total energy in the signal by writing the following script.

```
% Calculating the energy in the signal
% Defining the signal
x= [0 1 2 2 2];
E = sum(abs(x).^2) %the energy equation
```

The result will be 13 for the energy.

EOCE 1.2

Write MATLAB scripts to simulate the step and the impulse signals.

Solution

For the step signal

$$u(n)=\begin{cases}1 & n\geq 0\\ 0 & n<0\end{cases}$$

We can use the MATLAB function `ones` to generate sequences that have every value unity. We write `ones(1,L)` where L is the number of ones in this row vector that `ones` generate. However sometimes not all step signals start at $n = 0$. In this case we have

$$u(n-n_0)=\begin{cases}1 & n\geq n_0\\ 0 & n<0\end{cases}$$

We know that $u(n)$ is defined for $n \geq 0$ and we also know that we cannot generate $u(n)$ for an infinite number of samples. Therefore we will generate these sequences for a limited interval. We will denote n_1 to be the left limit and n_2 to be the right limit. Notice that `ones(1,L)` will generate `L` ones for $n = 0, 1, 2, 3, \ldots$ `L` $- 1$ automatically. To generate the most general step sequence $u(n - n_0)$ in the interval $n_1 \leq n \leq n_2$ we write the following MATLAB script that will generate a unit step signal that starts at $n_0 = -1$ and defined in the interval $-3 \leq n \leq 3$ first.

```
% entering the starting point and the limits
n0 = -1; n1 = -3; n2 = 3;
n = [n1: n2]; % generating the index n
% the following will generate the step signal desired
x=[(n- n0)>=0];
stem(n,x);
```

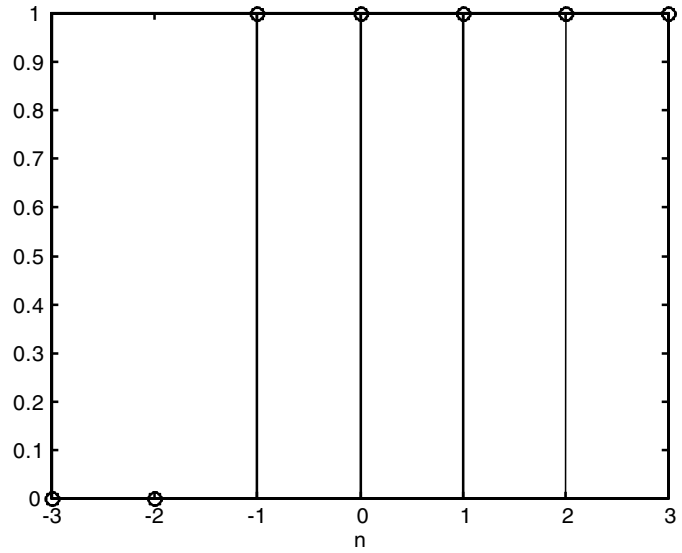

FIGURE 1.26 MATLAB-generated step signal starting at n = -1.

Note in the above script on the line before the last that for $n = -3$, $n_0 = -1$ (fixed), the logical expression $(-3 - (-1)) \geq 0$ results in $-3 + 1 \geq 0$ and evaluates to false, which in MATLAB evaluates to zero. Therefore for $n = -3$, $u(n + 1)$ is zero.

Now take $n = -2$; the expression $-2 + 1 \geq 0$ evaluates to false again. Take $n = -1$. The expression $-1 + 1 \geq 0$ evaluates to true and hence the first one appears. The plot is shown in Figure 1.26. To sketch $3u(n - 5)$ in the interval $-10 \leq n \leq 10$ we write the script

```
% entering the starting point and the limits
n0 = 5; n1 = -10; n2 = 10;
n = [n1: n2]; % generating the index n
% the following will generate the step signal desired
x=[(n- n0)>=0];
stem(n,x); xlabel('n');ylabel('Step at n = 5');
```

and the plot is shown in Figure 1.27.

For the impulse signal we have

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

We can use the MATLAB function zeros to generate a sequence of zeros of any length. For example, the command

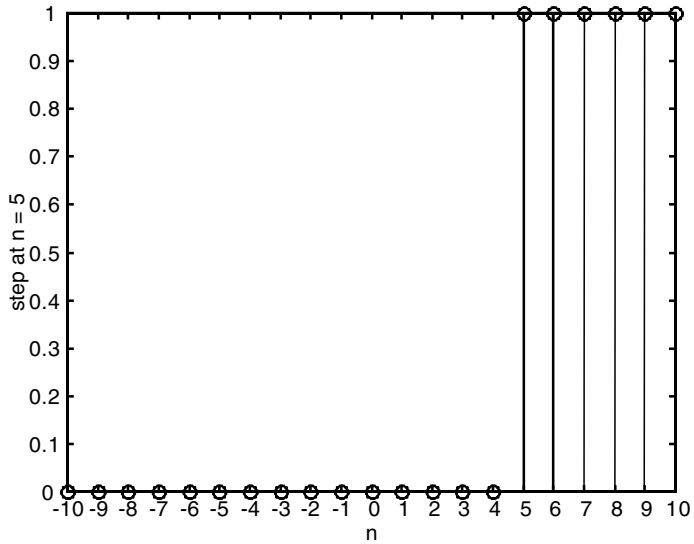

FIGURE 1.27 MATLAB-generated step signal starting at $n = 5$.

```
delta = zeros(1,L)
```

will generate a sequence of zeros of length `L`, Then we can make the first value in the sequence one to form the impulse signal and write

```
delta(0) = 1;
```

But again, suppose you want to generate an impulse sequence that has a certain value at $n = n_0$ and zero otherwise. We write

$$\delta(n-n_0)=\begin{cases}1 & n=n_0\\ 0 & n\neq n_0\end{cases}$$

For this we write the following MATLAB script that simulates $\delta(n-1)$.

```
%index where you want the impulse signal to have a value
n0=1;
%we also need a fixed interval for the signal
n1= -5;
n2= 5;
n = [n1 :n2]; %forming the index n
%this will generate the impulse at the specified index
x = [(n-n0)==0];
stem(n,x);
```

The plot is shown in Figure 1.28. Note the last line of the above MATLAB script

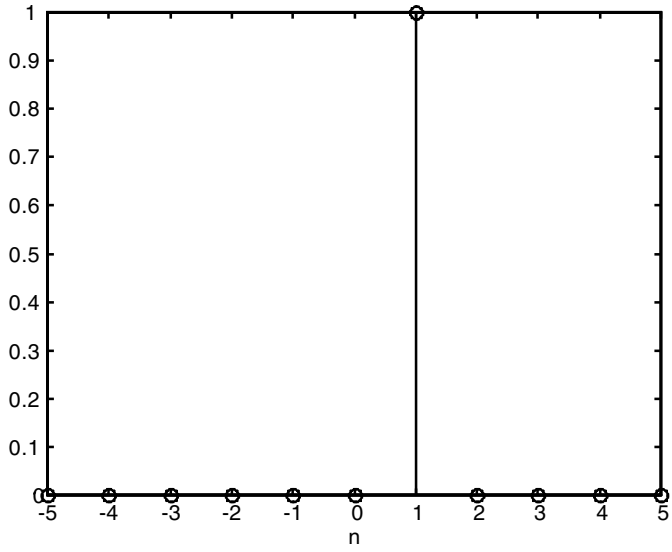

FIGURE 1.28 MATLAB-generated impulse signal at $n = 1$.

$$x = \left[\left(n - n_0\right) == 0\right]$$

If $n = -5$, $(-5 - 1 == 0)$ evaluates to false and the impulse at $n = -5$ is zero. If $n = 2$, $(2 - 1 == 0)$ also evaluates to zero and the impulse signal at $n = 2$ is zero. But if $n = 1$, $(1 - 1 == 0)$ evaluates to true and this is the only value in the interval $-5 \leq n \leq 5$ for the impulse to have a value other than zero.

EOCE 1.3

Write MATLAB scripts to simulate the exponential signal

$$x(n) = A(\alpha)^n$$

and the sinusoidal signal

$$x(n) = A\cos\left(\theta_0 n + \phi\right)$$

Solution

For

$$x(n) = A(\alpha)^n \quad -\infty < n < +\infty$$

we can generate the $x(n)$ sequence in a limited interval only. To simulate $3(.5)^n$ in the interval $-3 \leq n \leq 3$, we write the script

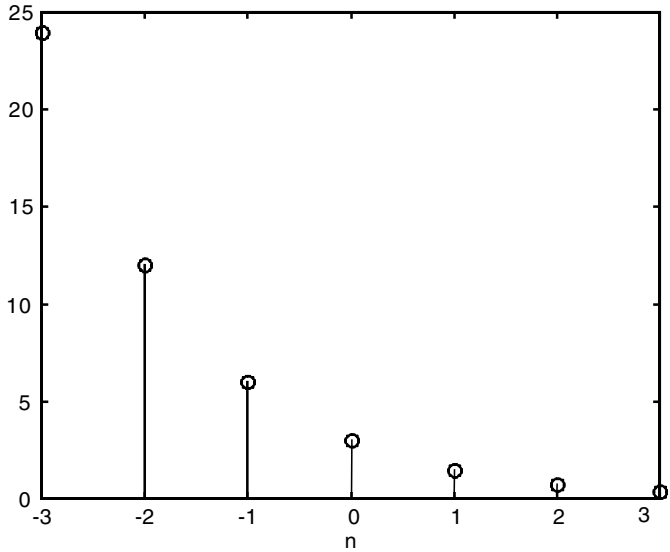

FIGURE 1.29 MATLAB-generated exponential decaying signal.

```
n1= -3;
n2= 3;
n = [n1: n2];
x = 3*(.5).^n;
stem(n,x);
% the .^ operator multiplies element-by-element
```

and the plot is shown in Figure 1.29. It is seen that this signal is bounded because $0 < \alpha < 1$.

For the sinusoid sequence

$$x(n) = A\cos(\theta_0 n + \varphi) \quad -\infty < n < +\infty$$

the signal can be simulated in a fixed interval. To do that let us look at the signal

$$x(n) = 3\cos(3\pi n + 5) \quad -10 < n < 10$$

and write the following script to simulate this signal.

```
n1 =-10;
n2 =11;
n = [n1: n2];
x = 3*cos(3*pi*n+5);
stem(n,x);
```

The plot is shown in Figure 1.30.

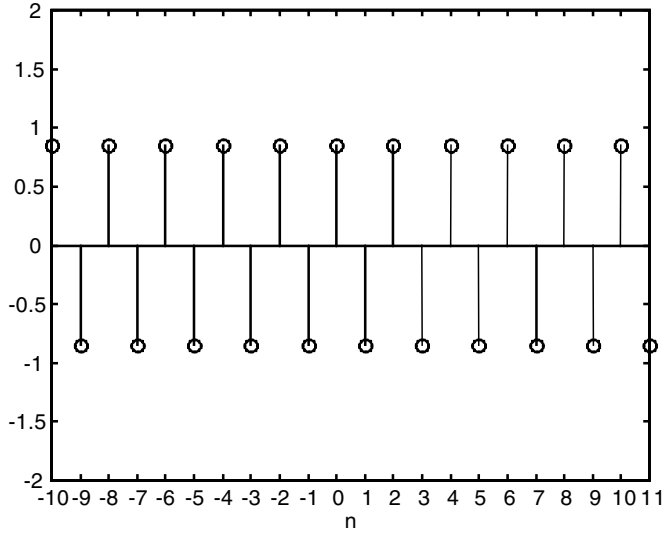

FIGURE 1.30 MATLAB-generated sinusoidal signal.

EOCE 1.4

Consider the following signals

$$x(n) = (.5)^n \cos(2n\pi + \pi)$$

$$x(n) = 5\cos(2n\pi + \pi) + 3$$

Are the signals periodic?

Solution

The first signal will decay to zeros as the index n get larger and it is not periodic. For the second signal, $\theta_0 = 2\pi$ and for periodicity, the ratio $2\pi/\theta_0$ must be rational. We have

$$\frac{2\pi}{\theta_0} = \frac{2\pi}{2\pi} = \frac{1}{1}$$

and therefore the signal is periodic. The period N is

$$N = k\left(\frac{2\pi}{\theta_0}\right)$$

For $k = 1$ we have $N = 1$. Note that the addition of 3 to $x(n)$ has no effect on the period N. We can use MATLAB to verify this and simulate $x(n)$ in the interval $-3 < n < 3$ and write the script.

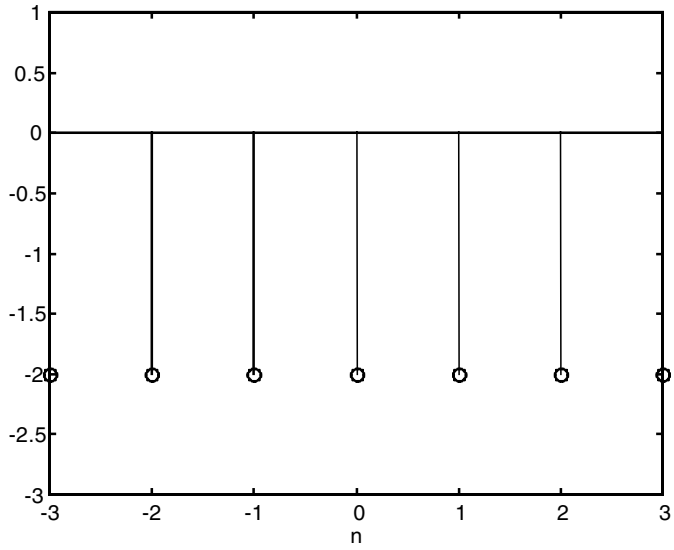

FIGURE 1.31 Signal for EOCE 1.4.

```
n1 = -3;
n2 = 3;
n = [n1: n2];
x = 5 *cos(2*pi*n + pi)+3;
stem(n, x);
```

and the plot is seen in Figure 1.31. It is seen from the figure that $N = 1$.

EOCE 1.5

Consider the following discrete signals

$$x_1(n) = \left\{\underset{\uparrow}{0},\ 1,\ 2,\ 3\right\}$$

$$x_2(n) = \left\{0,\ \underset{\uparrow}{1},\ 2,\ 3\right\}$$

Find $x_1(n) + x_2(n)$ and $x_1(n)\, x_2(n)$ analytically and using MATLAB.

Solution

Analytically, we can arrange the two signals as below and then add the corresponding samples. Remember we add samples with similar indexes.

$$x_1(n) = \left\{0, \underset{\uparrow}{0}, 1, 2, 3\right\}$$

$$+$$

$$x_2(n) = \left\{0, \underset{\uparrow}{1}, 2, 3, 0\right\}$$

$$x(n) = \left\{0, \underset{\uparrow}{1}, 3, 5, 3\right\}$$

$x_1(n)$ has an index that starts at $n = 0$ and ends at $n = 3$, and $x_2(n)$ has an index that starts at $n = -1$ and ends at $n = 2$. Note also that $x(n)$ starts at $n = -1$ and ends at $n = 3$. The initial index, $n = -1$, for $x(n)$ is the minimum of the minimum of the starting index for $x_1(n)$ and $x_2(n)$.

The last index $n = 3$ for $x(n)$ is the maximum of the maximum of the last index for $x_1(n)$ and $x_2(n)$. This note will help us do the addition with MATLAB.

For $x_1(n)$ $x_2(n)$ we have the same arrangement

$$x(n) = \left\{0, \underset{\uparrow}{0}, 1, 2, 3\right\}\left\{0, \underset{\uparrow}{1}, 2, 3, 0\right\} = \left\{0, \underset{\uparrow}{0}, 2, 6, 0\right\}$$

Using MATLAB, let n_1 represent the time scale (index) for $x_1(n)$ and n_2 represent the time scale for $x_2(n)$. Then the index for $x(n) = x_1(n) + x_2(n)$ will start with the minimum of n_1 and n_2 and end with the maximum of n_1 and n_2. Notice also when we added the signals analytically, we made both of the same length and the same length as $x(n)$. In MATLAB, the function find works as follows: find(x) returns the indices of the vector x that are nonzero. With that notice we now write the script that adds the two signals.

```
n1 = [0 1 2 3]; % index for first signal
x1= [0 1 2 3];
n2 = [ -1 0 1 2]; %index for the second signal
x2= [0 1 2 3];% signals with same value but different indices
% the starting index of the sum
n = min (min(n1) , min(n2) ): max (max(n1) , max(n2) );
%initializing x1i to zeros with the new index
x1i = zeros(1, length(n)); x2i = x1i; %copying x1i into x2i
% now we fill x1i and x2i
x1i (find((n >= min(n1)) & (n <= max(n1)) == 1)) = x1;
x2i (find((n >= min(n2)) & (n<= max(n2)) == 1)) = x2;
x = x1i + x2i % the addition result
```

The output is similar to what we got earlier.

Using MATLAB, we can simulate the element-by-element multiplication using a script similar to the one we just wrote for the addition.

```
n1 = [ 0 1 2 3]; % index for first signal
x1= [0 1 2 3];
n2 = [ -1 0 1 2]; %index for the second signal
x2= [0 1 2 3];% signals with same values but different indices
% the starting index of the sum
n = min (min(n1) , min(n2) ): max (max(n1) , max(n2) );
%initializing x1i to zeros with the new index
x1i = zeros(1, length(n)); x2i = x1i; %copying x1i into x2i
% now we fill x1i and x2i
x1i (find((n >= min(n1)) & (n <= max(n1)) == 1)) = x1;
x2i (find((n >= min(n2)) & (n<= max(n2)) == 1)) = x2;
x = x1i.*x2i % the element-by-element multiplication
```

and the result is again similar to what we arrived at before.

EOCE 1.6

The scripts that we have generated can be put in the form of functions. A function in MATLAB receives parameters and sends back results. Once this function is written and saved it can be utilized as often as desired. The function is typed in the MATLAB editor and then saved and given a name similar to the name of the function that was written. Let us write functions to implement the step signal, the impulse signal, the reflection of a signal, the sum of two signals, the product of two signals and the shifting by n_0 of a discrete signal.

Solution

The general form of a MATLAB function is

$$\text{function}[\text{rv1 rv2 } \ldots \text{ rvn}] = \text{Function\_Name}(\text{pv1, pv2, } \ldots \text{ pvn})$$

where rv1 is the returned value one and pv1 is the passed value one. Function_Name is the name of the function which should be the same name given to the file when the function is saved.

For the step discrete signal, let us call the function stepsignal. We will pass to stepsignal the time when the signal should start. We will call this time Sindex. We will also pass to it the starting and the ending of the time interval which we will call Lindex and Rindex for left index and right index. The function will return the signal $x(n)$ and its index. The function is

```
function[xofn, index]= stepsignal(Sindex, Lindex, Rindex)
index = [Lindex : Rindex];
xofn = [(index - Sindex) >= 0];
```

For the impulse signal, let us call the function impulsesignal. We will pass to it the point of application of the impulse signal, Sindex, and the range,

`Lindex` and `Rindex`. The function will send to us the impulse signal $x(n)$ and its index. The function is

```
function[xofn, index]= impulsesignal(Sindex, Lindex, Rindex)
index = [Lindex:Rindex];
xofn = [(index – Sindex) == 0];
```

The reflection of the signal $x(n)$ is implemented using the MATLAB function `fliplr`. We will use `fliplr` to flip the sample values for $x(n)$ and to flip the time index. The function will be called `xreflected` and will receive the original $x(n)$ and the original index. It will give back the reflected $x(n)$ and the new index. The function is

```
function [xnew, nnew] = xreflected(xold, nold);
xnew = fliplr(xold);
nnew = -fliplr(nold);
```

The function to add two discrete signals $x_1(n)$ and $x_1(n)$ will be called `x1plusx2`. It will receive the original signals and their original indices and return the sum of the two signals and the index for the sum. This function is

```
function[x, n] = x1plusx2 ( x1orig, x2orig, n1orig, n2orig)
n = min(min(n1orig), min(n2orig)): max(max(n1orig), max(n2orig));
x1i = zeros(1, length(n));
x2i= x1i
x1i (find((n >= min(n1orig))&(n <= max(n1orig))== 1))= x1orig;
x2i ( find((n >= min(n2orig))&(n<= max(n2orig)) == 1))= x2orig;
x = x1i+ x2i;
```

The function to multiply $x_1(n)$ and $x_2(n)$ is similar to the `x1plusx2` function and is given next.

```
function[x, n] = x1timesx2 ( x1orig, x2orig, n1orig, n2orig)
n = min(min(n1orig), min(n2orig)): max(max(n1orig), max(n2orig));
x1i = zeros(1, length(n));
x2i= x1i
x1i (find((n >= min(n1orig))&(n <= max(n1orig))== 1))= x1orig;
x2i ( find((n >= min(n2orig))&(n<= max(n2orig)) == 1))= x2orig;
x = x1i.* x2i;
```

A shifted version of $x(n)$ is $x(n - n_0)$

$$x_{\text{shift}}(n) = x\left(n - n_0\right)$$

If $n - n_0 = m$ then $x_{\text{shift}}(m + n_0) = x(m)$. This indicates that the sample values are not affected but the index is changed by adding the index shift n_0. We will call the function `xshifted` and pass to it the original $x(n)$, the original index n_1 and the amount of shift n_0.

```
function[xnew, nnew] = xshifted (xold, nold, n0)
nnew = nold + n0;
xnew = xold;
```

EOCE 1.7

Find

1. $x(n) = u(n) - 3\delta(n - 1)$ $\quad -3 \le n \le 3$
2. $x(n) = 3u(n - 3) + \delta(n - 2) + u(-n)$ $\quad -3 \le n \le 3$

Solution

For the first $x(n)$, we have the two signals

$$u(n) = \left\{0, 0, 0, \underset{\uparrow}{1}, 1, 1, 1\right\} \text{ and } -3\delta(n-1) = \left\{0, 0, 0, \underset{\uparrow}{0}, -3, 0, 0\right\}$$

Thus $u(n) - 3\delta(n - 1)$ is

$$x(n) = \left\{0, 0, 0, \underset{\uparrow}{1}, -2, 1, 1\right\}$$

We can use MATLAB to generate $x(n)$ and to plot it as follows.

```
n = [-3 : 3]; % generate the time index
x = stepsignal(0, -3, 3) -3* impulsesignal(1, -3, 3);
stem(n, x)
```

and the plot is shown in Figure 1.32.

For the second signal we have

$$3u(n-3) = \left\{0, 0, 0, \underset{\uparrow}{0}, 0, 0, 3\right\}$$

$$\delta(n-2) = \left\{0, 0, 0, \underset{\uparrow}{0}, 0, 1, 0\right\}$$

$$u(-n) = \left\{1, 1, 1, \underset{\uparrow}{1}, 0, 0, 0\right\}$$

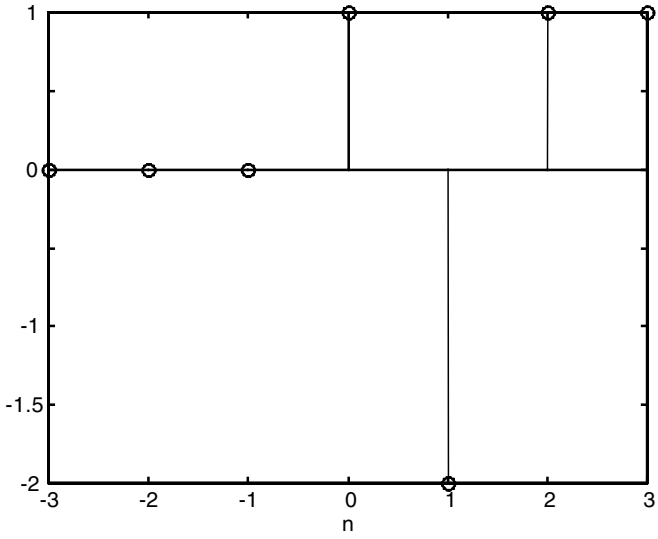

FIGURE 1.32 Signal for EOCE 1.7.

and the sum is

$$x(n) = \left\{1, 1, 1, \underset{\uparrow}{1}, 0, 1, 3\right\}$$

We can use MATLAB to find this result.

```
n= [-3 : 3];
x = 3 * stepsignal( 3, -3, 3) + impulsesignal(2, -3, 3)
     + xreflected (stepsignal(0, -3, 3), n);
```

The result is shown in Figure 1.33. Note in the above script that the function `xreflected` was called and one of the passed parameters is `stepsignal`(0, –3, 3), which is a function that will return the signal $u(n)$. Thus, $u(n)$ and n are passed to `xreflected`.

EOCE 1.8

Write a general-purpose script to find the odd and even parts of a discrete signal $x(n)$ defined on the interval $n_1 \le n \le n_2$.

Solution

Let us repeat the equations for the even and the odd part of $x(n)$

$$x_{even}(n) = \tfrac{1}{2}\left[x(n) + x(-n)\right]$$

$$x_{odd}(n) = \tfrac{1}{2}\left[x(n) - x(-n)\right]$$

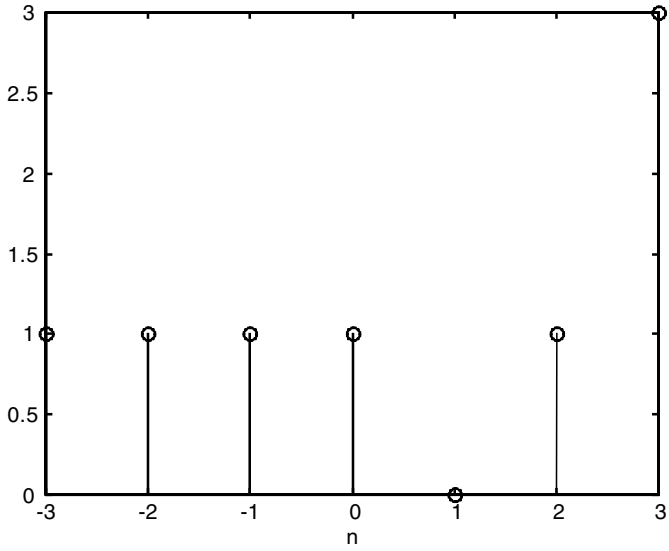

FIGURE 1.33 Signal for EOCE 1.7.

We can utilize the functions we have written so far to come up with the following script. Given $x(n)$ and its index n we write the script

```
[xref, n1] = xreflected(x, n);
[xe ne] = x1plusx2 (x, xref, n, n1);
[xo no] = x1plusx2 (x, -xref, n, n1);
% for plotting, we define the new indices a and b
a = min([ne, no])
b = max([ne, no])
c=-2; d=2; %y-axis range
subplot(2, 2, 1);
stem(n, x); title('The original signal'); xlabel('n');
axis([a b c d]);
subplot(2, 2, 2);
stem(n1,xref);
title('Reflected signal') ; xlabel('n');
axis([a b c d]);
subplot(2,2,3);
stem(ne, 0.5*xe); title('Even signal') ; xlabel('n');
axis([a b c d]);
subplot(2, 2, 4);
stem(no,0.5*xo);
title('Odd signal'); xlabel('n');
axis([a b c d]);
```

EOCE 1.9

Consider the following signal

$$x(n) = \delta(n+3) + \delta(n+2) + \delta(n+1) - \delta(n-1) - \delta(n-2) - \delta(n-3)$$

for $-5 \le n \le 5$. Find the even and odd parts of $x(n)$.

Solution

Although $x(n)$ can easily be seen as an odd signal, we will use MATLAB with the help of the functions and the scripts that we have written so far to plot the original signal, the reflected signal, the odd signal and the even signal.

First we call the function `impulsesignal` six times to get $x(n)$ and then use the script we just wrote for the odd and even parts of $x(n)$ to produce the plots.

```
n = [-5 : 5]; %the span of the original signal
x1 = impulsesignal(-3, -5, 5)+ impulsesignal(-2, -5, 5);
x2 = impulsesignal(-1, -5, 5) - impulsesignal(1, -5, 5);
x3 = impulsesignal(2, -5, 5) + impulsesignal(3, -5, 5);
x = x1 + x2 -x3;
[xref, n1] = xreflected(x, n);
[xe ne] = x1plusx2 (x, xref, n, n1);
[xo no] = x1plusx2 (x, -xref, n, n1);
% for plotting, we define the new indices a and b
a = min([ne, no])
b = max([ne, no])
c=-2; d=2; %y-axis range
subplot(2, 2, 1);
stem(n, x); title('The original signal'); xlabel('n');
axis([a b c d]);
subplot(2, 2, 2);
stem(n1,xref);
title('Reflected signal') ; xlabel('n');
axis([a b c d]);
subplot(2,2,3);
stem(ne, 0.5*xe); title('Even signal') ; xlabel('n');
axis([a b c d]);
subplot(2, 2, 4);
stem(no,0.5*xo);
title('Odd signal'); xlabel('n');
axis([a b c d]);
```

The plots are in Figure 1.34.

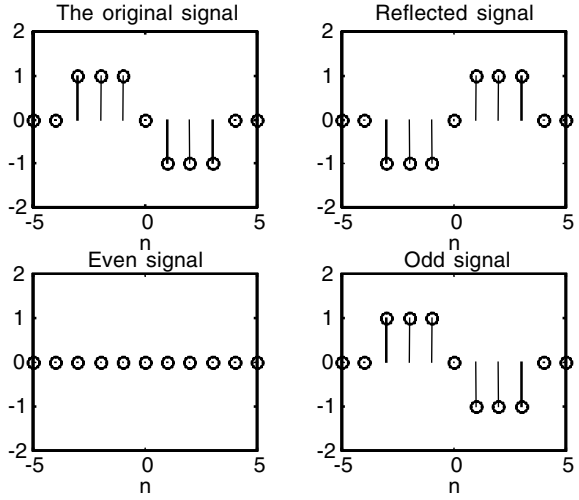

FIGURE 1.34 Signals for EOCE 1.9.

EOCE 1.10

Consider the following signal:

$$x(n) = u(n-1) + \delta(n+1) \quad -2 \le n \le 2$$

Find and plot the following signals:

1. $x(-n)$
2. $x(n-2)$
3. $x(n) + x(-n)$

Solution

1. Analytically,

$$x(n) = u(n-1) + \delta(n+1) = \left\{0, 0, \underset{\uparrow}{0}, 1, 1\right\} + \left\{0, 1, \underset{\uparrow}{0}, 0, 0\right\} = \left\{0, 1, \underset{\uparrow}{0}, 1, 1\right\}$$

and $x(-n)$ is $\{1, 1, \underset{\uparrow}{0}, 1, 0\}$. Using MATLAB, we first generate $x(n)$ then find its reflection.

```
n= [-2 : 2]
x1 = stepsignal(1, -2, 2);
x2= impulsesignal(-1, -2, 2);
x=x1+x2;
[xref nref] = xreflected(x, n);
```

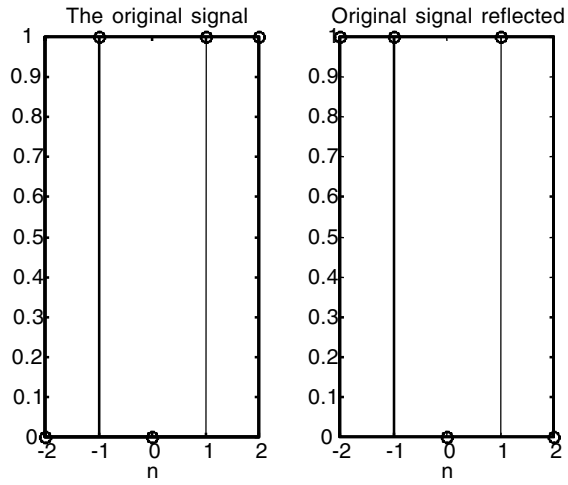

FIGURE 1.35 Signal for EOCE 1.10.

```
subplot(1,2,1);
stem(n,x);xlabel('n');title('The original signal');
subplot(1,2,2);
stem(nref, xref); title('Original signal reflected');
xlabel('n');
```

and the plot is shown in Figure 1.35.

2. Analytically,

$$x(n-2) = u(n-2-1) + \delta(n-2+1)$$

or we notice that $x(n–2)$ = is $x(n)$ shifted by 2 and is

$$x(n-2) = \left\{0, 1, 0, 1, 1\right\}_{\uparrow}$$

Using MATLAB, we use the shifting function we derived earlier to write

```
n= [-2 : 2]
x = stepsignal(1, -2, 2) + impulsesignal(-1, -2, 2)
[xshif, nshif] = xshifted(x,n, 2);
subplot(1,2,1);stem(n,x);xlabel('n');
title('The original signal');
subplot(1,2,2)
stem(nshif, xshif ); title('The shifted signal');
xlabel('n');
```

and the plot is shown in Figure 1.36.

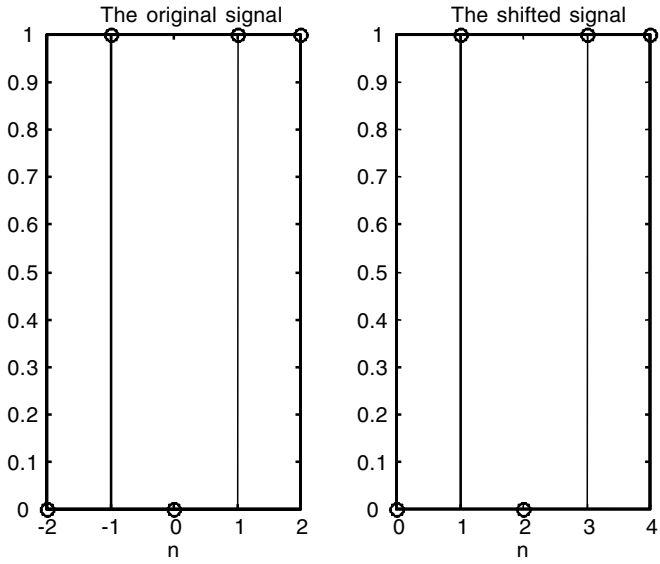

FIGURE 1.36 Signal for EOCE 1.10.

3. Using MATLAB, we can find $x(n) + x(-n)$ by writing the MATLAB script

```
n = [-2 : 2]
x1 = stepsignal(1, -2, 2)
x2=impulsesignal(-1, -2, 2)
x = x1 + x2;
[xref, nref] = xreflected(x, n);
[xfinal nfinal] = x1plusx2 (x, xref, n, nref);
stem(nfinal,xfinal); title('Original and the reflected
 added')
xlabel('n');
```

and the plot is in Figure 1.37.

1.23 End of Chapter Problems

EOCP 1.1

Analytically find the following signals if $x(n) = nu(n-1)$ $-\infty < n < \infty$

1. $x(-n)$
2. $x(-n+1)$

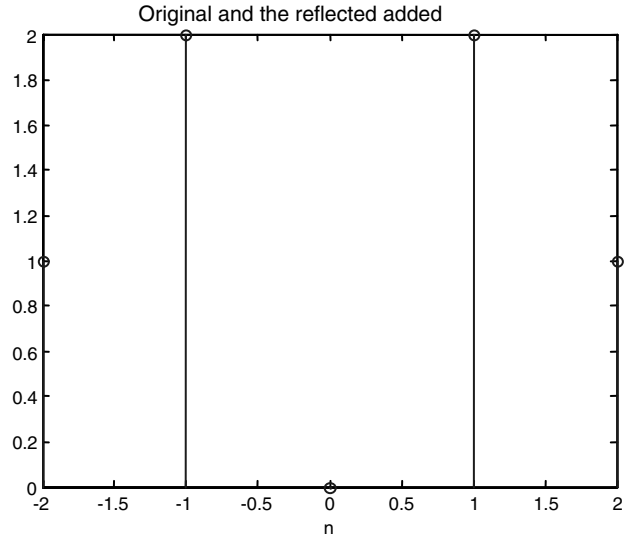

FIGURE 1.37 Signal for EOCE 1.10.

3. $x(n) + x(-n)$
4. $x(2n)$
5. $x\left(\frac{n}{3}\right) + x(-n)$
6. $x(n)\delta(n-1)$
7. $x(-n)\ u(n-2) + \delta(n)$
8. $x(n-2) + \delta(n)\ x(n)$
9. $u\left(\frac{n}{2}\right) - x(n)$
10. $x(-n-2) + u(n-2)$

EOCP 1.2

Use MATLAB to generate the following signals if $x(n) = u(n) - u(n-1)$ for $0 \leq n \leq 5$:

1. $x(-n)$
2. $x(n+2)$
3. $x(n) + x(-n)$
4. $x(n-2) + x(n+2)$
5. $x(-n-1) \cdot x(n)$
6. $x(-n) \cdot x(n) + x(-n-1)$
7. $x(n) + \cos(2\pi n + \pi)$
8. $x(-n) \cdot \cos(3\pi n + \frac{\pi}{2})$
9. $(.1)^n x(n) \cos(3\pi n + \frac{\pi}{2})$

EOCP 1.3

Check the periodicity for each of the following signals for $0 \le n \le \infty$. If they are periodic, what is the period?

1. $\cos(2\pi n + \pi)$
2. $(.1)^n \cos(5\pi n + \frac{\pi}{2})$
3. $u(n)$
4. $u(n) + 1$
5. $\delta(n) + u(n)$
6. $\cos\left(\sqrt{2}\pi n\right)$
7. $u(n) + \cos(2\pi n + \pi)$
8. $\cos(2\pi n + \pi) + \delta(n - 1)$
9. $2\cos(2n - \pi)$
10. $\cos\left(\frac{3}{2}n + \pi\right) + u(n)$

EOCP 1.4

Use MATLAB to check periodicity for the signal in EOCP 1.3.

EOCP 1.5

Find the power in the following signals:

1. $u(n)$ $\quad n \ge 0$
2. $u(n)$ $\quad n \ge 1$
3. $\sum_{m=0}^{\infty} \delta(n-m)$ $\quad n \ge 0$

EOCP 1.6

Find the energy in each of the following signals for $-5 \le n \le 5$:

1. $\delta(n)$
2. $\cos(2\pi n)$
3. $u(n) \,.\, \delta(n)$
4. $2u(n)\cos(2\pi n)$
5. $u(n) \,.\, u(-n)$
6. $n \cos(2\pi n)$

Find the energy in the following signals for $n > 0$:

1. $u(n)\,(.1)^n$
2. $(.1)^n \cos(2\pi n)$
3. $(.5)^n n$

EOCP 1.7

Consider the following signals.

1. $x(n) = u(n) + u(n-1) \quad 0 \le n \le 5$
2. $x(n) = nu(n) \quad 0 \le n \le 5$
3. $x(n) = (.1)^n \cos(2\pi n + 1) \quad 0 \le n \le 5$

a) Use MATLAB to sketch the even and the odd parts.
b) Show that the energy in $x(n)$ is the sum of the energy in its components, the even and the odd parts.
c) Are the signals bounded?

EOCP 1.8

Usually the discrete signals we deal with in engineering, $x(n)$, are obtained by taking samples from continuous signals $x(t)$. Give five examples where discrete signals are naturally discrete.

EOCP 1.9

Consider the following signals

1. $x(t) = e^{-3t}u(t)$
2. $x(t) = e^{-t}\cos(1000t)u(t)$

a) Let us take samples from both signals every 2 sec. Find $x(n)$ for both.
b) What is the time constant for the first signal?
c) If $0 \le n \le 10$, find the energy in $x(n)$ for both signals.

EOCP 1.10

Let $y(n) = y(n-1) + u(n)$ with $y(-1) = 1$ for $n \ge 0$

1. Write down the samples for $y(n)$.
2. Can you find a closed form equation for $y(n)$?

EOCP 1.11

Let $y(-1) = 1$ and consider the equation

$$y(n) = 2y(n-1) + u(n)$$

1. Find the samples for $y(n)$ for $n \ge 0$.
2. Find a mathematical closed form expression for $y(n)$.

2

The Discrete System

2.1 Definition of a System

A system is an assemblage of things that are combined to form a complex whole. When we study systems, we study the interactions and behaviors of such an assemblage when subjected to certain conditions. These conditions are called inputs. In its simplest case a system can have one input and one output. This book deals with linear systems. We will call the input $x(n)$ and the output $y(n)$ as depicted in Figure 2.1.

2.2 Input and Output

The discrete signal $x(n)$ is the continuous signal $x(t)$ sampled at $t = nT_s$ where t is the continuous time, n is an integer and T_s is the sampling interval. We will talk about sampling later in the text. If an input signal $x(t)$ is available at the input of a linear system, the system will operate on the signal $x(t)$ and produce an output signal that we call $y(t)$. As an example, consider the case of an elevator where you push a button to go to the fifth floor. Pushing the button is the input $x(t)$. The elevator is the system under consideration here. In addition to many other components, the elevator system consists of the small room to ride in and the motor that drives the elevator belt. The input signal $x(t)$ "asks" the elevator to move to the fifth floor. The elevator system will process this request and move to the fifth floor. The motion of the elevator to the selected floor is the output $y(t)$.

Pushing the button in this elevator case produces an electrical signal $x(t)$. This signal drives the motor of the elevator to produce a rotational motion which is transferred, via some gears, to a translational motion. This translational motion is the output $y(t)$. To summarize, when an electrical input signal or request $x(t)$ is applied to the elevator system, the elevator will operate on the signal and produce $y(t)$ which, in this example, can be thought of as a translational motion.

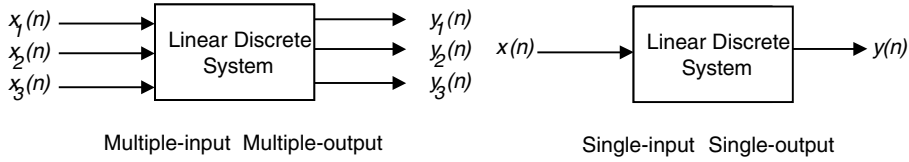

FIGURE 2.1 Linear system representation.

The example that we just considered is inherently a continuous system. An example of a discrete or digital system is the digital computer. The computer has many inputs and many outputs. An input can be generated by pressing any key on the keyboard and an output can be the display of the character that the user presses on the keyboard on the screen. The mouse and the scanner can be thought of as inputs as well. A computer program is a digital set of zeros and ones. It has an input set of raw data. The output of the computer program is the useful data.

2.3 Linear Discrete Systems

A linear discrete system has the following properties:

1. If the input is $\alpha\, x_1(n)$, the output is $\alpha\, y_1(n)$.
2. If the input is $x_1(n) + x_2(n)$, the output is $y_1(n) + y_2(n)$.

By combining the two conditions, a system is considered linearly discrete if for the input $\alpha\, x_1(n) + \beta\, x_1(n)$, the output is $\alpha\, y_1(n) + \beta\, y_2(n)$, where α and β are constants.

Example 2.1

Consider the input–output relation

$$y(n) = \left[\frac{R_2}{R_1 + R_2}\right] x(n)$$

where α and β are constants. Is this system linear?

Solution

Using the definition of linearity introduced previously we can proceed as follows. If the input is $x_1(n)$, the output is $y_1(n)$; therefore, we write

$$y_1(n) = \left[\frac{R_2}{R_1 + R_2}\right] x_1(n)$$

If the input is $x_2(n)$, the output is $y_2(n)$, therefore

$$y_2(n) = \left[\frac{R_2}{R_1 + R_2}\right] x_2(n)$$

If the input is $\alpha x_1(n) + \beta x_2(n)$, the output is $y(n)$, therefore

$$y(n) = \left[\frac{R_2}{R_1 + R_2}\right] \left(\alpha x_1(n) + \beta x_2(n)\right) = \alpha y_1(n) + \beta y_2(n)$$

Thus the system is said to be linear.

Example 2.2

Consider the following system

$$y(n) = \sqrt{x(n)}$$

Is this system linear?

Solution

Consider two cases of the input signals, $x_1(n)$ and $x_2(n)$. The corresponding outputs are then given by

$$y_1(n) = \sqrt{x_1(n)}$$

for $x_1(n)$ and

$$y_2(n) = \sqrt{x_2(n)}$$

for $x_2(n)$. Now let us consider further that $\alpha x_1(n) + \beta x_2(n)$ has been applied to the system as its input. The corresponding output is then given by

$$y(n) = \sqrt{\alpha x_1(n) + \beta x_2(n)}$$

But

$$y(n) = \sqrt{\alpha x_1(n) + \beta x_2(n)} \neq \alpha y_1(n) + \beta y_2(n) = \alpha \sqrt{x_1(n)} + \beta \sqrt{x_2(n)}$$

Therefore, the system is nonlinear.

Example 2.3

Consider the following system

$$y(n) = \left(2/\left[2x(n)+1\right]\right) x(n)$$

Is this system linear?

Solution

As in Example 2.2, consider two cases of the input signals, $x_1(n)$ and $x_2(n)$. The corresponding outputs are then given by

$$y_1(n) = \left(2/\left[2x_1(n)+1\right]\right) x_1(n)$$

and

$$y_2(n) = \left(2/\left[2x_2(n)+1\right]\right) x_2(n)$$

Now let us apply $\alpha x_1(n) + \beta x_2(n)$ to the system as its input. The corresponding output is then given by

$$y(n) = \left(2/\left[2\left(\alpha x_1(n) + \beta x_2(n)\right)+1\right]\right) \left(\alpha x_1(n) + \beta x_2(n)\right)$$

or

$$\begin{aligned} y(n) &= \left[2\alpha x_1(n) + 2\beta x_2(n)\right]/\left[2\alpha x_1(n) + 2\beta x_2(n) + 1\right] \\ &\neq \alpha y_1(n) + \beta y_2(n) \\ &= \left[2\alpha x_1(n)\right]/\left[2x_1(n)+1\right] + \left[2\beta x_2(n)\right]/\left[2x_2(n)+1\right] \end{aligned}$$

Therefore, the system is nonlinear.

2.4 Time Invariance and Discrete Signals

A system is said to be time invariant if, for a shifted input $x(n - n_0)$, the output of the system is $y(n - n_0)$. To see if a system is time invariant or time variant we do the following:

1. Find the output $y_1(n - n_0)$ that corresponds to the input $x_1(n)$.
2. Let $x_2(n) = x_1(n - n_0)$ and then find the corresponding output $y_2(n)$.
3. Check if $y_1(n - n_0) = y_2(n)$. If this is true then the system is time invariant. Otherwise it is time variant.

Example 2.4

Let $y(n) = \cos(x(n))$. Find out if the system is time variant or time invariant.

Solution

Step 1.

$$y_1(n) = \cos\left(x_1(n)\right)$$

$$y_1(n) \text{ shifted by } n_0 \text{ is } y_1\left(n - n_0\right) = \cos\left(x_1\left(n - n_0\right)\right)$$

Step 2.

$$y_2(n) = \cos\left(x_1\left(n - n_0\right)\right)$$

Step 3.

$$y_1\left(n - n_0\right) = y_2(n)$$

Thus the system is time invariant.

Example 2.5

Let $y(n) = x(n)\cos(n)$. Find out if the system is time variant or time invariant.

Solution

Step 1.

$$y_1(n) = x_1(n)\cos(n)$$

Therefore,

$$y_1\left(n - n_0\right) = x_1\left(n - n_0\right)\cos\left(n - n_0\right)$$

Step 2.

$$y_2(n) = x_1\left(n - n_0\right)\cos(n)$$

Step 3.

$$y_1(n-n_0) \neq y_2(n)$$

Therefore, the system is time variant.

Example 2.6

Let $y(n) = ne^{-n}x(n)$. Find out if the system is time variant or time invariant.

Solution

Step 1.

$$y_1(n) = ne^{-n}x_1(n)$$

Therefore,

$$y_1(n-n_0) = (n-n_0)e^{-(n-n_0)}x_1(n-n_0)$$

Step 2.

$$y_2(n) = ne^{-n}x_1(n-n_0)$$

Step 3.

$$y_1(n-n_0) \neq y_2(n)$$

Therefore, the system is time variant.

2.5 Systems with Memory

If at any value of n, $y(n)$ depends totally on $x(n)$ at that particular value, then in such a case we say the system is without memory. Otherwise the system is with memory.

Example 2.7

Consider the input-output relation

$$y(n) = (x(n))^2$$

Is the system with or without memory?

Solution

For any value of n, $y(n)$ depends on $x(n)$ at that particular value. If we look at the output at $n = 4$ then we look at the input at $n = 4$ as well. In this case, the system is without memory.

Example 2.8

Consider the system

$$y(n) = nx(n-1)$$

Is the system with or without memory?

Solution

The output $y(n)$ at $n = 0$ depends on $x(n)$ at $n = -1$. Therefore, the system is with memory.

2.6 Causal Systems

A causal system is a system where the output $y(n)$ at a certain time n_1 depends on the input $x(n)$ for $n < n_1$.

Example 2.9

If $x(n)$ is given as

$$x(n) = \left\{\underset{\uparrow}{1}, 1, 1\right\}$$

and the output $y(n)$ is

$$y(n) = \left\{\underset{\uparrow}{1}, 1/2, 1/4\right\}$$

Is this system causal?

Solution

The first sample of the input has appeared at $n = 0$, as does the first sample of the output. Therefore, the system is causal. Note that the system is causal even if the output starts at any value for which $n \geq 0$.

Example 2.10

Let the input $x(n)$ be as

$$x(n) = \left\{\underset{\uparrow}{1}, 0, 1, 0\right\}$$

Let this signal be the input to a system where the output was recorded as

$$x(n) = \left\{1, 1, 0, \underset{\uparrow}{1}, 0\right\}$$

Is the system causal?

Solution

The first input sample starts at $n = 0$ and the first output sample starts at $n = -3$. This makes the system noncausal.

2.7 The Inverse of a System

If we can determine the input by measuring the output, then the system under consideration is said to be invertible. Note that if two inputs give the same output the system is not invertible.

Example 2.11

Consider the following systems

1. $y(n) = x(n)$
2. $y(n) = 2x(n)$
3. $y(n) = a\cos(x(n))$

Are these systems invertible?

Solution

The first system is invertible. The corresponding pictorial representation is shown in Figure 2.2. The second system is also invertible. Figure 2.3 shows the corresponding pictorial depiction. The third system is not invertible. Why?

Let us consider two inputs, $x_1(n) = x(n)$ and $x_2(n) = x(n) + n\pi$, where n is an even integer. For this system the output corresponding to $x_1(n) = x(n)$ is given by

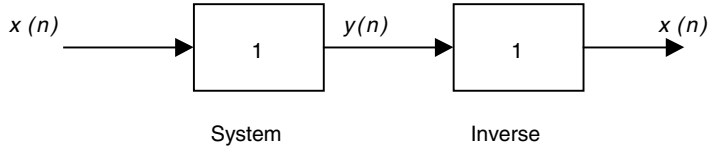

FIGURE 2.2 System for Example 2.11.

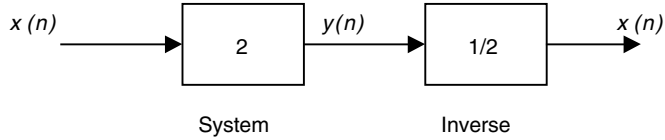

FIGURE 2.3 System for Example 2.11.

$$y_1(n) = a\cos(x(n))$$

and the output corresponding to $x_2(n) = x(n) + n\pi$ is given by

$$y_2(n) = a\cos(x(n) + n\pi) = a\cos(x(n))$$

We can see here that two different inputs produced the same output. Therefore, the system is not invertible as claimed.

2.8 Stable System

The signal $x(n)$ is considered bounded if $|x(n)| < \beta < \infty$ for all n, where β is a real number. A system is said to be BIBO (bounded-input bounded-output) stable if and when the input is bounded the output is also bounded. $y(n)$, the output, is bounded if $|y(n)| < \beta < \infty$.

Example 2.12

Consider the system

$$y(n) = \sum_{k=0}^{M-1} x(n-k)$$

and assume that $x(n)$ is bounded. Is the system stable?

Solution

If $x(n)$ is bounded, this implies that

$$|x(n)| < \beta$$

But a shifted version of $x(n)$ is also bounded for $x(n)$ is bounded. Therefore,

$$|y(n)| = \left| \sum_{k=0}^{M-1} x(n-k) \right| \leq M\beta \leq \beta$$

and the system is BIBO.

Example 2.13

Consider that $x(n)$ is bounded and applied to a system where $y(n)$ is obtained as

$$y(n) = n \sum_{k=0}^{M-1} x(n-k)$$

Is the system BIBO?

Solution

Since $x(n)$ is bounded, we have $|x(n)| < \beta$. We also know that a shifted version of $x(n)$ is also bounded. Thus

$$|y(n)| = \left| n \sum_{k=0}^{M-1} |x(n-k)| \right| < nM\beta$$

However, as n approaches infinity $y(n)$ will grow without bounds and the system is not BIBO.

2.9 Convolution

To find the output of a discrete system $y(n)$ to an input $x(n)$, we need the impulse response, $h(n)$, for the system. $h(n)$ is the output of the system when the input is $\delta(n)$, where $\delta(n)$ is the impulse signal. If we apply the signal $\delta(n)$ to the system as shown in Figure 2.4, the output is $h(n)$. Notice that we do

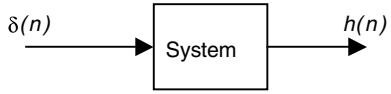

FIGURE 2.4 The response to the impulse signal.

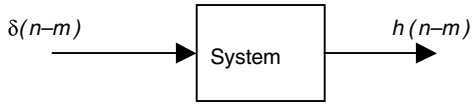

FIGURE 2.5 The response to the shifted impulse signal.

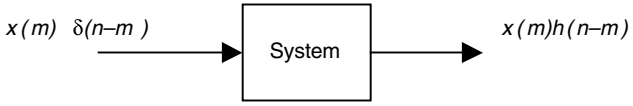

FIGURE 2.6 The response to the shifted impulse signal multiplied by a constant.

not know the system itself (the input–output relation) but we know that if the input is the impulse signal $\delta(n)$, the output will be the impulse response $h(n)$.

Now consider the shifted impulse signal $\delta(n - m)$ to the same system as shown in Figure 2.5. The output will also be shifted because we are considering only linear systems. Therefore, $\delta(n - m)$ will produce $h(n - m)$.

We have also seen in Chapter 1 that any discrete signal can be represented as the sum of weighted shifted impulses (samples). We have seen that the signal $x(n)$ can be represented as

$$x(n) = \sum_{m=-\infty}^{+\infty} x(m)\delta(n-m) \tag{2.1}$$

Note that each $x(m)$ is a sample and a constant. Also, if $x(m)$ is multiplied by $\delta(n - m)$ and applied to the discrete linear time-invariant system, the output will be $x(m)h(n - m)$ as shown in Figure 2.6.

Now let us say that we were to add all the shifted weighted samples

$$\sum_{m=-\infty}^{+\infty} x(m)\delta(n-m)$$

and present this as input to the same system. The output in this case is

$$\sum_{m=-\infty}^{+\infty} x(m)h(n-m)$$

which is the sum of all the responses to each weighted sample individually. But

$$\sum_{m=-\infty}^{+\infty} x(m)\delta(n-m) = x(n)$$

Knowing that when $x(n)$ is presented as an input to a linear time-invariant system the output is $y(n)$ allows us to write

$$y(n) = \sum_{m=-\infty}^{+\infty} x(m)h(n-m) \tag{2.2}$$

or

$$y(n) = x(n) * h(n)$$

The above equation is the convolution equation that, given $x(n)$, the input to a discrete system, and $h(n)$, the impulse response, will give you the output $y(n)$. This also tells you that, given $h(n)$ for any system, you can find $y(n)$ for any input $x(n)$.

Example 2.14

Consider the system in which the impulse response is known to be

$$h(n) = (2)^n u(n) \qquad n \geq 0$$

Find the output $y(n)$ for the input $x(n) = \delta(n)$.

Solution

Note that we expect the output to be the $h(n)$ given because the input is $\delta(n)$. Using the convolution sum we have

$$y(n) = \sum_{m=-\infty}^{\infty} x(m)h(n-m) = \sum_{m=-\infty}^{\infty} \delta(m)(2)^{n-m} u(n-m)$$

But $\delta(m)$ is only defined at $m = 0$. Therefore

$$y(n) = \delta(0)(2)^{n-0} u(n-0) = 1(2)^n u(n) = h(n) \qquad n \geq 0$$

as anticipated.

Example 2.15

Consider the input $x(n) = u(n)$ and the impulse response $h(n) = (.5)^n u(n)$ for a certain system. What is the output of the system?

Solution

Using the convolution equation we write

$$y(n) = \sum_{-\infty}^{+\infty} x(m)u(n-m) = \sum_{m=-\infty}^{+\infty} u(m)(.5)^{n-m}u(n-m)$$

But since $u(m) = 1$ for $n \geq 0$ and both $x(n)$ and $h(n)$ start at $n = 0$, we have

$$y(n) = \sum_{m=0}^{m=n}(.5)^{n-m} = (.5)^n \sum_{m=0}^{n}(.5)^{-m} = (.5)^n \sum_{m=0}^{n}\left((.5)^{-1}\right)^m$$

$$y(n) = (.5)^n \frac{\left[1-\left((.5)^{-1}\right)^{n+1}\right]}{1-(.5)^{-1}} \qquad n \geq 0$$

The last result was obtained using the geometric series sum

$$S = \sum_{m=0}^{m=n}(a)^m = \frac{\left[1-(a)^{n+1}\right]}{1-a}$$

Example 2.16

Let $x(n) = (.5)^n + (.6)^{n+1}$ and $h(n) = u(n)$ for $n \geq 0$. Find $y(n)$.

Solution

Using the convolution equation we write

$$y(n) = \sum_{m=-\infty}^{m=\infty} u(m)\left[(.5)^{n-m} + (.6)^{n-m+1}\right]u(n-m)$$

which reduces to

$$y(n) = \sum_{m=0}^{m=n}(.5)^{n-m} + (.6)^{n-m+1} = (.5)^n \frac{\left[1-\left((.5)^{-1}\right)^{n+1}\right]}{1-(.5)^{-1}} + (.6)^{n+1}\frac{\left[1-\left((.6)^{-1}\right)^{n+1}\right]}{1-(.6)^{-1}}$$

Example 2.17

Consider the system where $x(n) = Au(n)$ and $h(n) = Bu(n)$. Find the output $y(n)$.

Solution

Using the convolution sum we have

$$y(n) = \sum_{m=-\infty}^{+\infty} Au(m)Bu(n-m)$$

which reduces to

$$y(n) = \sum_{m=0}^{m=n} AB = ABn \qquad n \geq 0$$

Note that this system grows without bounds as n approaches infinity.

2.10 Difference Equations of Physical Systems

A difference equation that represents the input–output relation for a discrete linear time-invariant system is of the form

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k) \tag{2.3}$$

with the initial conditions $y(-1)$, $y(-2)$,…$y(-N)$. The relation in the equation above is a general relation between $x(n)$, the input, and $y(n)$, the output of the discrete system. The order of this general difference equation is N. We also say that N is the degree of the system. An example is the difference equation

$$y(n) - 2y(n-1) = x(n)$$

where N here is 1, the order of the discrete difference equation; thus it is a first-order difference equation. Our goal is to find the output $y(n)$ given the input $x(n)$.

The total solution has two parts: the homogeneous solution and the particular solution. We will learn how to find the total solution in the following section. But first, we will see how to find the homogeneous solution.

2.11 The Homogeneous Difference Equation and Its Solution

The general form of a homogeneous difference equation is

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = 0 \tag{2.4}$$

with initial conditions $y(-1)$, $y(-2)$,…$y(-N)$ and all inputs set to zero. You can see that without inputs, the homogeneous solution is zero unless we have nonzero initial conditions.

To find $y_h(n)$, the homogeneous solution, we assume a solution of the form $c(p)^n$. If $c(p)^n$ is a solution it must satisfy the homogeneous equation above. So we substitute $c(p)^n$ into Equation (2.4) to get

$$c(p)^n - \sum_{k=1}^{N} a_k c(p)^{n-k} = 0$$

Since the summation in the above equation is over k, we write

$$c(p)^n - c(p)^n \sum_{k=1}^{N} a_k p^{-k} = 0$$

By expanding the above equation we get

$$c(p)^n \left[1 - a_1 p^{-1} - a_2 p^{-2} - a_3 p^{-3} - \ldots - a_N p^{-N}\right] = 0$$

To satisfy the above equation, either

$$c(p)^n = 0$$

or

$$\left[1 - a_1 p^{-1} - a_2 p^{-2} - a_3 p^{-3} - \ldots - a_N p^{-N}\right] = 0$$

But $c(p)^n$ cannot be zero. Therefore,

$$\left[1 - a_1 p^{-1} - a_2 p^{-2} - a_3 p^{-3} - \ldots - a_N p^{-N}\right] = 0 \tag{2.5}$$

This equation is called the characteristic equation of the system. If we multiply this equation by p^N we will have

$$\left[p^N - a_1 p^{N-1} - a_2 p^{N-2} - a_3 p^{N-3} - \ldots - a_N\right] = 0$$

So the homogeneous solution $y_h(n)$ is

$$y_h(n) = c_1(p_1)^n + c_2(p_2)^n + \ldots + c_N(p_N)^n \tag{2.6}$$

where $p_1, p_2, \ldots, p_N$ are the roots of

$$\left[p^N - a_1 p^{N-1} - a_2 p^{N-2} - a_3 p^{N-3} - \ldots - a_N\right] = 0$$

and $c_1, c_2, \ldots, c_N$ are constant to be determined using the given initial conditions.

Example 2.18

Consider the homogeneous difference equation that describes a discrete system as

$$y(n) - 2y(n-1) = 0$$

with the initial condition $y(-1) = +1$. What is $y(n)$? Check your answer using iterations.

Solution

The solution $y(n)$ is the sum of the homogeneous and the particular parts. The homogeneous part is due to the initial condition and the particular part is due to the existing external inputs. In this case there are no external inputs and so the particular part is zero. The homogeneous part is calculated by first assuming a solution of the form $y_h(n) = cp^n$ and then substituting in the given equation to get

$$cp^n - 2cp^{n-1} = 0$$

or

$$cp^n\left(1 - 2p^{-1}\right) = 0$$

The characteristic equation is then $1 - 2p^{-1} = 0$. Solving for p we get $p = 2$. The homogeneous solution is then

$$y_h(n) = c_1(2)^n$$

To find c_1 we use the initial condition and write

$$y(-1) = 1 = c_1(2)^{-1}$$

The final solution is then

$$y(n) = y_h(n) + y_p(n) = 2(2)^n u(n)$$

We can check the validity of the solution obtained by iteration. We can rewrite the given difference equation as

$$y(n) = 2y(n-1)$$

With the given initial condition we can find $y(n)$ for $n \geq 0$ as in the following:

$$\text{For } n = 0,\ y(0) = 2y(-1) = 2(1) = 2$$

$$\text{For } n = 1,\ y(1) = 2y(0) = 2(2) = 4$$

$$\text{For } n = 2,\ y(2) = 2y(1) = 2(4) = 8$$

$$\text{For } n = 3,\ y(3) = 2y(2) = 2(8) = 16$$

These values can be checked via the closed form solution

$$y(n) = 2(2)^n$$

$$\text{For } n = 0,\ y(0) = 2(2)^0 = 2$$

$$\text{For } n = 1,\ y(1) = 2(2)^1 = 4$$

$$\text{For } n = 2,\ y(2) = 2(2)^2 = 8$$

$$\text{For } n = 3,\ y(3) = 2(2)^3 = 16$$

If we are interested in the first few values of $y(n)$ we can use iteration, and if we are interested, for example, in $y(1000)$, we better find the closed form solution.

2.11.1 Case When Roots Are All Distinct

When all roots are distinct the form of the homogeneous solution is

$$y_h(n) = c_1 p_1^n + c_2 p_2^n + \ldots + c_n p_n^n \tag{2.7}$$

where $p_1, p_2, \ldots, p_n$ are the roots of the characteristic equation and $c_1, c_2, \ldots, c_n$ are to be determined using the given initial conditions.

2.11.2 Case When Two Roots Are Real and Equal

If we consider a second order discrete system, the roots in this case will be equal and real. Denoting the roots as $p_1 = p_2 = p$, the homogeneous solution in this case is

$$y_h(n) = c_1 p^n + c_2 n p^n \tag{2.8}$$

The reason for multiplying the second term by n is to make the two terms independent. If we have three equal real and repeating roots, the homogeneous solution is

$$y_h(n) = c_1 p^n + c_2 n p^n + c_3 n^2 p^n \tag{2.9}$$

If two roots are real and equal and one root is real then

$$y_h(n) = c_1 p^n + c_2 n p^n + c_3 \left(p_3\right)^n \tag{2.10}$$

where $p = p_1 = p_2$ and p_3 is the other real root.

2.11.3 Case When Two Roots Are Complex

Suppose the roots in this case are p_1 and p_2. Complex roots always appear as complex conjugates. So if $p_1 = a + \mathrm{j}b$, then $p_2 = a - \mathrm{j}b$. Then we can put p_1 and p_2 in polar form and get

$$p_1 = Me^{j\theta}$$

$$p_2 = Me^{-j\theta}$$

and the homogeneous solution is then

$$y_h(n) = c_1 \left(Me^{j\theta}\right)^n + c_2 \left(Me^{-j\theta}\right)^n$$

For this solution to be real, c_1 must be the complex conjugate of c_2. Because of that we write the solution as

$$y_h(n) = c_1 \left(Me^{j\theta}\right)^n + c_1^* \left(Me^{-j\theta}\right)^n$$

Since c_1 is the complex conjugate of c_2, the two terms in the above solution are conjugates. If

$$z_1 = a + jb$$

$$z_2 = a - jb$$

then

$$z_1 + z_2 = 2a = 2\text{real}(z_1) = 2\text{real}(z_2)$$

where real stands for the real part of the complex number. The constants can be written in polar form as $c_1 = Qe^{j\beta}$ and $c_2 = Qe^{-j\beta}$. With this at hand, we rewrite the homogeneous solution as

$$y_h(n) = Qe^{j\beta}\left(Me^{j\theta}\right)^n + Qe^{-j\beta}\left(Me^{-j\theta}\right)^n = 2\text{real}\left[Qe^{j\beta}\left(Me^{j\theta}\right)^n\right] = 2\text{real}\left[QM^n e^{j(\beta+\theta n)}\right]$$

$$y_h(n) = 2QM^n\left(\text{real}\left[\cos(\theta n+\beta) + j\sin(\theta n+\beta)\right]\right) = 2QM^n\cos(\theta n+\beta) \tag{2.11}$$

where $\theta = \tan^{-1}(b/a)$, Q is the magnitude of c_1 and $\beta = \angle c_1$ is the angle of the complex number c_1. The only two constants to be found in Equation (2.11) are β and Q. They can be found using the initial conditions.

2.12 Nonhomogeneous Difference Equations and their Solutions

The solution of the nonhomogeneous difference equation

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_x x(n-k)$$

can be obtained by finding the homogeneous solution of

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = 0$$

with the initial conditions

$$y(-1),\ y(-2), \ldots, y(-N)$$

then adding it to the particular solution of

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_x x(n-k)$$

So we write the total solution as

$$y(n) = y_h(n) + y_p(n)$$

Note that if we are given a nonhomogeneous difference equation with initial conditions, we first find $y_h(n)$ and do not evaluate the constants associated with $y_h(n)$ because these initial conditions are given for the total solution $y(n)$. The following example will illustrate this point.

Example 2.19

Consider the system described by the difference equation

$$y(n) - 2y(n-1) = x(n)$$

where $x(n) = u(n)$ for $n \geq 0$ and the initial conditions are $y(-1) = 1$.

Solution

We start by finding the solution to the homogeneous part

$$y(n) - 2y(n-1) = 0$$

The characteristic equation is obtained by inspection as $p - 2 = 0$, and this gives $p = 2$. The homogeneous solution is then

$$y_h(n) = c_1(2)^n$$

We will not use the initial conditions to find c_1, but we will wait until we find $y_p(n)$ and then apply them to the total solution $y(n)$.

For the particular solution, and since our input is $u(n)$, a constant for $n \geq 0$, we predict that $y_p(n)$ is the constant k_1. This k_1 should be evaluated by substituting $y_p(n) = k_1$ in the given difference equation to obtain

$$k_1 - 2k_1 = 1 \text{ or } k_1 = -1$$

Therefore, the total solution $y(n)$ is

$$y(n) = y_h(n) + y_p(n) = c_1 2^n - 1$$

In this total solution, we use the initial conditions to find c_1. We have $y(-1) = 1$. Thus,

$$y(-1) = c_1 2^{-1} - 1 = 1$$

and the constant is $c_1 = 4$. The total solution is then

$$y(n) = (4)2^n - 1 \text{ for } n \geq 0$$

TABLE 2.1

Particular Solutions for Selected Inputs

$x(n)$	$y(n)$
$ku(n)$	k_1
$k\alpha^n$	$k_1\alpha^n$
kn	$k_1 n + k_2$
$k\delta(n)$	0
$k\cos(n\theta)$	$k_1 \cos(n\theta) + k_2 \sin(n\theta)$
$k\sin(n\theta)$	$k_1\cos(n\theta) + k_2 \sin(n\theta)$
$k\alpha^n \cos(n\theta)$	$k_1\alpha^n \cos(n\theta) + k_2\alpha^n \sin(n\theta)$

To test the validity of this solution, let us try to find the first few values of $y(n)$ by iteration. We were given that

$$y(n) - 2y(n-1) = u(n)$$

So

$$y(0) = 2y(-1) + u(0) = 2(1) + 1 = 3$$

and

$$y(1) = 2y(0) + u(1) = 2(3) + 1 = 7$$

In the derived closed form solution, $y(n) = (4)2^n - 1$, so $y(0) = (4)2^0 - 1 = 3$, and $y(1) = (4)2^1 - 1 = 7$.

2.12.1 How Do We Find the Particular Solution?

Usually the particular solution is the solution because of the inputs applied to the systems. There are no general rules as to the form of the particular solution. All we can do is guess. For a long time, certain particular solution forms were proven true for certain inputs. The forms are listed in Table 2.1 along with the given input $x(n)$.

2.13 The Stability of Linear Discrete Systems: The Characteristic Equation

2.13.1 Stability Depending On the Values of the Poles

We can determine if the discrete linear system is stable or not by evaluating the roots of the characteristic equation

$$p^N + a_1 p^{N-1} + a_2 p^{N-2} + \ldots + a_N = 0$$

If the roots are within the unit circle, then the system is stable. If any root is outside this range, then the system is unstable. In this case we can solve for

the eigenvalues or the poles of the system using the MATLAB function roots. If the characteristic equation is given as

$$P^2 + 0.5p + 1 = 0$$

then using MATLAB we type the command

```
roots([1 0.5 1])
```

to get the roots. But we are interested in the magnitude of the roots to make sure that this magnitude is within the unit circle. Thus we modify the command and write

```
abs(roots([1 0.5 1]))
```

2.13.2 Stability from the Jury Test

Sometimes the real coefficients, a_1, a_2, … , a_N in the characteristic equation are not constants; they are variables on which the stability of the system depends. In such a case we can use the Jury test to find out about the stability of the system. For higher order systems the Jury test is superior.

To understand this test we will consider the following general characteristic polynomial

$$a_5\, p^5 + a_4 p^4 + a_3 p^3 + a_2 p^2 + a_1 p^1 + a_0 = 0$$

with a_5 different from zero and positive. Next we arrange the leading coefficients as in the following table.

$a_5 \quad a_4 \quad a_3 \quad a_2 \quad a_1 \quad a_0$	
$\left[a_0 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5\right] a_0/a_5$	Subtract
$\boxed{b_4} \quad b_3 \quad b_2 \quad b_1 \quad b_0 \quad 0$	
$\left[b_0 \quad b_1 \quad b_2 \quad b_3 \quad b_4\right] b_0/b_4$	Subtract
$\boxed{c_3} \quad c_2 \quad c_1 \quad c_0 \quad 0$	
$\left[c_0 \quad c_1 \quad c_2 \quad c_3\right] c_0/c_3$	Subtract
$\boxed{d_2} \quad d_1 \quad d_0 \quad 0$	
$\left[d_0 \quad d_1 \quad d_2\right] d_0/d_2$	Subtract
$\boxed{e_1} \quad e_0 \quad 0$	
$\left[e_0 \quad e_1\right] e_0/e_1$	Subtract
$\boxed{f_0} \quad 0$	

The first row in the table contains the coefficients in the characteristic equation in descending powers of p. The second row is the first row reversed. We then multiply the second row by the last element in row 1 divided by the last element in row 2, then subtract [row 2]a_0/a_5 from row 1 to get the third row. The process continues in this fashion until we get to the last row with one element only, f_0 in this case.

For the system represented by this characteristic equation to be stable, the leading boxed coefficients must all be positive. Let us look at an example to illustrate the process. Consider the characteristic equation

$$3p^2 + 2kp + 1 = 0$$

The table is arranged as in the following.

$3 \quad 2k \quad 1$ $[1 \quad 2k \quad 3]1/3 \qquad$ subtract	
$3-1(1/3)=\boxed{8/3} \quad 2k-2k(1/3)=4k/3$	0
$[4k/3 \qquad 8/3](4k/3)/(8/3)$	
$8/3-4k/3[(4k/3)/(8/3)]=\boxed{\dfrac{8-2k^2}{3}}$	0

Stability requires that the boxed elements in the table be positive. This requires that

$$\frac{8-2k^2}{3} > 0$$

or

$$8-2k^2 = (2-k)(2+k) > 0$$

Thus for stability, the values for k should be limited to $-2 < k < 2$.

Example 2.20

Consider the system

$$y(n) - 2y(n-1) = x(n)$$

Is the system stable?

Solution

The character equation is

$$p - 2 = 0$$

with $p = 2$ as the root. Since $p > 1$, the system is unstable. With the Jury test we have

$1 \quad -2$	
$[-2 \quad 1](-2/1)$	subtract
$\boxed{-3} \quad 0$	

–3 is not positive. Thus the system is unstable.

Example 2.21

Consider the system

$$y(n) - 0.5y(n-1) = x(n)$$

Is the system stable?

Solution

The character equation is

$$p - 0.5 = 0$$

with $p = 5$. Since $0 \le 5 \le 1$, the system is stable.

With the jury test we have

$1 \quad -0.5$	
$[-0.5 \quad 1](-0.5/1)$	subtract
$\boxed{3/4} \quad 0$	

3/4 is positive and the system is stable.

2.14 Block Diagram Representation of Linear Discrete Systems

So far we have seen linear discrete systems represented as difference equations. These systems can also be represented as block diagrams. The main components for the block diagrams are given next.

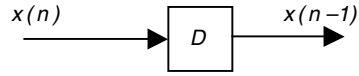

FIGURE 2.7 The delay element.

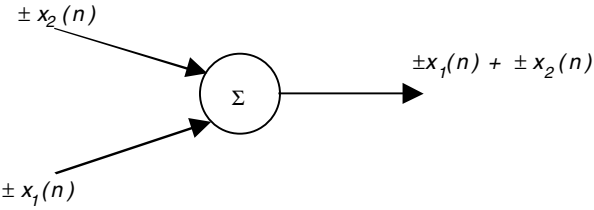

FIGURE 2.8 The summing/subtracting junction.

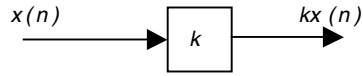

FIGURE 2.9 The multiplier element.

2.14.1 The Delay Element

The delay element is shown in Figure 2.7. In this case

$$y(n) = x(n-1)$$

where D in the diagram is the delay time. The delay element can be implemented physically as a shift register.

2.14.2 The Summing/Subtracting Junction

This junction is shown in Figure 2.8. The output here is

$$y(n) = x_1(n) \pm x_2(n)$$

2.14.3 The Multiplier

The multiplier is shown in Figure 2.9. In this case

$$y(n) = kx(n)$$

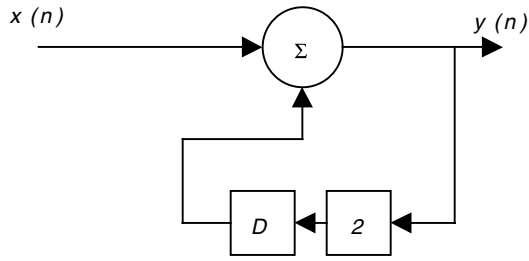

FIGURE 2.10 Block diagram for Example 2.22.

Example 2.22

Consider the system

$$y(n) - 2y(n-1) = x(n)$$

Represent this system in block diagram form.

Solution

This is a first-order system where only $y(n)$ is delayed. The block is shown in Figure 2.10.

$$y(n) = 2y(n-1) + x(n) = 2Dy(n) + x(n)$$

The direction of the arrows is important. It means that the signals are flowing in the indicated direction.

Example 2.23

Consider the system

$$y(n) - 5y(n-1) - 3y(n-2) = x(n-1)$$

Give the block diagram representation.

Solution

We rewrite the output as

$$y(n) = 5y(n-1) + 3y(n-2) + x(n-1)$$

In terms of the D operator we have

$$y(n) = 5Dy(n) + 3D^2 y(n) + Dx(n)$$

The representation is shown in Figure 2.11.

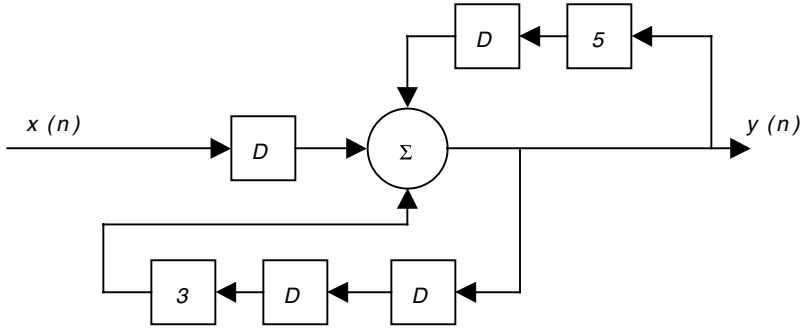

FIGURE 2.11 Block diagram for Example 2.23.

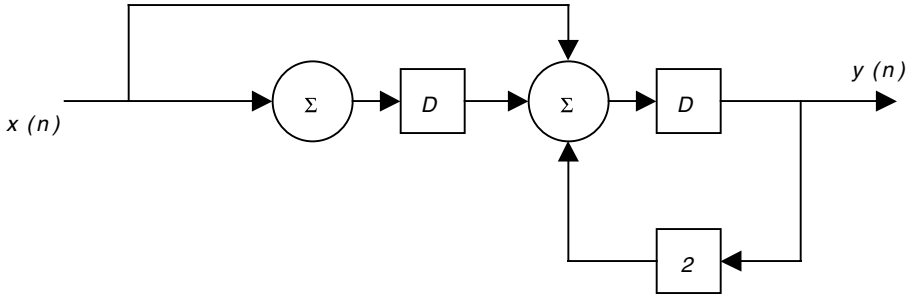

FIGURE 2.12 Block diagram for Example 2.24.

2.15 From the Block Diagram to the Difference Equation

This is best illustrated by examples. The rule is that you find signals at the junctions proceeding from the input side going right to the output side.

Example 2.24

Consider the block diagram in Figure 2.12. Find the difference equation represented in the diagram.

Solution

Looking at the output of the first summing junction from the left we have the signal $x(n)$. Looking at the output of the second summing junction, we have the signal $x(n) + x(n - 1) + 2y(n)$. The output of the last summing junction is $y(n)$. Coming to this last junction is the signal $x(n - 1) + x(n - 2) +$

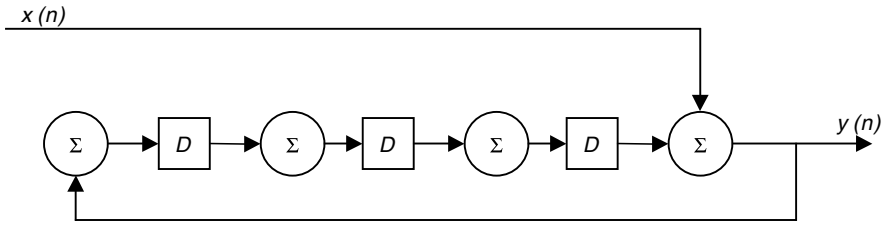

FIGURE 2.13 Block diagram for Example 2.25.

$2y(n-1)$. Thus the output of the last junction is $y(n) = 2y(n-1) + x(n-1) + x(n-2)$. The difference equation is then

$$y(n) - 2y(n-1) = x(n-1) + x(n-2)$$

Example 2.25

Consider the system in Figure 2.13. What is the difference equation?

Solution

The output of the first summer is $y(n)$. The output of the second summer is $y(n-1)$. The output of the third summer is $y(n-2)$. The output of the forth summer is $y(n)$, which is $y(n-3) + x(n)$. Therefore, the system representation as a difference equation is

$$y(n) - y(n-3) = x(n)$$

2.16 From the Difference Equation to the Block Diagram: A Formal Procedure

This procedure will also be discussed using examples.

Example 2.26

Consider the system

$$y(n) - 2y(n-1) = x(n) + x(n-1)$$

Draw the block diagram.

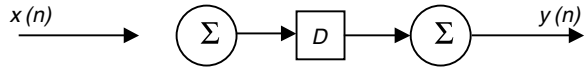

FIGURE 2.14 Block diagram for Example 2.26.

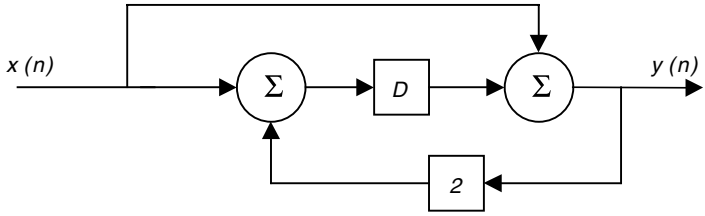

FIGURE 2.15 Block diagram for Example 2.26.

Solution

1. The given system is first order in $y(n)$. Therefore, we will need only one delay element.
2. We initially draw the diagram in Figure 2.14 where we have one delay element preceded and followed by a summing junction. The input and output lines are drawn with $x(n)$ line not connected.
3. In the given equation, solve for $y(n)$ to get

$$y(n) = 2y(n-1) + x(n) + x(n-1)$$

Let us represent a delay by D, two delays by D^2 and so on to get

$$y(n) = 2Dy(n) + x(n) + Dx(n)$$

$$y(n) = D\left[2y(n) + x(n)\right] + x(n)$$

We will feed $(2y(n) + x(n))$ to the summer before the delay and $x(n)$ to the summer following the delay as shown in the final diagram in Figure 2.15

Example 2.27

Consider the system

$$y(n) - 0.5y(n-1) - 0.3y(n-2) = 3x(n) + x(n-1)$$

Draw the block diagram.

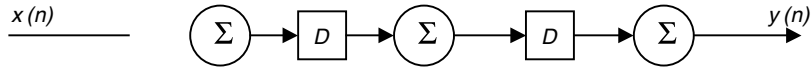

FIGURE 2.16 Block diagram for Example 2.27.

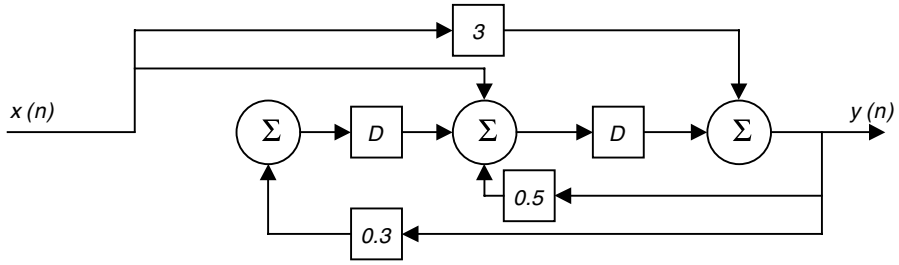

FIGURE 2.17 Block diagram for Example 2.27.

Solution

1. Our system in second order in $y(n)$, so we need to delay elements.
2. Draw the initial block diagram as in Figure 2.16 where every delay is preceded and followed by a summing junction and $x(n)$ is hanging and not connected.
3. Solve for $y(n)$ as

$$y(n) = 0.5y(n-1) + 0.3y(n-2) + 3x(n) + x(n-1)$$

Represent each delay by D and so on to get

$$y(n) = 0.5Dy(n) + 0.3D^2y(n) + 3x(n) + Dx(n)$$

$$y(n) = D^2\left[0.3y(n)\right] + D\left[0.5y(n) + x(n)\right] + 3x(n)$$

Feed $0.3y(n)$ to the summer before the first delay, $(0.5y(n) + x(n))$ to the summer before the second delay, and $3x(n)$ to the summer following the last delay to get the block diagram in Figure 2.17.

Example 2.28

Consider the system

$$y(n) - 3y(n-3) = x(n-3)$$

Draw the block diagram.

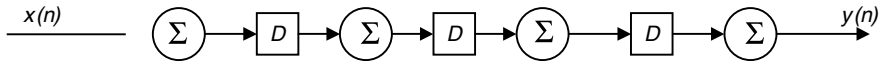

FIGURE 2.18 Block diagram for Example 2.28.

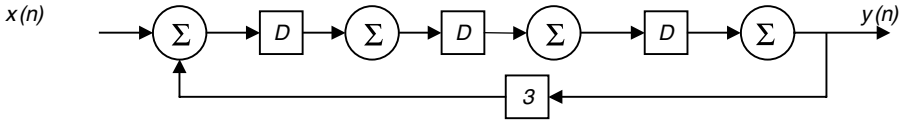

FIGURE 2.19 Block diagram for Example 2.28.

Solution

1. The system is third order in $y(n)$ and we will need three delay elements.
2. We initially draw the block as shown in Figure 2.18 with each delay preceded and followed by a summer and the $x(n)$ input line hanging.
3. Next we solve for $y(n)$ as

$$y(n) = 3D^3y(n) + D^3x(n) = \left[3y(n) + x(n)\right]D^3 + [0]D^2 + [0]D + [0]$$

$3y(n) + x(n)$ is fed to the summer preceding the first delay, 0 to the summer preceding the third delay, and 0 to the summer following the last delay as shown in Figure 2.19. Note that 0 means nothing is connected.

2.17 The Impulse Response

The impulse response is the response due to an impulsive input. We will call the output $y(n)$, $h(n)$ when the input $x(n)$ is $\delta(n)$.

Example 2.29

Find the impulse responses for the system

$$y(n) - .5y(n-1) = x(n)$$

Solution

The output will be $h(n)$ if $x(n) = \delta(n)$. The difference equation becomes

$$h(n) - .5h(n-1) = \delta(n)$$

This is a nonhomogeneous difference equation in $h(n)$ and has a solution that contains two parts; $h_h(n)$ and $h_p(n)$. For $h_h(n)$, the character equation is $p - .5 = 0$ and $p = 5$. So

$$h_h(n) = c_1(.5)^n$$

For the particular solution we look at Table 2.1 and find that if the input is an impulse, the particular solution is zero. Thus $h_p(n) = 0$. Therefore,

$$h(n) = h_h(n) + h_p(n) = c_1(.5)^n$$

Next we find c_1. From the above equation we have

$$h(0) = c_1(.5)^0 = c_1$$

But we do not know the values for h(0). From the given system with $x(n) = \delta(n)$ we have

$$h(0) = .5h(-1) + \delta(0) = 0 + 1 = 1$$

Next we equate the $h(0)$ values to get

$$h(0) = 1 = h(0) = c_1$$

And finally

$$h(n) = (.5)^n \text{ for } n \geq 0$$

Example 2.30

Consider the system

$$y(n) + 5y(n-1) + 6y(n-2) = x(n)$$

What is the impulse response?

Solution

The characteristic equation is

$$p^2 + 5p + 6 = 0$$

which gives $p_1 = -2$ and $p_2 = -3$.

Since $x(n) = \delta(n)$, the particular solution for $h(n)$ is $h_p(n) = 0$ as seen in Table 2.1. Thus the total solution is

$$h(n) = c_1(-2)^n + c_2(-3)^n \qquad n \geq 0$$

To find c_1 and c_2 we substitute $n = 0$ and $n = 1$ in the above solution for $h(n)$ to get

$$h(0) = c_1 + c_2$$

$$h(1) = -2c_1 - 3c_2$$

But we do not know the values of $h(0)$ and $h(1)$. However, from the given system with $x(n) = \delta(n)$ and $n = 0$ we get

$$h(0) + 5h(-1) + 6h(-1) = \delta(0) = 1$$

with $h(0) = 1$.

And with $n = 1$ we have

$$h(1) + 5h(0) + 6h(-1) = 0$$

with $h(1) = -5$. Thus we will end up with the two algebraic equations obtained by equating values for $h(0)$ and $h(1)$. The equations are

$$h(0) = 1 = c_1 + c_2$$

$$h(1) = -5 = -2c_1 - 3c_2$$

Solving the two equations gives $c_1 = -2$ and $c_2 = 3$ and the impulse response is then

$$h(n) = -2(-2)^n + 3(-3)^n \qquad n \geq 0$$

2.18 Correlation

Correlation between two finite duration signals $x_1(n)$ and $x_2(n)$ is referred to as cross-correlation while correlation between the finite signal $x(n)$ and itself is referred to as auto-correlation. Next we explain both.

2.18.1 Cross-Correlation

The cross-correlation between the two signals $x_1(n)$ and $x_2(n)$ is an indication of the similarities between the two signals as a function of the delay between

them. The cross-correlation between $x_1(n)$ and $x_2(n)$ is written mathematically as

$$R_{x_1x_2}(k) = \sum_{n=-\infty}^{\infty} x_1(n)x_2(n-k) \tag{2.12}$$

The index k is known as the lag of $x_1(n)$ relative to $x_2(n)$. If we let $n - k = m$, then the cross-correlation equation becomes

$$R_{x_1x_2}(k) = \sum_{m=-\infty}^{\infty} x_1(m+k)x_2(m) = \sum_{n=-\infty}^{\infty} x_1(n+k)x_2(n) \tag{2.13}$$

This shows that we can use two equations to evaluate cross-correlation. The question whether $R_{x_1x_2}(k)$ is the same as $R_{x_2x_1}(k)$ is examined next. According to the formula for cross-correlation we have

$$R_{x_2x_1}(k) = \sum_{n=-\infty}^{\infty} x_2(n)x_1(n-k)$$

But

$$R_{x_1x_2}(k) = \sum_{n=-\infty}^{\infty} x_1(n+k)x_2(n)$$

and

$$R_{x_1x_2}(-k) = \sum_{n=-\infty}^{\infty} x_1(n)x_2(n+k) = \sum_{n=-\infty}^{\infty} x_1(n-k)x_2(n)$$

Thus we see that

$$R_{x_2x_1}(k) = R_{x_1x_2}(-k) \tag{2.14}$$

The above equations for cross-correlation are defined for energy signals where the summation converges to some constant. If the signals are power signals (periodic signals are examples of this type) then the summation will not converge and thus we will use average values. In this case if the period

of the discrete sequence is N then the cross-correlation is taken over one period (which is the same as averaging over an infinite interval) and is defined as

$$R_{x_1x_2}(k) = \frac{1}{N}\sum_{n=0}^{N} x_1(n)x_2(n-k) \tag{2.15}$$

We have also seen before that the convolution between the two signals $x(n)$ and $h(n)$ is given by

$$y(k) = \sum_{n=-\infty}^{\infty} x(n)h(k-n)$$

and the convolution between $x(n)$ and $h(-n)$ is

$$y(k) = \sum_{n=-\infty}^{\infty} x(n)h(k+n) \tag{2.16}$$

But the quantity to the right of the equal sign in Equation (2.16) is nothing but the cross-correlation between $x(n)$ and $h(n)$. Therefore, we conclude that the convolution between $x(n)$ and $h(-n)$ is the cross-correlation between $x(n)$ and $h(n)$. We write

$$x(n) * h(-n) = R_{xh}(k) = \sum_{n=-\infty}^{\infty} x(n)h(k+n) \tag{2.17}$$

2.18.2 Auto-Correlation

Auto-correlation is defined between the signal and itself. It is defined for energy signals as

$$R_{xx}(k) = \sum_{n=-\infty}^{\infty} x(n)x(n-k) \tag{2.18a}$$

and for power signals of period N it is defined as

$$R_{xx}(k) = \frac{1}{N}\sum_{n=0}^{N} x(n)x(n-k) \tag{2.18b}$$

2.19 Some Insights

Let us say that we have a first-order system with the output $y(n)$ given as

$$y(n) = (0.5)^n \text{ for } n > 0$$

As n approaches infinity, the output will approach the value zero. In this sense we say the output is stable for our particular input. For first-order systems (that are described by first-order difference equations) the output will have one term of the form $(a)^n$. For second-order systems, the output will have two terms similar to $(a)^n$ at the most. For third-order systems we will have three terms, and so on.

In many systems of order greater than two, and for the purpose of analysis and design, we can reduce the order of the system at hand to a second-order system due to the fast decay of some of these terms. The solution for the output for these systems is in the following form

$$y(n) = c_1\left(a_1\right)^n + c_2\left(a_2\right)^n$$

The stability of the system is determined by the values of a_1 and a_2. If any of the a's has a magnitude that is greater than 1, the output $y(n)$ will grow wild as n approaches infinity. If all the a's have a magnitude not greater than 1, then the output $y(n)$ will decay gradually and stays at a fixed value as n progresses. The a's are called the eigenvalues of the system. Therefore, we can say that a linear time-invariant system is stable if the eigenvalues of the system have magnitudes not greater than 1.

2.19.1 How Can We Find These Eigenvalues?

A linear time-invariant system can always be represented by a linear difference equation with constant coefficients

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k)$$

We can look at the auxiliary algebraic equation by setting the input, $x(n)$, to zero

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = 0$$

and letting $y(n - k) = D^k y(n)$ to get

$$y(n) - \sum_{k=1}^{N} a_k D^k y(n) = 0$$

We can expand the above equation to get

$$y(n) - a_1 D y(n) - a_2 D^2 y(n) - ... - a_N D^N y(n) = 0$$

We can factor out $y(n)$ as

$$y(n)\left[1 - a_1 D - a_2 D^2 - \ldots - a_N D^N\right] = 0$$

$y(n)$ cannot be zero (in which case the output of the system would be zero at all times) and therefore

$$\left[1 - a_1 D - a_2 D^2 - \ldots - a_N D^N\right] = 0$$

or

$$\left[1/a_N - a_1/a_N D - a_2/a_N D^2 - \ldots - D^N\right] = 0$$

This is an N^{th} order algebraic equation with N roots. Let us call them the N a's. These are the eigenvalues of the system.

2.19.2 Stability and Eigenvalues

To summarize, any linear time-invariant system can be modeled by a linear difference equation with constant coefficients. The auxiliary algebraic equation that can be obtained from the difference equation will have a number of roots called the eigenvalues of the system. The stability of the system is determined by these roots. These roots may be real or complex. If *all* the magnitudes of the roots are less than or equal to 1, then the system is stable. If *any* of the roots has a magnitude that is greater than 1, the system is unstable. The eigenvalues of the system are responsible for the shape of the output, $y(n)$. They dictate the shape of the transients of the system as well.

2.20 End of Chapter Examples

EOCE 2.1

Are the following systems linear?

1. $y(n) = (.5)^n x(n) + 1$
2. $y(n) = (.5)^n \cos(2x(n))$
3. $y(n) = \sin(n) - x(n)$

Solution

1. For an input $x_1(n)$ the output is

$$y_1(n) = (.5)^n x_1(n) + 1$$

For an input $x_2(n)$ the output is

$$y_2(n) = (.5)^n x_2(n) + 1$$

If the input is $\alpha x_1(n) + \beta x_2(n)$, then the output is

$$y(n) = (.5)^n \left[\alpha x_1(n) + \beta x_2(n)\right] + 1$$

$$= (.5)^n \alpha x_1(n) + (.5)^n \beta x_2(n) + 1$$

But $\alpha y_1(n) + \beta y_2(n)$ is

$$\alpha(.5)^n x_1(n) + \beta(.5)^n x_2(n) + 2 \neq \alpha(.5)^n x_1(n) + \beta(.5)^n x_2(n) + 1$$

Thus the system is not linear.

2. For $y = (.5)^n \cos(2x(n))$ and if the input is $x_1(n)$ then

$$y_1(n) = (.5)^n \cos\left(2x_1(n)\right)$$

If the input is $x_2(n)$, then

$$y_2(n) = (.5)^n \cos\left(2x_2(n)\right)$$

Now

$$\alpha y_1(n) + \beta y_2(n) = \alpha(.5)^n \cos\left(2x_1(n)\right) + \beta(.5)^n \cos\left(2x_2(n)\right)$$

If the input is $\alpha x_1(n) + \beta x_2(n)$, then

$$y(n) = (.5)^n \cos\left(2\alpha x_1(n) + \beta x_2(n)\right)$$

which is clearly not equal to $\alpha y_1(n) + \beta y_2(n)$. Thus the system is not linear.

3. For $y = \sin(n) - x(n)$, if the input is $x_1(n)$, then

$$y_1(n) = \sin(n) - x_1(n)$$

If the input is $x_2(n)$, then

$$y_2(n) = \sin(n) - x_2(n)$$

Now

$$\alpha y_1(n) + \beta y(n) = \alpha\big(\sin(n) - x_1(n)\big) + \beta\big(\sin(n) - x_2(n)\big)$$

If the input is $\alpha x_1(n) + \beta x_2(n)$

$$y(n) = \sin(n) - \big(\alpha x_1(n) + \beta x_2(n)\big)$$

$$= \sin(n) - \alpha x_1(n) - \beta x_2(n)$$

which is clearly not equal to $\alpha y_1(n) + \beta y_2(n)$. Thus the system is not linear.

We can use MATLAB to generate the sequence $\alpha(\sin(n) - x_1(n)) + \sigma(\sin(n) - x_2(n))$, the sequence $\sin(n) - \alpha x_1(n) - \beta x_2(n)$ and find the difference between them. If the difference is zero, then the system is linear. Otherwise it is nonlinear. We will let $\alpha = \beta = 1$ and $x_1(n) = n$ and $x_2(n) = 2n$. The MATLAB script is

```
n = 0:10; % generate only 11 samples
y1 = (sin(n)-n) + (sin(n)-2*n);
y2 = (sin(n)-n)-2*n;
y = y1 - y2; % the diff. vector of samples.
% adding all y values and taking the absolute values.
e = abs( sum (y));
```

The answer will be 1.4112. Because `e` in the above script is not zero, this proves that the system is nonlinear.

EOCE 2.2

Consider the same systems as in EOCE 2.1. Are they time-variant systems?

Solution

1. For $x_1(n)$ as an input

$$y_1(n) = (.5)^n x_1(n) + 1$$

$y_1(n)$ shifted by n_0 is

$$y_1\big(n - n_0\big) = (.5)^{n-n_0} x_1\big(n - n_0\big) + 1$$

If we apply a shifted version of $x_1(n)$, $x_1(n - n_0)$, then

$$y_2(n) = (.5)^n x_1(n - n_0) + 1$$

But $y_2(n) \neq y_1(n - n_0)$, so the system is time variant.

2. For $x_1(n)$ as an input

$$y_1(n) = (.5)^n \cos(2x_1(n))$$

$y_1(n)$ shifted by n_0 is

$$y_1(n - n_0) = (.5)^{n-n_0} \cos(2x_1(n - n_0))$$

If we apply $x_1(n - n_0)$ to the system, the output would be

$$y_2(n) = (.5)^n \cos(2x_1(n - n_0))$$

But $y_1(n - n_0) \neq y_2(n)$, so the system is time variant.

3. For $y = \sin(n) - x(n)$, if we apply $x_1(n)$ as input the output is

$$y_1(n) = \sin(n) - x_1(n)$$

$y_1(n)$ shifted by n_0 is

$$y_1(n - n_0) = \sin(n - n_0) - x_1(n - n_0)$$

If we apply $x_1(n - n_0)$ to the system, then the output is

$$y_2(n) = \sin(n) - x_1(n - n_0)$$

But $y_1(n - n_0) \neq y_2(n)$, so it is a time-variant system.

We can use MATLAB to prove these results. We will compare $y_1(n - n_0)$ to $y_2(n)$ given above. Let $x(n) = n$, and let us shift by 2 samples. The script is

```
n = 0:10;
m = n;
x1 = n;
n = n + 2;
x1 = x1 +2;
y1 = sin(n) -x1;
```

```
y2 = sin(m)-x1;
y = y1 - y2;
e = abs(sum(y)) % difference between samples
```

The result will be 2.378. This proves the analytical result.

EOCE 2.3

Consider the systems for $n \geq 0$

1. $y(n) = (.5)^n\, x(n)$
2. $y(n) = n(.5)^n\, x(n)$
3. $y(n) = (.5)^n\, x(n)$

If $x(n)$ is bounded, what about $y(n)$?

Solution

1. Since $x(n)$ is bounded we write

$$|x(n)| < \beta < \infty$$

If $(.5)^n$ is bounded then we will conclude that $y(n)$ is bounded too.

$$\sum_{n=0}^{N} (.5)^n = \frac{1-(.5)^{N+1}}{1-.5}$$

as $N \to \infty$, $(.5)^{N+1}$ goes to zero and

$$\sum_{n=0}^{\infty} (.5)^n = \frac{1}{1-.5} = 2$$

Thus $y(n)$ is also bounded.

2. Since $x(n)$ is bounded, we will check the signal $n(.5)^n$. As n approaches ∞, $(.5)^n$ will approach zero. The multiplicative term $n(.5)^n$ will die as n approaches infinity, so the system output is bounded.
3. Because $x(n)$ is known to be bounded, we will check the sum $(5)^n$ for $n \geq 0$.

$$\sum_{n=0}^{N} (5)^n = \frac{1-(5)^{N+1}}{1-5}$$

As N approaches infinity, $(5)^{n+1}$ will grow without bounds. Thus

$$\sum_{n=0}^{N}(5)^n$$

will not converge to any value. Hence the system output is not bounded.

EOCE 2.4

Consider the following finite duration signals.

1. $x_1(n) = \{\underset{\uparrow}{1}, 1, 1\}$ $x_2(n) = \{\underset{\uparrow}{1}, 1, 1\}$
2. $x_1(n) = \{\underset{\uparrow}{1}, 1, 1\}$ $x_2(n) = \{1, 1, \underset{\uparrow}{1}\}$
3. $x_1(n) = \{\underset{\uparrow}{1}, 1, 1\}$ $x_2(n) = \delta(n) = \{\underset{\uparrow}{1}\}$

Find $x_1(n) * x_2(n)$, the convolution result, for the above cases.

Solution

For two finite signals $x_1(n)$ and $x_2(n)$, and if $x_1(n)$ is defined on the interval $sn_1 < n_1 < en_1$ and $x_2(n)$ is defined on the interval $sn_2 < n_2 < en_2$, then $y(n) = x_1(n) * x_2(n)$ will start at the index $sn_1 + sn_2$ and ends at the index $en_1 + en_2$.

1. $y(n) = \sum_{0}^{2} x_1(m)x_2(n-m)$.

In this case, $x_1(n)$ is defined on $0 < n_1 < 2$ and $x_2(n)$ is defined on the interval $0 < n_2 < 2$. $y(n)$ will start at $0 + 0 = 0$ and ends at $2 + 2 = 4$. So we write

$$y(0) = x_1(0)x_2(0) + x_1(1)x_2(-1) + x_1(2)x_2(-2) = 1 + 0 + 0 = 1$$

$$y(1) = x_1(0)x_2(1) + x_1(1)x_2(0) + x_1(2)x_2(-1) = 1 + 1 + 0 = 2$$

$$y(2) = x_1(0)x_2(2) + x_1(1)x_2(1) + x_1(2)x_2(0) = 1 + 1 + 1 = 3$$

$$y(3) = x_1(0)x_2(3) + x_1(1)x_2(2) + x_1(2)x_2(1) = 0 + 1 + 1 = 2$$

$$y(4) = x_1(0)x_2(4) + x_1(1)x_2(3) + x_1(2)x_2(2) = 0 + 0 + 1 = 1$$

Therefore, the convolution result is

$$y(n) = x_1(n) * x_2(n) = \left\{\underset{\uparrow}{1}, 2, 3, 2, 1\right\}$$

We can use MATLAB to find $y(n) = x_1(n) * x_2(n)$ if both signals start at $n = 0$. Since the signals here both start at $n = 0$, we can use MATLAB and write the script

```
n=0:2;
x1 = [ 1 1 1];
x2 = [ 1 1 1];
y = conv(x1, x2);
subplot(1,3,1)
stem(n,x1);xlabel('n');ylabel('The first signal');
subplot(1,3,2);stem(n,x2);xlabel('n');ylabel('The second
 signal');
n=0:4;
subplot(1,3,3);stem(n,y);
xlabel('n');ylabel('The result of the convolution');
```

to get the same result as shown in Figure 2.20.

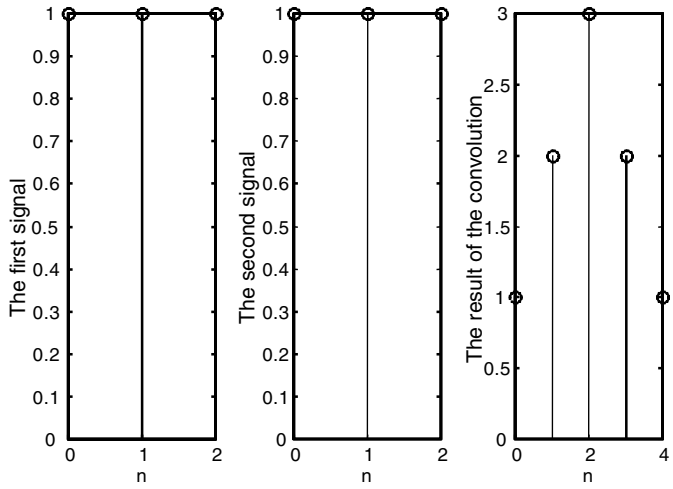

FIGURE 2.20 Signals for EOCE 2.4.

2. In this case $x_1(n)$ starts at $n = 0$ and ends at $n = 2$, and $x_2(n)$ starts at $n = -2$ and ends at $n = 0$. $y(n) = x_1(n)^* x_2(n)$ will start at $0 + (-2) = -2$ and ends at $n = 2 - 2 = 0$. So we write

$$y(n) = \sum_{m=0}^{2} x_1(m)x_2(n-m)$$

and

$$y(-2) = x_1(0)x_2(-2-0) + x_1(1)x_2(-2-1) + x_1(2)x_2(-2-2) = (1)(1) + (1)(0) + (1)(0) = 1$$

$$y(-1) = x_1(0)x_2(-1-0) + x_1(1)x_2(-1-1) + x_1(2)x_2(-1-2) = (1)(1) + (1)(1) + (1)(0) = 2$$

$$y(0) = x_1(0)x_2(0-0) + x_1(1)x_2(0-1) + x_1(2)x_2(0-2) = (1)(1) + (1)(1) + (1)(1) = 3$$

$$y(1) = x_1(0)x_2(1-0) + x_1(1)x_2(1-1) + x_1(2)x_2(1-2) = (1)(0) + (1)(1) + (1)(1) = 2$$

$$y(2) = x_1(0)x_2(2-0) + x_1(1)x_2(2-1) + x_1(2)x_2(2-2) = (1)(0) + (1)(0) + (1)(1) = 1$$

Therefore, we have

$$y(n) = x_1(n) * x_2(n) = \left\{1, 2, \underset{\uparrow}{3}, 2, 1\right\}$$

We can use MATLAB to find $y(n)$, but, since $x_2(n)$ does not start at $n = 0$, we will do some modification before we can use the `conv` function from MATLAB. The following MATLAB script will do that.

```
x1 = [ 1 1 1]; x2 = [ 1 1 1];
n1=[0 1 2];
n2=[-2 -1 0];
sn1 = 0; % starting index for x1
sn2 = -2; % ending for x1
en1= 2; % starting for x2
en2 = 0; % ending for x2
sn = sn1 + sn2; % starting for y(n)
en = en1 + en2; % ending for y(n)
n = sn : en ; % index for y(n).
y = conv(x1, x2);
subplot(1,3,1)
stem(n1,x1);xlabel('n');ylabel('The first signal');
subplot(1,3,2);stem(n2,x2);xlabel('n');ylabel('The
 second signal');
subplot(1,3,3);stem(n,y);
xlabel('n');ylabel('The result of the convolution');
```

The plot is shown in Figure 2.21.

3. The starting index of $x_1(n)$ is $sn_1 = 0$ and the ending index is $en_1 = 2$. The starting index of $x_2(n)$ is $sn_2 = 0$ and the ending index is $en_2 = 0$. Therefore, $y(n)$ will start at $ns = sn_1 + sn_2 = 0 + 0 = 0$ and will end at $en = en_1 + en_2 = 2$.

$$y(n) = \sum_{m=0}^{2} x_1(m)x_2(n-m)$$

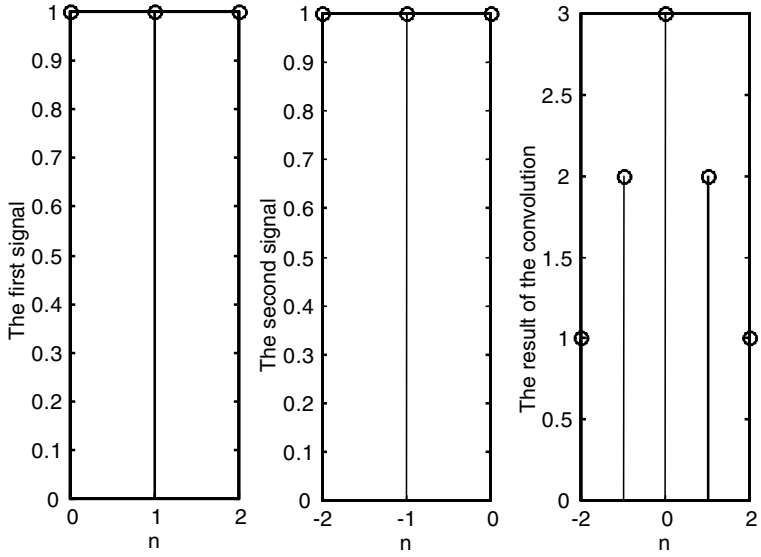

FIGURE 2.21 Signals for EOCE 2.4.

$$y(0) = x_1(0)x_2(0-0) + x_1(1)x_2(0-1) + x_1(2)x_2(0-2) = (1)(1) + (1)(0) + (1)(0) = 1$$

$$y(1) = x_1(0)x_2(1-0) + x_1(1)x_2(1-1) + x_1(2)x_2(1-2) = (1)(0) + (1)(1) + (1)(0) = 1$$

$$y(2) = x_1(0)x_2(2-0) + x_1(1)x_2(2-1) + x_1(2)x_2(2-2) = (1)(0) + (1)(0) + (1)(1) = 1$$

$$y(n) = x_1(n) * x_2(n) = \left\{ \underset{\uparrow}{1},\ 1,\ 1 \right\}$$

We can use MATLAB here directly since both signals start at $n = 0$ and write

```
n1=[0 1 2];
n2=[0];
x1 = [ 1 1 1];
x2 = [ 1 ];
y = conv(x1, x2);
subplot(1,3,1)
stem(n1,x1);xlabel('n');ylabel('The first signal');
subplot(1,3,2);stem(n2,x2);xlabel('n');
ylabel('The second signal');
n=0:2;
subplot(1,3,3);stem(n,y);
xlabel('n');ylabel('The result of the convolution');
```

The result is shown in Figure 2.22.

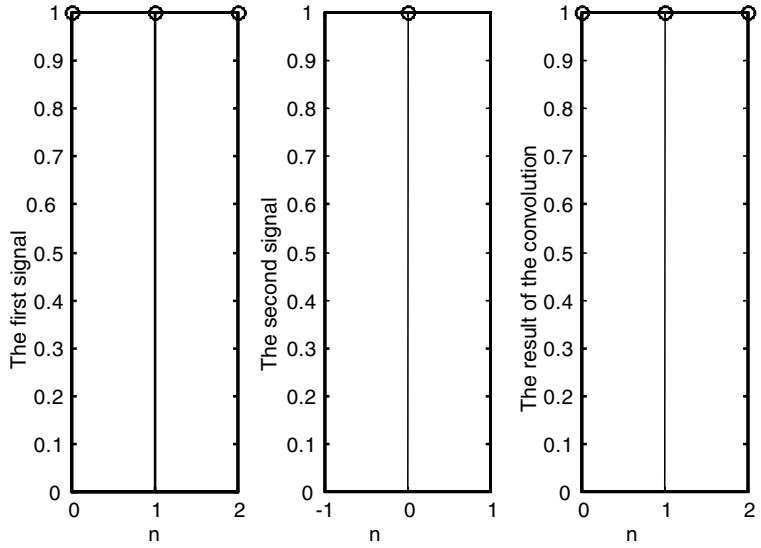

FIGURE 2.22 Signals for EOCE 2.4.

EOCE 2.5

Consider the following system represented by the impulse response $h(n)$.

$$h(n) = (.5)^n u(n)$$

with

1. $x(n) = u(n)$
2. $x(n) = (.5)^n u(n)$

Find $y(n) = x(n) * h(n)$ for both cases.

Solution

1. For the first case

$$y(n) = \sum_{m=-\infty}^{+\infty} u(m)(.5)^{n-m} u(n-m)$$

Since both signals start at $n = 0$, $y(n)$ will start at $n = 0$.

$$y(n) = \sum_{m=0}^{m=n} (.5)^{n-m} = (.5)^n \sum_{m=0}^{m=n} \left(.5^{-1}\right)^m = (.5)^n \left[\frac{1 - \left((.5)^{-1}\right)^{n+1}}{1 - (.5)^{-1}} \right] \qquad n \geq 0$$

After simplification we get

$$y(n) = 2 - (.5)^n \qquad n \geq 0$$

2. For the second case

$$y(n) = \sum_{m=-\infty}^{+\infty} u(m)(.5)^m (.5)^{n-m} u(n-m)$$

Since both $x(n)$ and $h(n)$ in this case start at $n = 0$, we have

$$y(n) = \sum_{m=0}^{m=n} (.5)^m (.5)^n (.5)^{-m} = \sum_{m=0}^{n} (.5)^n = (.5)^n \sum_{m=0}^{n} 1$$

And after simplifying we arrive at

$$y(n) = n(.5)^n \qquad n \geq 0$$

EOCE 2.6

Consider the following homogeneous difference equations

1. $y(n) + .6y(n-1) = 0$ with $y(-1) = 1$
2. $y(n) + y(n-1) + (1/4)\, y(n-2) = 0$ with $y(-1) = y(-2) = 0$

a) Find the characteristic equation for the above systems and see if they are stable.
b) Find the homogeneous solution for both. The homogeneous solution is the solution due to the initial conditions.

Solution

For the first system the characteristic equation is

$$p + .6 = 0 \text{ or } p = -.6$$

Since $0 < .6 < 1$, the system is stable. The homogeneous solution is

$$y_{\mathrm{h}}(n) = c_1(-.6)^n \text{ for } n \geq 0$$

With $y(-1) = 1$, we have

$$1 = c_1(-.6)^{-1}$$

$$c_1 = -.6$$

Thus the solution is

$$y_h(n) = -0.6(-.6)^n \text{ for } n \geq 0$$

For the second system the characteristic equation is

$$p^2 + p + \frac{1}{4} = 0$$

By factoring we get

$$\left(p + \frac{1}{2}\right)\left(p + \frac{1}{2}\right) = 0$$

$$p_1 = p_2 = -\frac{1}{2}$$

Since $0 < .5 < 1$, the system is stable. The homogeneous solution is then

$$y_h(n) = c_1\left(-\frac{1}{2}\right)^n + nc_2\left(-\frac{1}{2}\right)^n$$

With the initial conditions given we can write

$$y(-1) = 1 = -2c_1 + 2c_2$$

$$y(-2) = 0 = 4c_1 - 8c_2$$

Solving these equations we arrive at $c_2 = -(1/2)$ and $c_1 = -1$. The final solution is then

$$y_n(n) = -\left(-\frac{1}{2}\right)^n - \frac{1}{2}\left(-\frac{1}{2}\right)^n n \qquad n \geq 0$$

EOCE 2.7

Consider the following difference equations

1. $y(n)+.6y(n-1)=u(n) \qquad y(-1)=1$
2. $y(n)+y(n-1)+\frac{1}{4}y(n-2)=\delta(n) \qquad y(-1)=1; y(-2)=0$

Find the total solutions for the two systems.

Solution

For the first system, the characteristic equation is $p + .6 = 0$ or $p = -.6$ and the homogeneous part of the solution is

$$y_h(n)=c_1(-.6)^n$$

The particular solution is taken to be

$$y_p(n)=c_2$$

This particular solution is taken from Table 2.1 with $x(n) = u(n)$, a constant. We will substitute the particular solution into the difference equation given. We will get

$$c_2 + .6c_2 = 1$$

and this gives us $c_2 = \frac{1}{1.6}$. The total solution now is

$$y(n)=y_h(n)+y_p(n)=c_1(-.6)^n+\frac{1}{1.6}$$

To find c_1 we use the initial condition given to us and write

$$y(-1)=1=c_1\left(\frac{1}{-.6}\right)+\frac{1}{1.6}$$

$$c_1=\frac{6}{16}-.6$$

and finally the solution is

$$y(n)=\left(\frac{6}{16}-.6\right)(-.6)^n+\frac{1}{1.6} \qquad n\geq 0$$

For the second system the characteristic equation is

$$p^2 + p + \frac{1}{4} = 0 \text{ with } p_1 = p_2 = -\frac{1}{2}$$

The homogeneous solution is

$$y_h(n) = c_1\left(-\frac{1}{2}\right)^n + c_2 n\left(-\frac{1}{2}\right)^n$$

The particular solution is $y_p(n) = 0$ and is obtained from Table 2.1. The total solution is

$$y(n) = c_1\left(-\frac{1}{2}\right)^n + nc_2\left(-\frac{1}{2}\right)^n \qquad n \geq 0$$

Next we find the constants c_1 and c_2 using the given initial conditions.

$$y(-1) = +1 = -2c_1 + 2c_2$$
$$y(-2) = 0 = 4c_1 - 8c_2$$

Solving the above equation gives $c_1 = -1$ and $c_2 = -\frac{1}{2}$, and the total solution is then

$$y(n) = -\left(-\frac{1}{2}\right)^n - \frac{1}{2}n\left(-\frac{1}{2}\right)^n \qquad n \geq 0$$

EOCE 2.8

Consider the system

$$y(n) + y(n-1) = x(n)$$

For $x(n) = u(n)$, find $y(n)$ due to the input $x(n)$ using the convolution summation equation.

Solution

We will first find $h(n)$ and then find $y(n)$ using the convolution equation $y(n) = h(n) * x(n)$. To find $h(n)$ for the system, we set $x(n)$ equal to $\delta(n)$. In this case, with a characteristic equation of $p + 1 = 0$, and a particular solution of zero as seen in Table 2.1, we have

$$h(n) = c_1(-1)^n \qquad n \geq 0$$

To find c_1, we have from the solution for $h(n)$ that $h(0) = c_1$. Also we can substitute in the given difference equation with $x(n) = \delta(n)$ to find $h(0)$. We will have

$$h(0) + h(-1) = \delta(0)$$

But $h(-1) = 0$ since the output starts at $n = 0$. Thus

$$h(0) = \delta(0) = 1$$

We then equate $h(0)$ obtained from the solution and $h(0)$ obtained from the given system to get

$$h(0) = c_1 = h(0) = 1$$

$$c_1 = 1$$

And the solution for the impulse response is finally

$$h(n) = (-1)^n \qquad n \geq 0$$

Once we have $h(n)$ we can find $y(n)$ using the convolution equation

$$y(n) = x(n) * h(n) = \sum_{m=-\infty}^{+\infty} u(m)(-1)^{n-m}u(n-m) = \sum_{m=0}^{n}(-1)^{n-m} = (-1)^n \sum_{m=0}^{m=n}\left[(-1)^{-1}\right]^m$$

By using the geometric series summation we simplify to get

$$y(n) = (-1)^n\left[\frac{1-(-1)^{n+1}}{1-(-1)}\right] = \frac{(-1)^n}{2}\left[1-(-1)^{n+1}\right] = \frac{(-1)^n - (-1)^{2n+1}}{2} \qquad n \geq 0$$

EOCE 2.9

Consider the following systems

1. $y(n) + .6y(n-1) = x(n)$
2. $y(n) + y(n-1) + .1y(n-2) = x(n) - x(n-1)$
3. $y(n) + y(n-1) + .1y(n-2) + 8y(n-3) = x(n)$
4. $y(n) - .8y(n-4) = x(n)$

For all systems, use MATLAB to find the impulse responses in the range $-10 \leq n \leq 20$, the step responses and the sinusoidal responses for $x(n) = .5\sin(n)u(n)$.

Solution

As we know, the general difference equation is given as

$$y(n) = \sum_{m=0}^{M} b_m x(n-m) - \sum_{k=1}^{N} a_k y(n-k)$$

In MATLAB, the function called `filter` is used to solve difference equations for a particular input $x(n)$. The function `filter` is used by typing

```
y = filter(b, a, x)
```

where `y` is the output, `b` is the input row coefficients associated with the input, `a` is the input row coefficients associated with the output y as seen in the general equation above and `x` is the given input.

For the first system

$$y(n) + 0.6y(n-1) = u(n)$$

we will enter the `b` and the `a` vectors and use the functions `stepsignal` and `inpulsesignal` defined in Chapter 1 and write the following script to find the corresponding plots.

```
a=[1 0.6];
b=[1];
x=impulsesignal(0,-10, 20);% generating the input
n=[-10:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,3,1)
stem(n,impulse);xlabel('n');title('The impulse response');
axis([-10 20 -1 1])
x=stepsignal(0,-10,20);
step=filter(b,a,x); %the output due to the step input
subplot(1,3,2)
axis([-10 20 0 1])
stem(n,step);xlabel('n');title('The step response');
x=0.5*sin(n);
axis([-10 20 -1 1])
sinusoidal=filter(b,a,x);
subplot(1,3,3);axis([-10 20 0 1])
stem(n,sinusoidal);xlabel('n');
title('The sinusoidal response');
axis([-10 20 -1 1])
```

and the plots are in Figure 2.23. You can see that the system is stable.

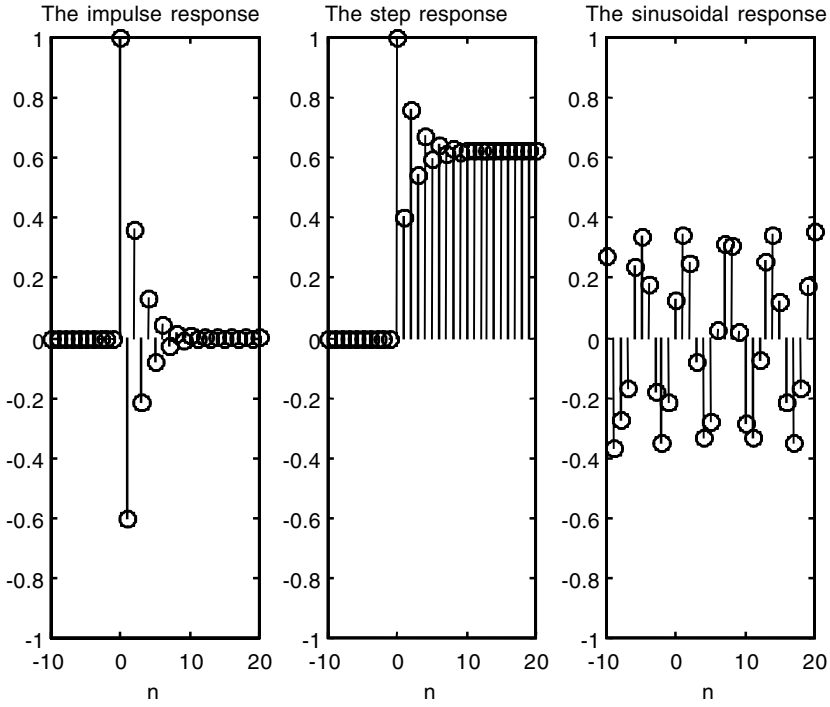

FIGURE 2.23 Signals for EOEC 2.9.

For the second system

$$y(n) + y(n-1) + .1y(n-2) = x(n)$$

we will use the same MATLAB script above with the changes

```
b = [1];
a =[ 1 1 0.1];
```

The plots are in Figure 2.24. The system is stable. We can see that using MATLAB by typing

```
Eigenvalues = roots([1 1 0.1])
```

to get

Eigenvalues =
–0.8873
–0.1127

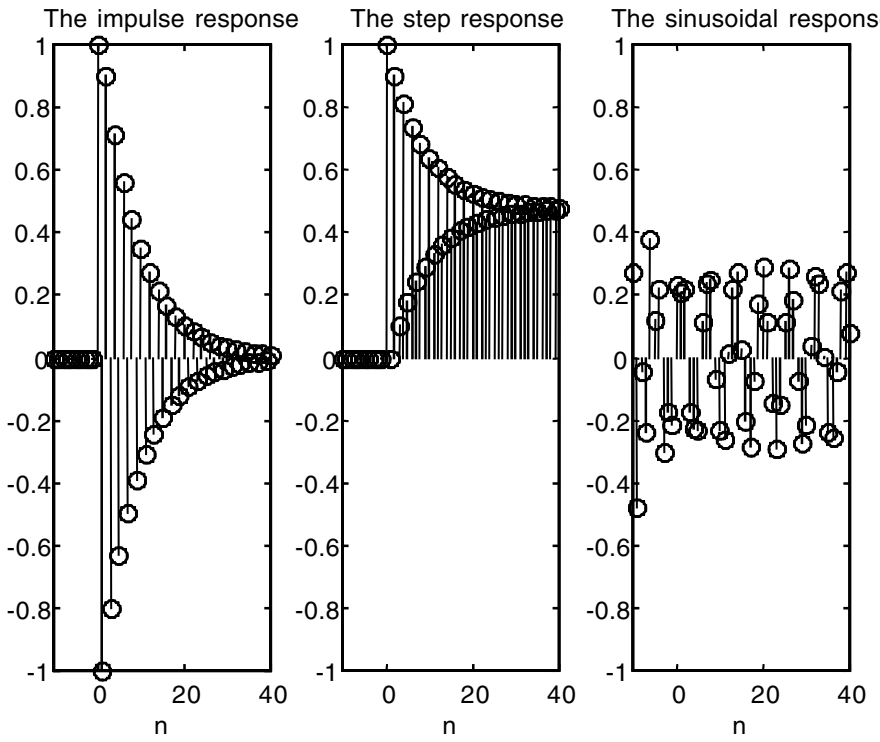

FIGURE 2.24 Signals for EOEC 2.9.

For the third system

$$y(n) + y(n-1) + .1y(n-2) + 8y(n-3) = x(n) - x(n-1)$$

we will use the above script with the changes

```
b = [ 1 -1];
a = [1 1 0.1 8];
```

to get the plots in Figure 2.25. The system is not stable. We can see that using MATLAB by typing

```
Eigenvalues = roots([1 1 0.1])
```

to get

```
Eigenvalues =
    –2.3755
     0.6878 + 1.7014i
     0.6878 – 1.7014i
```

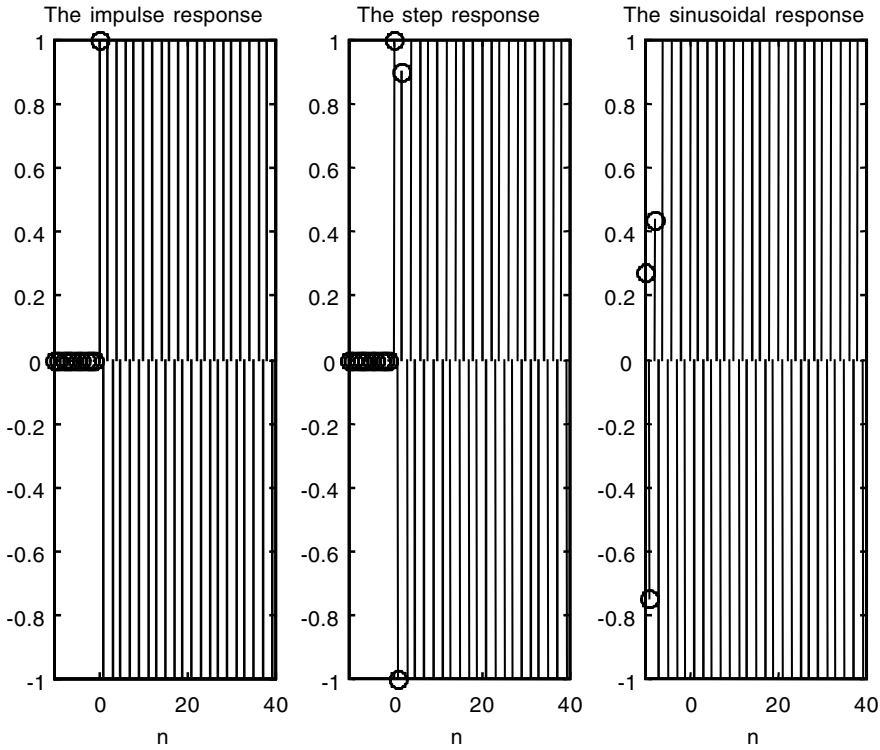

FIGURE 2.25 Signals for EOEC 2.9.

For the last system

$$y(n) - .8y(n-4) = x(n) + .1x(n-2)$$

we use

```
b = [1 0 0.1];
a = [1 0 0 0 -0.8];
```

The plots are in Figure 2.26. The system is stable. That is asserted by typing

```
Eigenvalues = roots([1 0 0 0 -.8])
```

to get

```
Eigenvalues =
    –0.9457
    –0.0000 + 0.9457i
    –0.0000 – 0.9457i
     0.9457
```

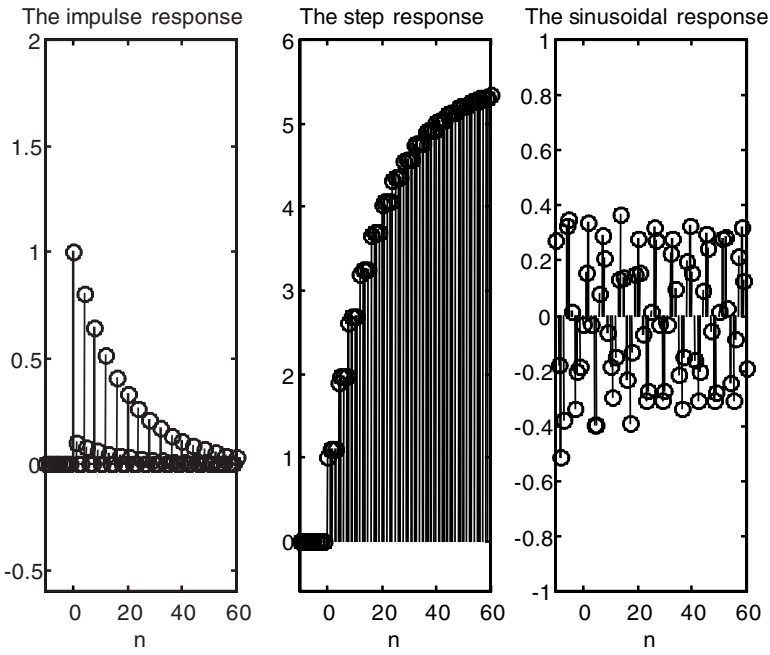

FIGURE 2.26 Signals for EOEC 2.9.

EOCE 2.10

Consider the system

$$y(n) + y(n-2) = x(n)$$

1. Find the impulse response.
2. Find the output if $x(n) = u(n)$.

Solution

1. The characteristic equation by inspection is $p^2 + 1 = 0$ with the roots $p_1 = j$ and $p_2 = -j$ or $p_1 = e^{j\pi/2}$ and $p_1 = e^{-j\pi/2}$. Thus

$$h(n) = c_1 e^{j\pi/2 n} + c_2 e^{-j\pi/2 n}$$

We must find c_1 and c_2 to completely find $h(n)$. From the solution just obtained we have

$$h(0) = c_1 + c_2$$

and

$$h(1) = c_1 e^{j\pi/2} + c_2 e^{-j\pi/2} = jc_1 - jc_2$$

We now find the same from the system given to us with the input being the impulse signal and get

$$h(0) + h(-2) = \delta(0) = 1$$

and

$$h(1) + h(-1) = \delta(1) = 0$$

We now equate the $h(0)$ and the $h(1)$ obtained from the solution we arrived at with $h(0)$ and $h(1)$ obtained from the difference equation that was given to us. We will have the two simultaneous algebraic equations

$$c_1 + c_2 = 1$$

$$jc_1 - jc_2 = 0$$

The result will be

$$c_1 = c_2 = \frac{1}{2}$$

and the impulse response is then

$$h(n) = \frac{1}{2}\left[e^{j\frac{\pi}{2}n} + e^{-j\frac{\pi}{2}n}\right] = \cos\left(\frac{\pi}{2}n\right) \qquad n \geq 0$$

2. Using convolution, the output $y(n)$ is

$$y(n) = h(n) * x(n) = \sum_{m=-\infty}^{+\infty} u(m)\cos\left(\frac{\pi}{2}n - \frac{\pi}{2}m\right)u(n-m) = \sum_{m=0}^{n} \cos\left(n\frac{\pi}{2} - m\frac{\pi}{2}\right)$$

Using trigonometric identities we arrive at

$$y(n) = \sum_{m=0}^{n} \cos\left(n\frac{\pi}{2}\right)\cos\left(m\frac{\pi}{2}\right) + \sin\left(n\frac{\pi}{2}\right)\sin\left(m\frac{\pi}{2}\right)$$

$$= \cos\left(n\frac{\pi}{2}\right)\sum_{m=0}^{n} \cos\left(m\frac{\pi}{2}\right) + \sin\left(n\frac{\pi}{2}\right)\sum_{m=0}^{n} \sin\left(m\frac{\pi}{2}\right)$$

EOCE 2.11

So far in this chapter we have learned that a discrete linear system can be represented by its block diagram, its difference equation and its impulse response $h(n)$. Given below are systems represented as difference equations. Find the other two representations.

1. $y(n) + 2y(n-1) = x(n)$
2. $y(n) - 4y(n-2) = x(n)$
3. $y(n) - .9y(n-1) + 10y(n-2) + y(n-3) = x(n) + x(n-1)$

Solution

1a. The Block Diagram Representation

Let $y(n-1) = Dy$ and $y(n-2) = D^2y$, and so on. The difference equation can be written then as

$$y(n) + 2Dy(n) = x(n)$$

Solve for $y(n)$ to get

$$y(n) = -2Dy(n) + x(n)$$

The system is first order in y and should have one delay element in the block diagram. As we did earlier we will have a summer before and after each delay, we will feed $-2y(n)$ to the summer before the delay and $x(n)$ to the summer after the delay. The block diagram is shown in Figure 2.27.

1b. The Impulse Response Representation

The impulse response representation completely describes the discrete system since, given any input we can find the output as $y = x(n) * h(n)$. We start by letting $x(n) = \delta(n)$ in the difference equation given

$$y(n) + 2y(n-1) = \delta(n)$$

The character equation is $p + 2 = 0$ or $p = -2$ is the root. Thus

$$h(n) = c_1(-2)^n \qquad n \geq 0$$

To find c_1 we have

$$h(0) = c_1(-2)^0 = c_1$$

and $h(0)$ in the given difference equation is found as

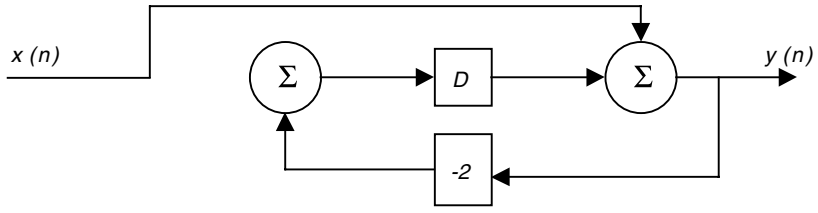

FIGURE 2.27 System for EOCE 2.11.

$$h(0) + 2h(-1) = \delta(0) = 1$$

So $h(0) = 1 = c_1$ and finally

$$h(n) = 1(-2)^n \qquad n \geq 0$$

This impulse response can be found by using MATLAB as in the following script

```
a=[1 2];
b=[1];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

The plot is in Figure 2.28.

2a. The Block Diagram Representation

The difference equations can be written as

$$y(n) = 4D^2 y(n) + x(n)$$

We need two delays in this case and the block diagram is shown in Figure 2.29.

2b. The Impulse Response Representation

The difference equation is

$$y(n) - 4y(n-2) = \delta(n)$$

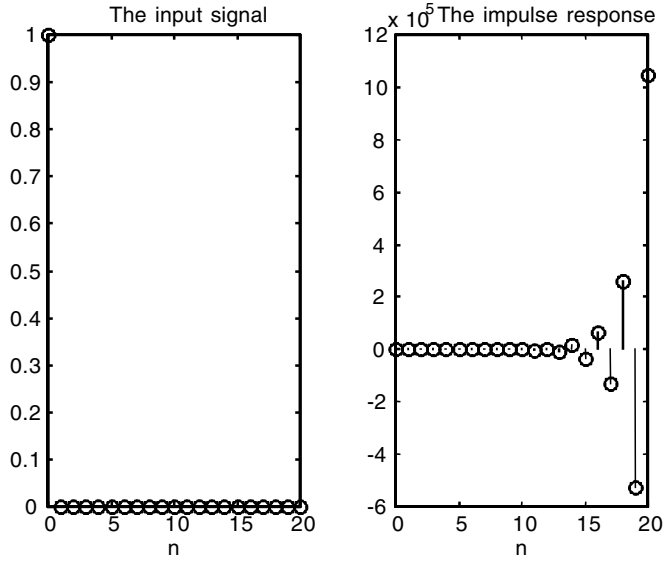

FIGURE 2.28 Signals for EOCE 2.11.

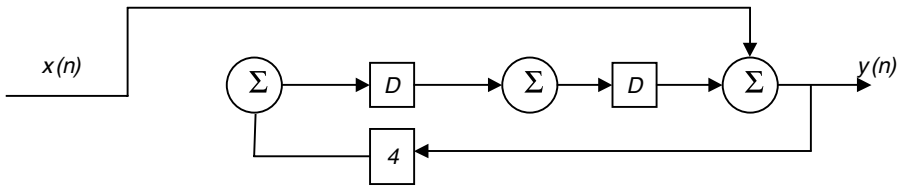

FIGURE 2.29 System for EOCE 2.11.

The character equation is

$$p^2 - 3 = 0 \text{ or } p^2 = 3$$

$$p = \pm\sqrt{4} = \pm 2$$

The impulse response is then

$$h(n) = c_1(2)^n + c_2(-2)^n$$

From this last equation for $h(n)$ we get

$$h(0) = c_1 + c_2$$

$$h(1) = 2c_1 - 2c_2$$

and so we need values for $h(0)$ and $h(1)$. We do that using the given difference equation and write

$$h(0) - 4h(-2) = \delta(0) = 1$$

and

$$h(1) - 4h(-1) = \delta(1) = 0$$

Therefore, we have now the two algebraic equations to solve

$$1 = c_1 + c_2$$
$$0 = 2c_1 - 2c_2$$

The results are

$$c_1 = c_2 = \frac{1}{2}$$

and the final solution is then

$$h(n) = \frac{1}{2}\left[(2)^n + (-2)^n\right] \qquad n \geq 0$$

$h(n)$ can also be found using MATLAB as in the following script.

```
a=[1 0 -4];
b=[1];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

The plot is shown in Figure 2.30.

3a. The Block Diagram Representation

The difference equation can be written as

$$y(n) = 0.9Dy(n) - 10D^2y(n) - D^3y(n) + x(n) + Dx(n)$$

Here we need three delay elements since the system is third order. The block diagram is shown in Figure 2.31.

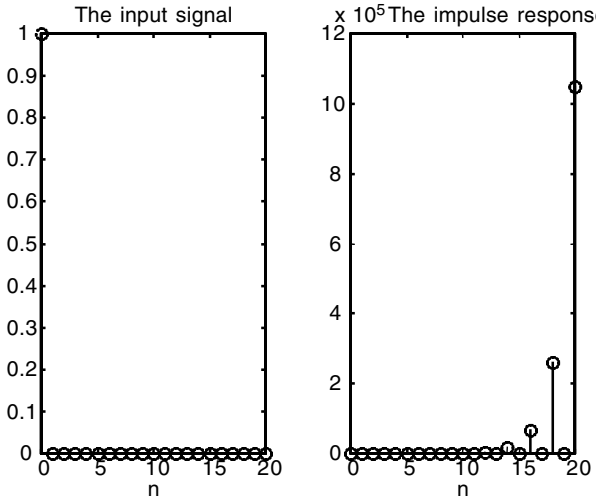

FIGURE 2.30 Signals for EOCE 2.11.

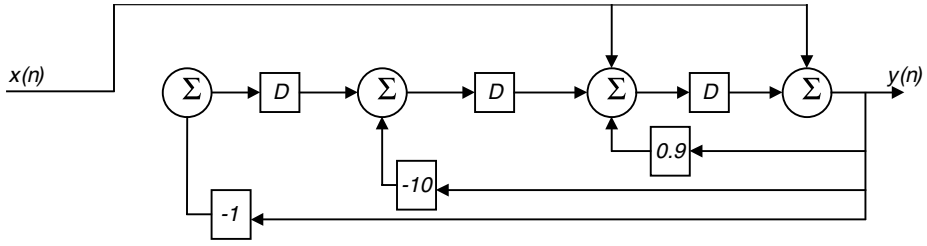

FIGURE 2.31 System for EOCE 2.11.

3b. The Impulse Response Representation

We will use MATLAB to find the impulse response as in the next script.

```
a=[1 -.9 10 1];
b=[1 1];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

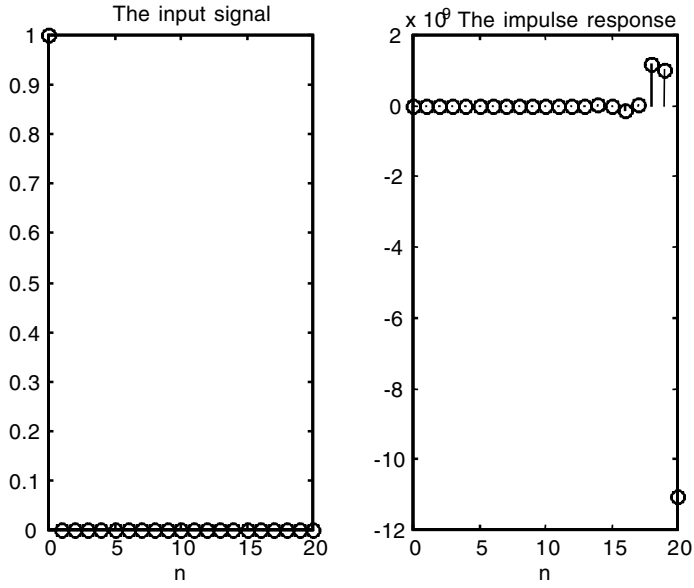

FIGURE 2.32 Signals for EOCE 2.11.

The plot is shown in Figure 2.32. The system is unstable since the magnitude of some of the roots is bigger than 1. The MATLAB command

```
EigenValues = roots([1 -.9 10 1])
```

will result in

```
EigenValues =
      0.4995 + 3.1384i
      0.4995 – 3.1384i
     –0.0990
```

EOCE 2.12

Consider the block diagrams in Figures 2.33 through 2.36. Find the other representations.

Solution

For the system in Figure 2.33 the output after the first summer is $x(n)$ + $0.1y(n)$. The output at the second summer is $x(n-1) + 0.1y(n-1) + x(n)$. The output of the second summer is also $y(n)$. Therefore,

$$y(n) = 0.1y(n-1) + x(n) + x(n-1)$$

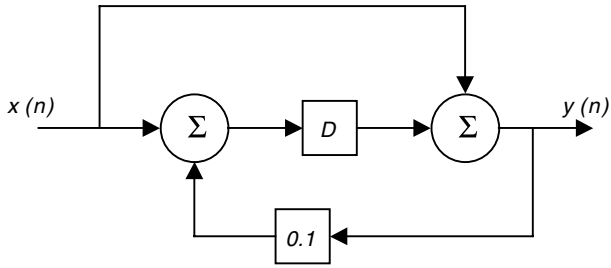

FIGURE 2.33 System for EOCE 2.12.

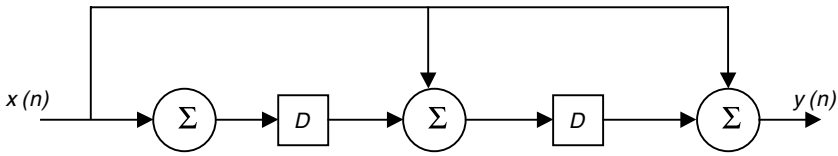

FIGURE 2.34 System for EOCE 2.12.

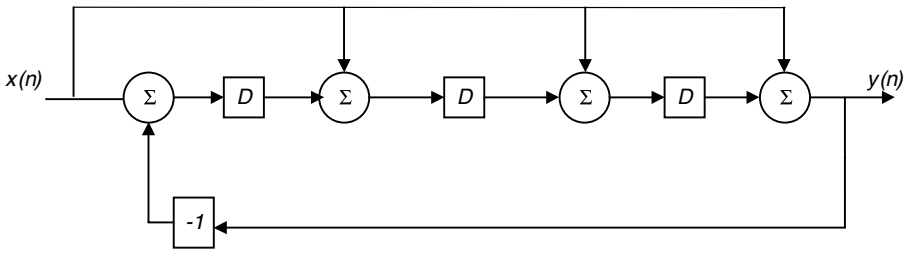

FIGURE 2.35 System for EOCE 2.12.

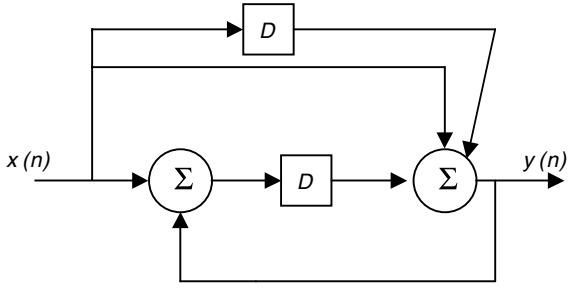

FIGURE 2.36 System for EOCE 2.12.

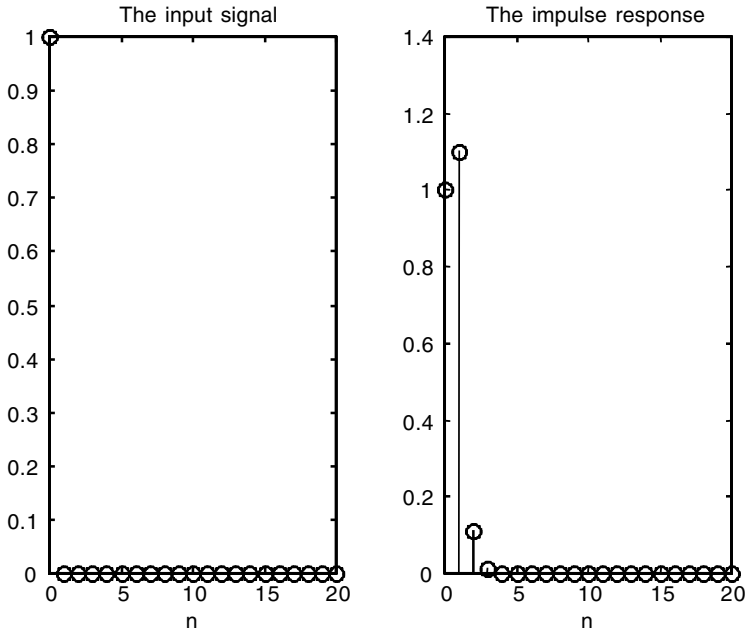

FIGURE 2.37 Signals for EOCE 2.12.

The impulse response is obtained graphically using MATLAB as in the following script.

```
a=[1 -.1];
b=[1 1];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

and the plot is shown in Figure 2.37.

For the system in Figure 2.34 the output after the first summer is $x(n)$. After the second summer the output is $x(n-1) + x(n)$. After the third summer the output is $x(n-2) + x(n-1) + x(n)$. This output is actually $y(n)$. Therefore,

$$y(n) = x(n) + x(n-1) + x(n-2)$$

To calculate the impulse responses we simply let $x(n) = \delta(n)$. Therefore,

$$h(n) = \delta(n) + \delta(n-1) + \delta(n-2)$$

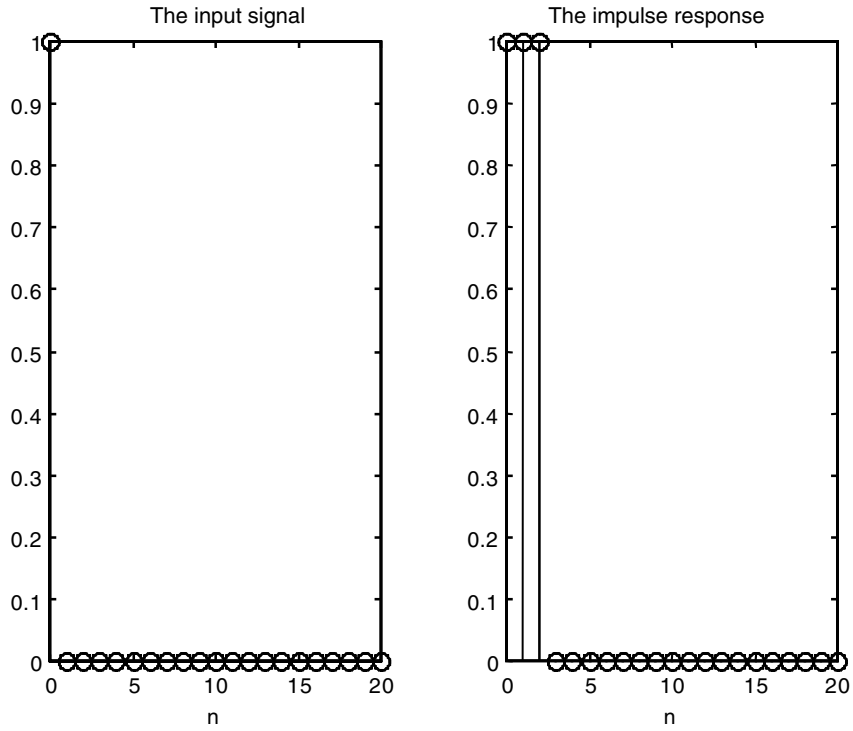

FIGURE 2.38 Signals for EOCE 2.12.

We can also use MATLAB to find the impulse response as in the following script.

```
a=[1];
b=[1 1 1];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

and the plot is shown in Figure 2.38.

For the system in Figure 2.35 the output of the first summer is $x(n) + y(n)$. At the output of the second summer we have $x(n) + x(n-1) + y(n-1)$. At the output of the third summer we have $x(n) + x(n-1) + x(n-2) + y(n-2)$. At the output of the last summer we have

$$y(n) = x(n) + x(n-1) + x(n-2) + x(n-3) + y(n-3)$$

and the difference equation is

$$y(n) - y(n-3) = x(n) + x(n-1) + x(n-2) + x(n-3)$$

The impulse response can be evaluated using MATLAB as in the following script.

```
a=[1 0 0 -1];
b=[1 1 1 1];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

The plot is shown in Figure 2.39.

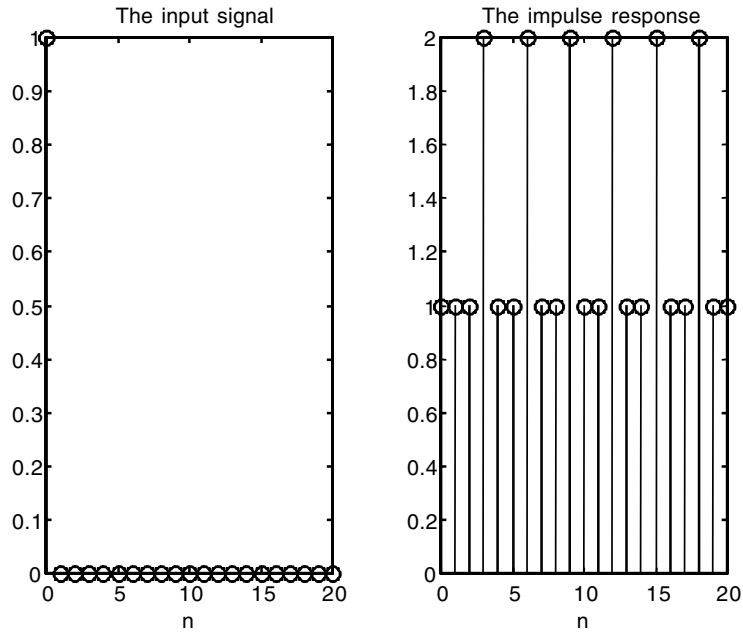

FIGURE 2.39 Signals for EOCE 2.12.

For the system in Figure 2.36 the output of the first summer is $x(n) + y(n)$. At the output of the second summer we have

$$y(n) = x(n) + 2x(n-1) + y(n-1)$$

The impulse response using MATLAB is obtained using the script

```
a=[1 -1];
b=[1 2];
x=impulsesignal(0,0,20);% generating the input
n=[0:20];
impulse=filter(b,a,x); %the output due to x(n)
subplot(1,2,1)
stem(n,x);xlabel('n');title('The input signal');
subplot(1,2,2)
stem(n,impulse);xlabel('n');title('The impulse response');
```

The plot is seen in Figure 2.40.

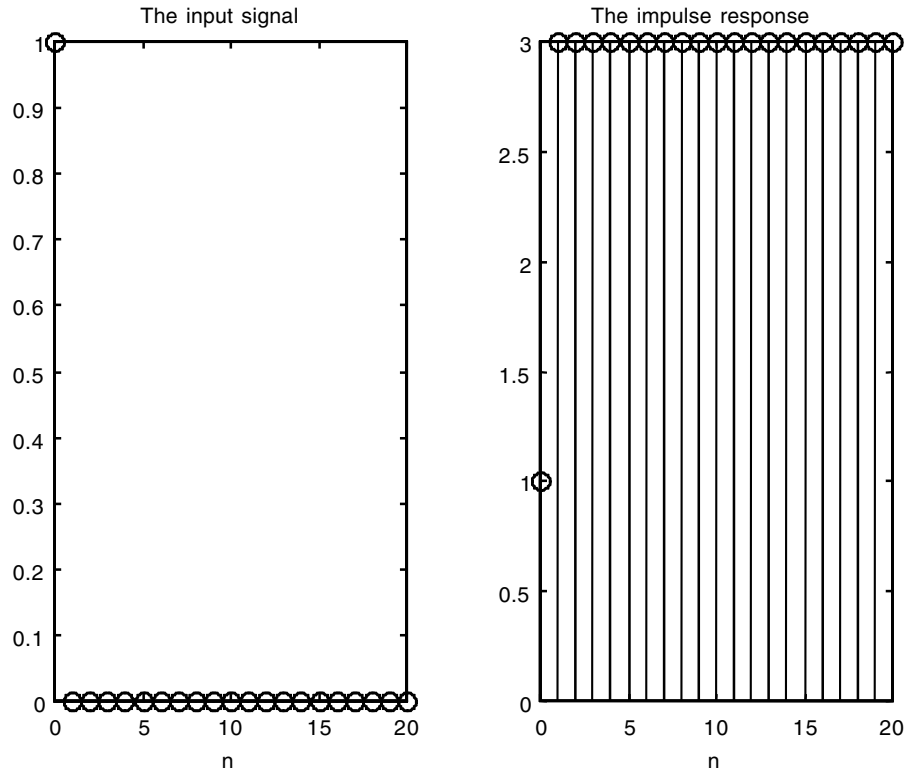

FIGURE 2.40 Signals for EOCE 2.12.

We can also find this response analytically. Recall the general form of the difference equation

$$y(n) + \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k)$$

If N, the degree of the difference equation, is less than M, the general form of $h(n)$ is

$$h(n) = \sum_{k=1}^{N} c_k \left(p_k\right)^n + \sum_{k=0}^{M} A_k \delta(n-k)$$

where p_k is the k^{th} root of the characteristic equation of the system. In this example, $N = 1$ and $M = 1$. Therefore,

$$h(n) = \sum_{k=1}^{1} c_k \left(P_k\right)^n + \sum_{k=0}^{0} A_k \delta(n-k) = c_1 \left(p_1\right)^n + A_0 \delta(n)$$

We need to find the constant c_1 and A_0. We do that by evaluating the equation for $h(n)$ at $n = 0$ and $n = 1$ to get

$$h(0) = c_1 + A_0$$

$$h(1) = p_1 c_1$$

But p_1 is the characteristic root of the characteristic equation and is 1. Thus,

$$h(0) = c_1 + A_0$$

$$h(1) = c_1$$

To solve for c_1 and A_0 we evaluate the difference equation at $n = 0$ and $n = 1$ when $x(n) = \delta(n)$. We will get

$$h(0) - h(-1) = \delta(0) + 2\delta(-1)$$

which gives $h(0) = 1$. We also have

$$h(1) - h(0) = \delta(1) + 2\delta(0)$$

which gives $h(1) = 3$. By comparing this $h(1)$ with the $h(1)$ obtained above, we get $c_1 = 3$ and $A_0 = h(0) - c_1 = 1 - 3 = -2$. Therefore, the final impulse response is

$$h(n) = 3(1)^n - 2\delta(n) = 3 - 2\delta(n)$$

which agrees with the plot in Figure 2.40.

EOCE 2.13

Consider the discrete systems represented by the impulse responses.

1. $h(n) = (.3^n)u(n)$
2. $h(n) = ((.3)^n + (.2)^n)u(n)$
3. $h(n) = \delta(n) + \delta(n-1) + \delta(n-2) + \delta(n-3)$

Find the difference equations first then use MATLAB to find the output $y(n)$ if $x(n) = u(n) - u(n-5)$.

Solution

1. For the first system
   Multiply $h(n) = (.3)^n\, u(n)$ by .3 and shift by 1 to get

   $$(.3)h(n-1) = (.3)(.3)^{n-1}u(n-1)$$

   If we subtract these last two equations we will have

   $$h(n) - .3h(n-1) = (.3)^n u(n) - (.3)(.3)^{n-1}u(n-1)$$

   We can rearrange terms and write

   $$h(n) - .3h(n-1) = (.3)^n u(n) - (.3)^n u(n-1) = (.3)^n\left[u(n) - u(n-1)\right]$$

   But $[u(n) - u(n-1)]$ is the impulse at $n = 0$. Thus

   $$h(n) - .3h(n-1) = (.3)^n\delta(n) = \delta(n)$$

   since $(.3)^n\delta(n)$ has value only when $n = 0$. We know that if the input $x(n)$ is $\delta(n)$ the output is $y(n) = h(n)$. Therefore, the difference equation is

   $$y(n) - .3y(n-1) = x(n)$$

   At this point we have the difference equation and the input, so we can use the MATLAB filter function to find $y(n)$. We will use the MATLAB

functions `stepsignal` and `x1plusx2` to generate $u(n)$ and $-u(n-1)$ and then add them to get $u(n) - u(n-1)$. We do that because these two step signals start at different indices. The MATLAB script follows.

```
n = -10 : 10;
a = [1 -.3];
b = [1];
[x1 n1] = stepsignal(0, -10, 10);
[x2 n2] = stepsignal(5, -5, 10);
[x n] = x1plusx2(x1, -x2, n1, n2);
y = filter(b, a, x);
subplot (1, 2, 1), stem(n, x); ylabel('input signal');
xlabel('n');
subplot (1, 2, 2); stem(n, y); ylabel('output signal');
xlabel ('n');
```

The plots are shown in Figure 2.41.

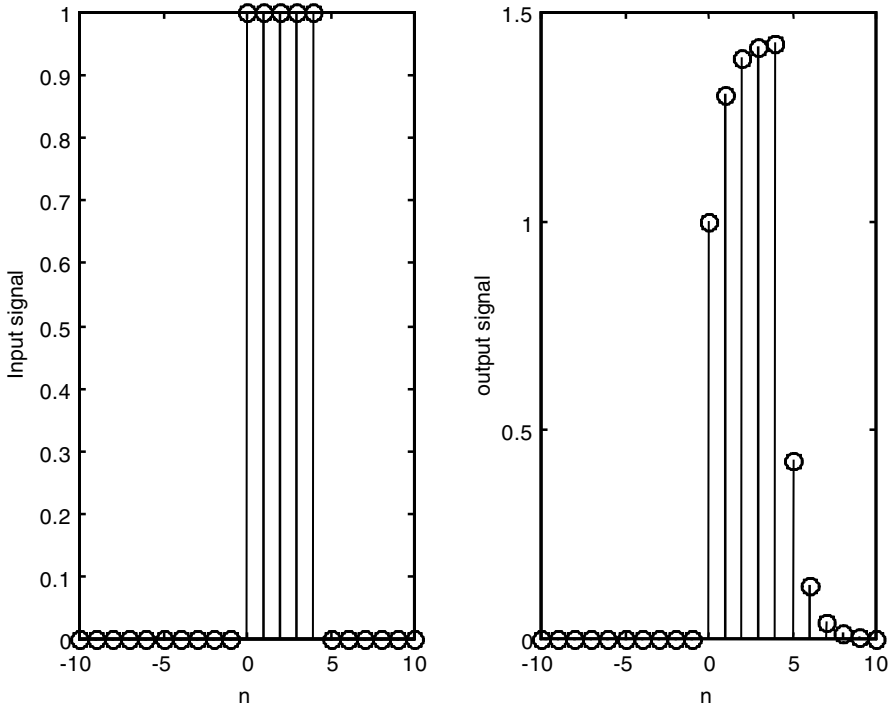

FIGURE 2.41 Signals for EOCE 2.13.

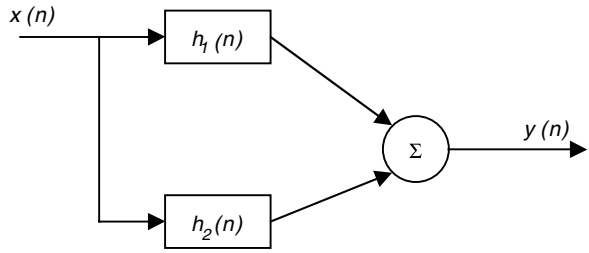

FIGURE 2.42 System for EOCE 2.13.

2. For the second system

$$h(n) = (.3)^n u(n) + (.2)^n u(n)$$

This impulse response can be thought of as a sum of the output responses each due to one system as shown in Figure 2.42. We have

$$h_1(n) = (.3)^n u(n)$$

Multiply by .3 and shift by 1 to get

$$.3h_1(n-1) = (.3)(.3)^{n-1} u(n-1)$$

By subtracting the last two equations we get

$$h_1(n) - .3h_1(n-1) = (.3)^n u(n) - (.3)^n u(n-1) = (.3)^n \left[u(n) - u(n-1)\right]$$

$$= (.3)^n \delta(n) = \delta(n)$$

Therefore, the difference equation is

$$y_1(n) - .3y_1(n-1) = x(n)$$

Similarly, $h_2(n) = (.2)^n u(n)$ can be shown to have the difference equation representation

$$y_2(n) - .2y_2(n-1) = x(n)$$

We will use MATLAB to find $y(n) = y_1(n) + y_2(n)$ for $x(n) = u(n) - n(n-5)$ by writing the following script.

```
n = -10 : 10;
a1 = [1 -.3]; a2 =[ 1 -0.2];
b1 = [1]; b2 = b1;
[x1 n1] = stepsignal(0, -10, 10);
[x2 n2] = stepsignal(5, -5, 10);
```

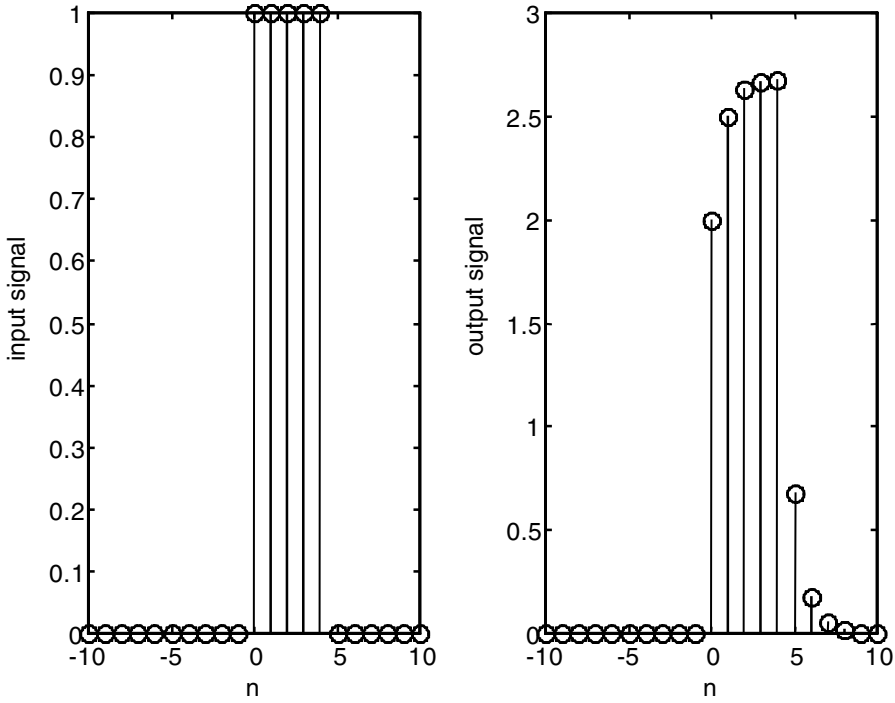

FIGURE 2.43 System for EOCE 2.13.

```
[x n] = x1plusx2(x1, -x2, n1, n2);
y1 = filter(b1, a1, x);
y2 = filter(b2, a2, x);
y = y1 + y2;
subplot (1, 2, 1), stem(n, x); ylabel('input signal');
xlabel('n');
subplot (1, 2, 2); stem(n, y); ylabel('output signal');
xlabel ('n');
```

The plots are shown in Figure 2.43.

3. For the third system

$$h(n) = \delta(n) + \delta(n-1) + \delta(n-2) + \delta(n-3)$$

We know that when $x(n) = \delta(n)$, then $y(n) = h(n)$. The difference equation is then

$$y(n) = x(n) + x(n-1) + x(n-2) + x(n-3)$$

The output is found using MATLAB as in the following script.

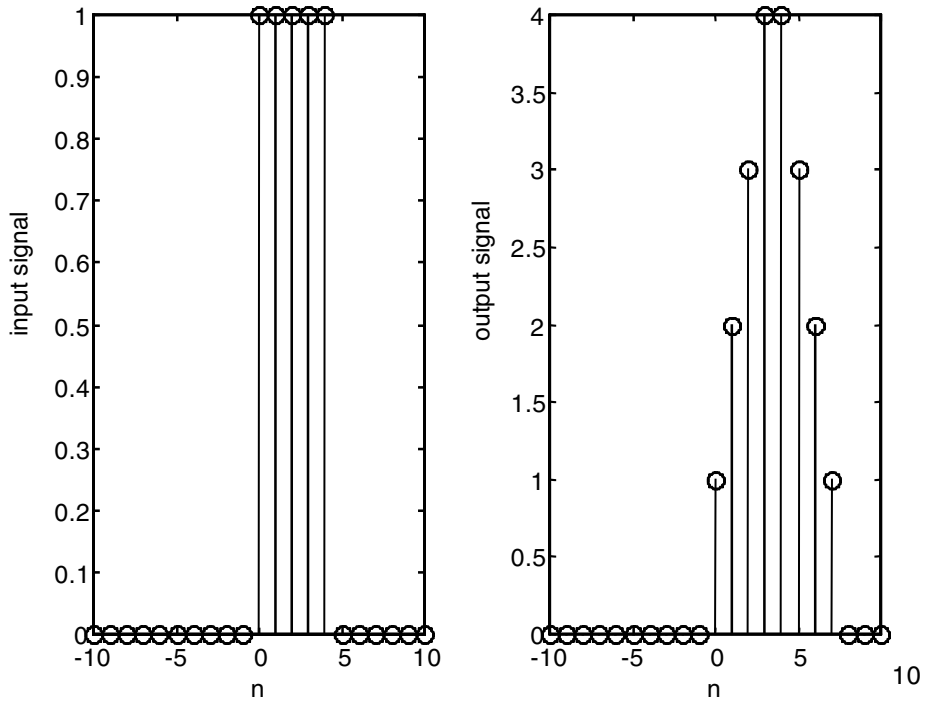

FIGURE 2.44 Signals for EOCE 2.13.

```
n = -10 : 10;
a = [1];
b = [1 1 1 1];
[x1 n1] = stepsignal(0, -10, 10);
[x2 n2] = stepsignal(5, -5, 10);
[x n] = x1plusx2(x1, -x2, n1, n2);
y = filter(b, a, x);
subplot (1, 2, 1), stem(n, x); ylabel('input signal');
xlabel('n');
subplot (1, 2, 2); stem(n, y); ylabel('output signal');
xlabel ('n');
```

The plots are shown in Figure 2.44.

EOCE 2.14

Consider the following systems shown in Figure 2.45 and 2.46. Let $x(n) = u(n)$ in Figure 2.45 and $x(n) = \delta(n)$ in Figure 2.46. Find $y(n)$ for both cases. Let

$$h_1(n) = h_2(n) = h_3(n) = (.5)^n u(n)$$

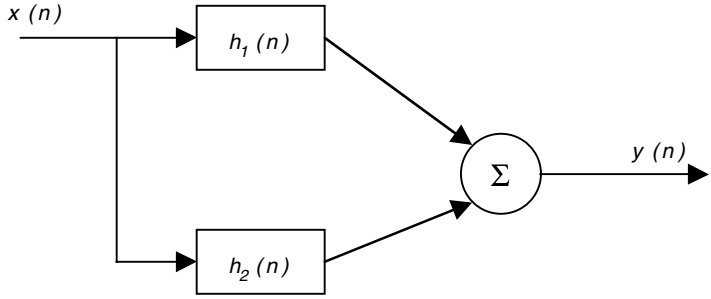

FIGURE 2.45 System for EOCE 2.14.

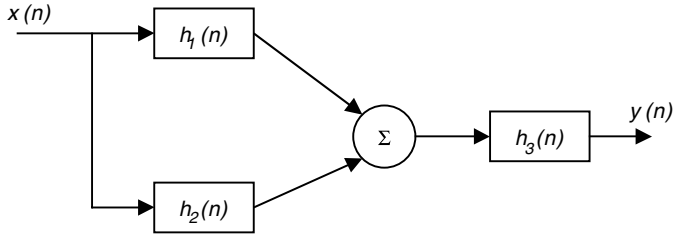

FIGURE 2.46 System for EOCE 2.14.

Solution

For the system in Figure 2.45, the output $y(n)$ is

$$y(n) = x(n) * h_1(n) + x(n) * h_2(n)$$

and

$$x(n) * h_1(n) = \sum_{m=-\infty}^{+\infty} x(m)h_1(n-m) = \sum_{m=-\infty}^{+\infty} u(m)(.5)^{n-m}u(n-m)$$

$$= (.5)^n \sum_{m=0}^{n} \left((.5)^{-1}\right)^m = (.5)^n \frac{\left[1-\left((.5)^{-1}\right)^{n+1}\right]}{1-(.5)^{-1}}$$

$$= -(.5)^n + (.5)^{-1} = 2-(.5)^n \qquad n \geq 0$$

Similarly

$$x(n) * h_2(n) = 2-(.5)^n \qquad n \geq 0$$

Therefore, the final output is

$$y(n) = 4-2(.5)^n \qquad n \geq 0$$

For the system in Figure 2.46, the output of the summer is

$$x(n) * h_1(n) + x(n) * h_2(n) = z(n)$$

The output of the system is

$$y(n) = z(n) * h_3(n)$$

We have

$$x(n) * h_1(n) = 2 - (.5)^n \qquad n \geq 0$$

and

$$x(n) * h_2(n) = 2 - (.5)^n \qquad n \geq 0$$

The output of the summer is

$$z(n) = 4 - 2(.5)^n \qquad n \geq 0$$

and the output of the whole system is then

$$y(n) = \sum_{m=-\infty}^{+\infty} z(m)h_3(n-m) = \sum_{m=-\infty}^{+\infty} \left(4 - 2(.5)^m\right)u(m)(.5)^{n-m}u(n-m)$$

$$y(n) = 4(.5)^n \sum_{m=0}^{n} \left(.5^{-1}\right)^m - 2(.5)^n \sum_{m=0}^{n} 1$$

Finally, the output is

$$y(n) = 4(.5)^n \frac{\left[1 - \left((.5)^{-1}\right)^{n+1}\right]}{1 - (.5)^{-1}} - 2(.5)^n[n+1] = 8 - 4(.5)^n - 2(.5)^n[n+1] \qquad n \geq 0$$

EOCE 2.15

Consider the following signals

$$x_1(n) = u(n) \quad 0 \leq n \leq 9$$

$$x_2(n) = u(n) \quad 9 \leq n \leq 18$$

$$x_3(n) = u(n) \quad -4 \leq n \leq 5$$

$$x_4(n) = u(n) \quad -9 \leq n \leq 0$$

Correlate $x_1(n)$ with itself and all other signals.

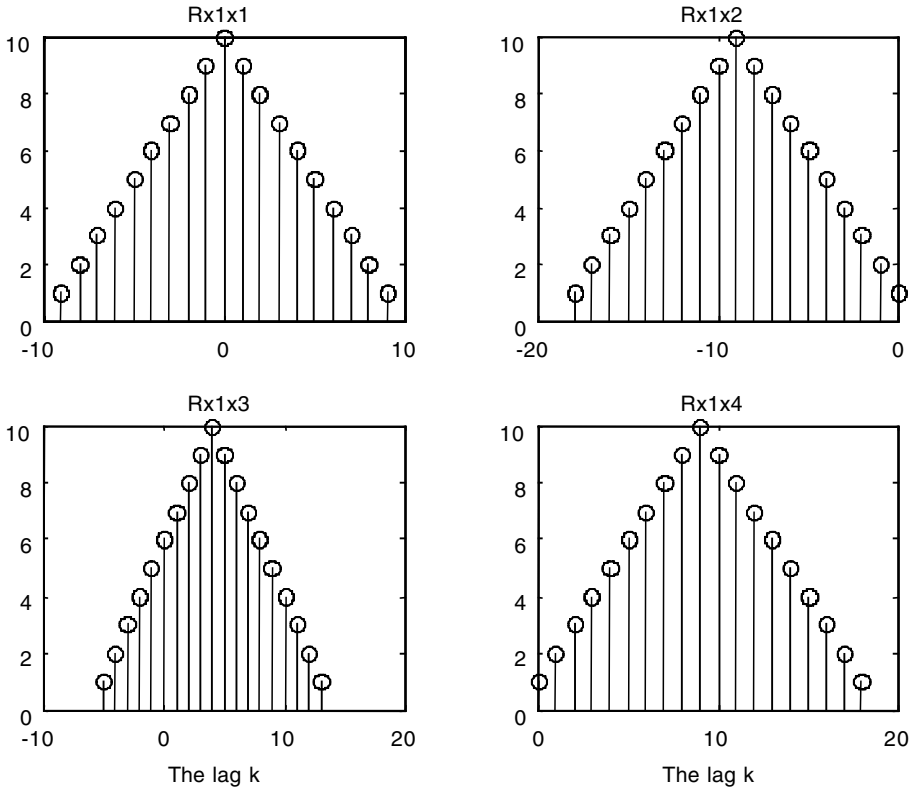

FIGURE 2.47 Plots for EOCE 2.15.

Solution

We will use MATLAB to do this. We can see clearly that for $R_{x_1x_1}$ the strongest correlation is when the lag k is equal to zero. In this case $R_{x_1x_1}$ at $k = 0$ is obtained by summing 1(1) ten times to get 10 as the strength. For $R_{x_1x_2}$ the maximum strength is at $k = -9$. This is true because for a full overlap we need to shift $x_2(n)$ in the reverse direction which means negative k. If $k = 0$ in this case we have only one sample overlap: 1(1) = 1. This is evident from the plot shown in Figure 2.47.

```
%We will compute the correlation using convolution
n1=0:9;n2=9:18;n3=-4:5;n4=-9:0;
x1=ones(size(n1));% x1(n) with 10 ones
x2=x1; x3=x1; x4=x1;%all signals filled with 10 ones
x1ref=fliplr(x1);x2ref=fliplr(x2);% reflecting the signals
x3ref=fliplr(x3); x4ref=fliplr(x4);
n1ref=-fliplr(n1); n2ref=-fliplr(n2);% reflecting indices and
 changing sign
```

```
n3ref=-fliplr(n3); n4ref=-fliplr(n4);
% minimum index for the length of the correlation result
k1min=n1(1)+n1ref(1);
k1max=n1(length(n1))+n1ref(length(n1ref));% maximum length
Rx1x1=conv(x1, x1ref);% the cross-correlation result
k1=k1min:k1max;
k2min=n1(1)+n2ref(1);
k2max=n1(length(n1))+n2ref(length(n2ref));
k2=k2min:k2max;
Rx1x2=conv(x1, x2ref);
k3min=n1(1)+n3ref(1);
k3max=n1(length(n1))+n3ref(length(n3ref));
k3=k3min:k3max;
Rx1x3=conv(x1, x3ref);
k4min=n1(1)+n4ref(1);
k4max=n1(length(n1))+n4ref(length(n4ref));
k4=k4min:k4max;
Rx1x4=conv(x1, x4ref);
subplot(2,2,1);stem(k1,Rx1x1);title('Rx1x1');
subplot(2,2,2);stem(k2,Rx1x2);title('Rx1x2');
subplot(2,2,3);stem(k3,Rx1x3);title('Rx1x3');
xlabel('The lag k');
subplot(2,2,4);stem(k4,Rx1x4);title('Rx1x4');
xlabel('The lag k');
```

EOCE 2.16

One important application of cross-correlation is the estimation of the impulse response $h(n)$. Given a system with impulse response $h(n)$, the output using convolution is $y(n) = x(n)*h(n)$. If we cross-correlate the output of the system with a noisy input with normal distribution, then the cross-correlation

$$R_{yr}(k) = y(n) * r(-n) = \left[r(n) * h(n)\right] * r(-n) = R_{rr}(k) * h(n)$$

where $r(n)$ is a noisy random input; in the above equation we have used the fact that $r(n)*h(n) = h(n)*r(n)$. The signal $R_{rr}(k)$ is the auto-correlation of the input noise. If the input noise has a bandwidth that is much greater than the bandwidth of the system, we write

$$R_{yr}(k) \approx h(n)$$

The reason is that if the auto-correlation of the input noise is approximately an impulse, then we know that it contains a very wide band of frequencies. To see that let us look at the noise auto-correlation. We will use MATLAB to do that.

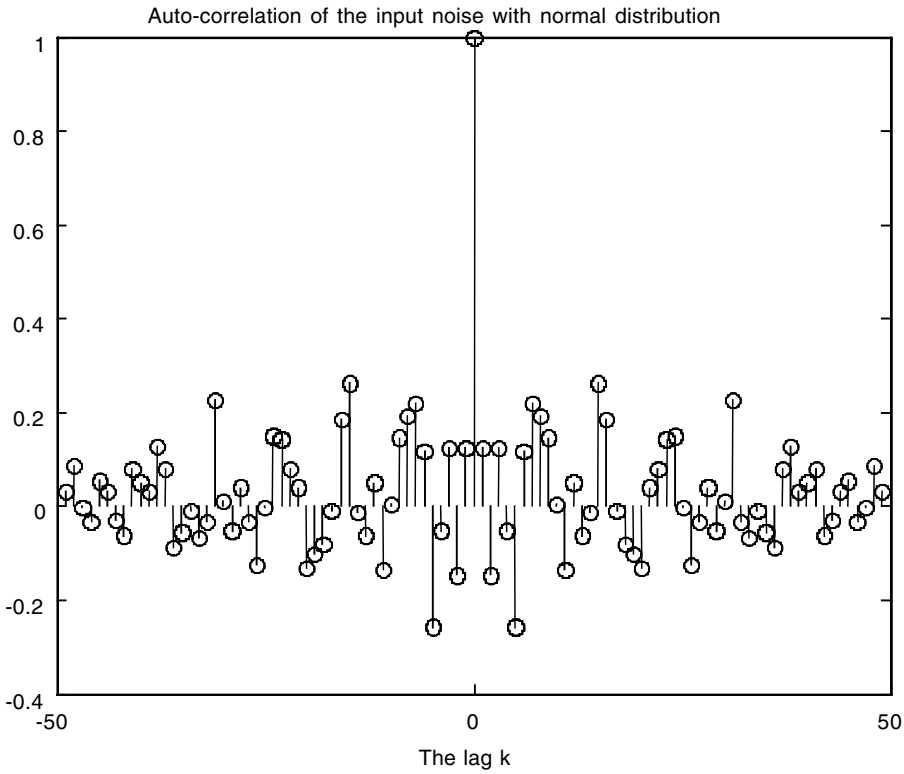

FIGURE 2.48 Plot for EOCE 2.16.

```
N=50;
r=randn(1, N); % generating 1000 random numbers
rref=fliplr(r);
Rrr=conv(r,rref);
k=(-(N-1):(N-1));
title('Auto-correlation of the input noise with normal
 distribution');
stem(n,Rrr/Rrr(N)); xlabel('The lag k');
```

The plot is seen in Figure 2.48. You can easily see that the auto-correlation is approximately the same as the impulse signal, which has a huge bandwidth. Therefore, we say that if we cross-correlate the random normally distributed input noise with the output of the system, we will have an approximation of the impulse response of the system.

Let us look at an example. Consider the system

$$y(n) + .5y(n-1) = x(n)$$

Solution

We will plot its actual impulse response and then the approximation to it using cross-correlation between the output and the input noise. We do that using MATLAB.

```
N=500;
nr=0:499;
ny=nr;
r=randn(1,N);
y=zeros(size(r)); % output initialized to zeros
for n = 2: 500
  y(n)=r(n)-0.5*y(n-1);
end
rr=fliplr(r);
nrr=-fliplr(nr);
Ryr=conv(y, rr);
%k=-(N-1):(N-1);
kmin=ny(1)+nrr(1);
kmax=ny(length(ny))+nrr(length(nrr));
k=kmin:kmax;
subplot(2,1,1);
stem(k,Ryr/Ryr(N));axis([-1 15 -1 1.2]);
title('Approximation of impulse response using cross-
 correlation');
num = [1 0]; den=[ 1 0.5];
n=0: 500;
x=zeros(size(n)); x(1)=1;
[y,v]=dlsim(num,den,x);
yy=conv(y,x);
n=0:1000;
subplot(2,1,2);stem(n,yy); title('Actual impulse response');
axis([-1 15 -1 1.2]);
```

The plots are shown in Figure 2.49.

2.21 End of Chapter Problems

EOCP 2.1

Consider the following systems

1. $y(n) = nx(n)$
   $n \geq 0$

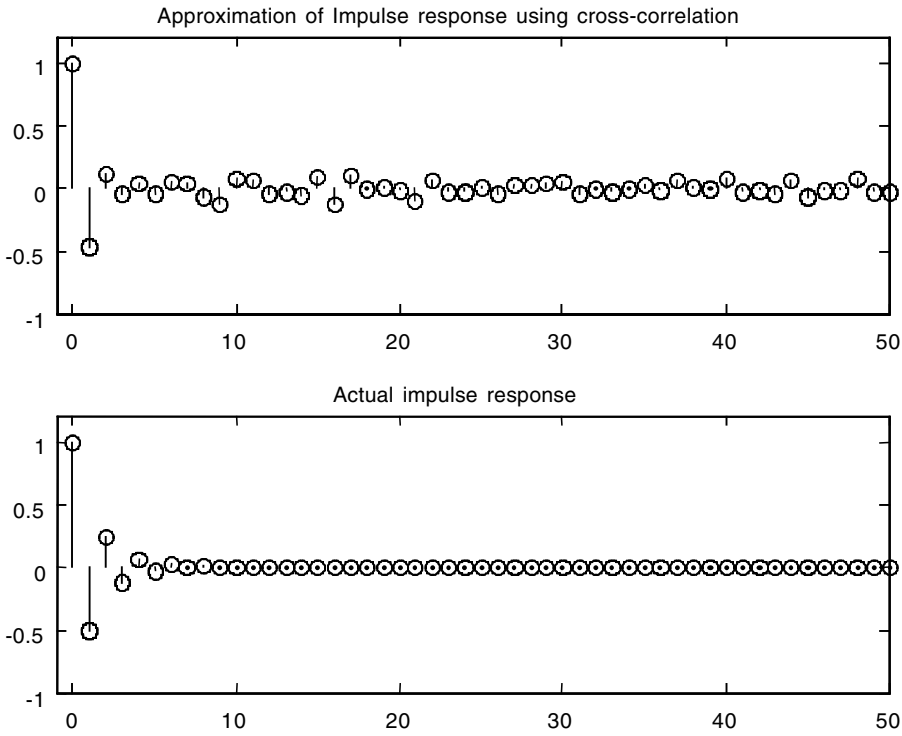

FIGURE 2.49 Plots for EOCE 2.16.

2. $y(n) = \cos(x(n)) + u(n)$
   $n \geq 0$
3. $y(n) = \sqrt{n}x(n)$
4. $y(n) = (4)^n \sin(n)$
5. $y(n) = \cos\left(n\frac{\pi}{2} - 1\right)$
6. $y(n) = \frac{\sin(nx(n))}{n}$
7. $y(n) = \sin(n)\cos(x(n))$
8. $y(n) = y(n-1) + x(n)$
9. $y(n) = ny(n-1) + x(n)$
10. $y(n) = y(n-1) + y(n-2) + x(n)$

Are the above systems linear? Are they time-invariant systems? Show work.

EOCP 2.2

Consider the following systems

1. $h(n) = u(n),\ x(n) = u(n)$
2. $h(n) = (.2)^n u(n),\ x(n) = \delta(n)$
3. $h(n) = (.3)^n u(n),\ x(n) = u(n)$
4. $h(n) = (.3)^n u(n),\ x(n) = (.2)^n u(n)$
5. $h(n) = (.3)^n nu(n),\ x(n) = \delta(n)$
6. $h(n) = n(.5)^n \cos\left(\frac{\pi n}{2} + 1\right) u(n),\ x(n) = \delta(n)$
7. $h(n) = (.4)^n u(n),\ x(n) = u(n) - u(n-2)$
8. $h(n) = u(n) - u(n-2),\ x(n) = u(n) - u(n-2)$
9. $h(n) = (.4)^n u(n),\ x(n) = (.5)^n [u(n) - u(n-1)]$
10. $h(n) = (.1)^n u(n),\ x(n) = (.5)^n [u(n) - u(n-5)]$

Find the output $y(n)$ for each system using the convolution sum equation. Are the systems stable?

EOCP 2.3

Consider the following finite signals:

1. $x_1(n) = \{\underset{\uparrow}{1}\ 1\ 1\ 1\}\quad x_2(n) = \{\underset{\uparrow}{1}\ 1\ 1\ 1\}$
2. $x_1(n) = \{1\ 1\ \underset{\uparrow}{1}\ 1\}\quad x_2(n) = \{1\ 1\ \underset{\uparrow}{1}\ 1\}$
3. $x_1(n) = \{-1\ 2\ 1\ \underset{\uparrow}{3}\}\quad x_2(n) = \{\underset{\uparrow}{1}\ 2\}$
4. $x_1(n) = \{-1\ -2\ \underset{\uparrow}{-3}\ -4\}\quad x_2(n) = \{\underset{\uparrow}{1}\ 2\}$
5. $x_1(n) = \{1\ -1\ 2\ \underset{\uparrow}{-2}\ 3\ -3\}\quad x_2(n) = \{1\ \underset{\uparrow}{2}\ 3\}$

Find $x(n) = x_1(n) * x_2(n)$ for each case above.

EOCP 2.4

Consider the following systems with the initial conditions.

1. $y(n) - .6\, y(n-1) = 0,\quad y(-1) = 1$
2. $y(n) - .6\, y(n-2) = 0,\ y(-1) = 0, y(-2) = 1$
3. $y(n) - .6\, y(n-1) + .6\, y(n-2) = 0,\quad y(-1) = 0, y(-2) = -1$
4. $y(n) - .1\, y(n-3) = 0,\quad y(-1) = y(-2) = 0, y(-3) = 1$
5. $y(n) + .1\, y(n-1) + y(n-3) = 0, y(-1) = 0,\ y(-2) = 1, y(-3) = 0$
6. $y(n) + .6y(n-2) = 0,\quad y(-1) = y(-2) = 1$

Are the above systems stable? What is the output due to the given initial conditions?

EOCP 2.5

Consider the systems

1. $y(n) - .6y(n-1) = \sin(n), \quad y(-1) = 0$
2. $y(n) - .6y(n-2) = \delta(n), \quad y(-1) = 0, y(-2) = 1$
3. $y(n) - .6y(n-1) + .6y(n-2) = 3\delta(n), \quad y(-1) = 0, y(-2) = 0$
4. $y(n) - .1y(n-3) = \delta(n), \quad y(-1) = y(-2) = y(-3) = 0$
5. $y(n) + .1y(n-1) + y(n-3) = 2\delta(n), \quad y(-1) = y(-2) = y(-3) = 0$
6. $y(n) + .6y(n-2) = 3u(n), \quad y(-1) = y(-2) = 0$

Find the total output, $y(n)$, for each system above.

EOCP 2.6

Consider the following difference equations

1. $y(n) - .6y(n-1) = x(n)$
2. $y(n) - .6y(n-2) = x(n) - x(n-1)$
3. $y(n) - .6y(n-1) + .6y(n-2) = x(n)$
4. $y(n) - .1y(n-3) = x(n) + x(n-1)$
5. $y(n) + .1y(n-1) + y(n-3) = x(n)$
6. $y(n) + .6y(n-2) = x(n-2)$

Draw the block diagram for each system.

EOCP 2.7

Consider the following systems

1. $h(n) = \delta(n) - \delta(n-1)$
2. $h(n) = \delta(n) - \delta(n-1) + \delta(n-2) + \delta(n-4)$
3. $h(n) = u(n)$
4. $h(n) = (.3)^n u(n)$
5. $h(n) = (.3)^n u(n) + (.6)^n u(n)$
6. $h(n) = u(n), \quad 0 \le n \le 5$

What are the difference equations that represent these systems?

EOCP 2.8

Consider the following difference equations

1. $y(n) + 3y(n-1) = x(n)$
2. $y(n) + 3y(n-1) + 3y(n-2) = x(n)$

3. $y(n) - .1y(n-2) = x(n)$
4. $y(n) + .1y(n-2) = x(n)$
5. $y(n) + 3y(n-1) = x(n) + x(n-1)$
6. $y(n) + 3y(n-1) + 3y(n-2) = x(n) + x(n-1) + x(n-2)$
7. $y(n) - .1y(n-2) = x(n) + x(n-1) + x(n-2)$
8. $y(n) + .1y(n-2) = x(n) + x(n-1) + x(n-2) + x(n-4)$
9. $y(n) + 3y(n-1) = x(n) + x(n-1) + x(n-2)$
10. $y(n) + 3y(n-1) = x(n) + x(n-1) + x(n-2) + x(n-3)$

What are the impulse responses for each system above?

EOCP 2.9

Consider the block diagrams in Figures 2.50 through 2.59. What are the impulse responses for each system?

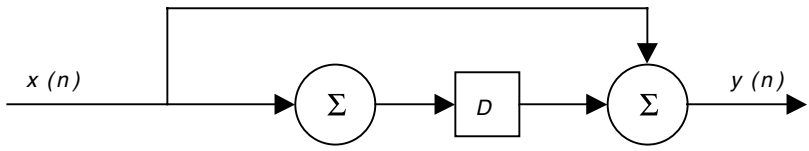

FIGURE 2.50 System for EOCP 2.9.

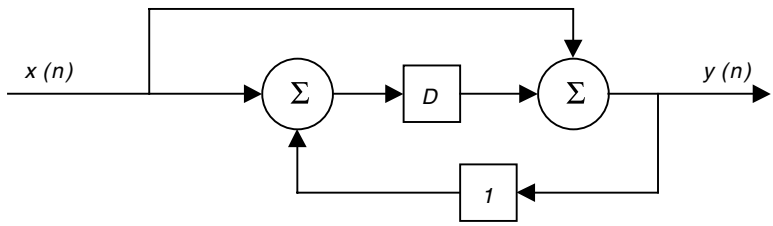

FIGURE 2.51 System for EOCP 2.9.

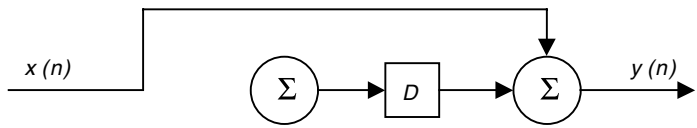

FIGURE 2.52 System for EOCP 2.9.

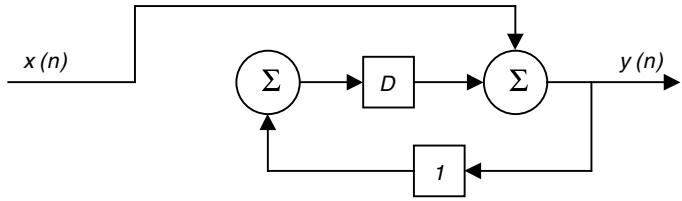

FIGURE 2.53 System for EOCP 2.9.

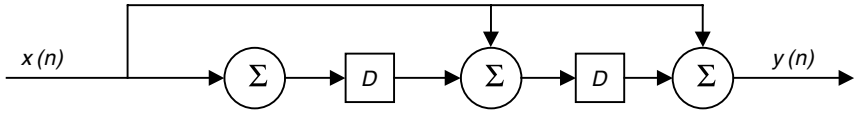

FIGURE 2.54 System for EOCP 2.9.

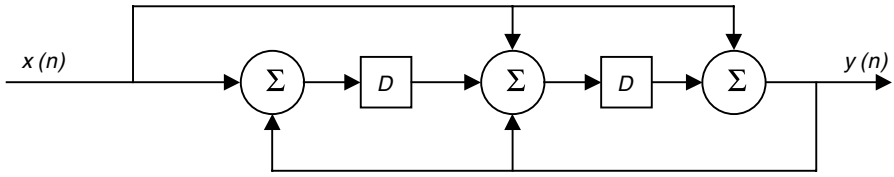

FIGURE 2.55 System for EOCP 2.9.

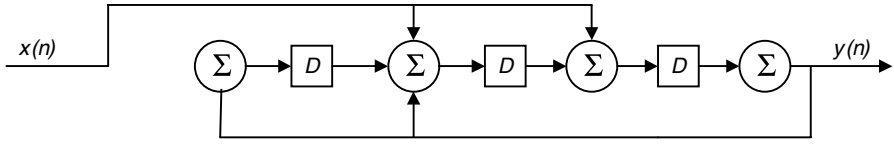

FIGURE 2.56 System for EOCP 2.9.

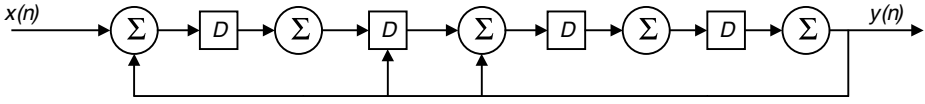

FIGURE 2.57 System for EOCP 2.9.

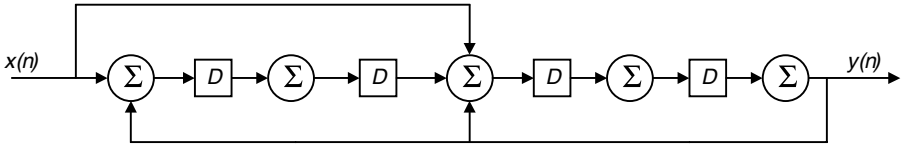

FIGURE 2.58 System for EOCP 2.9.

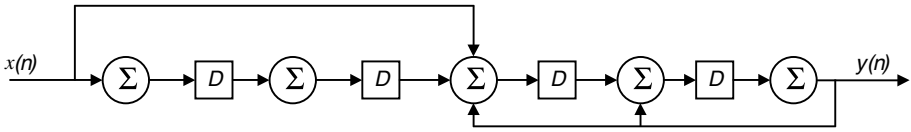

FIGURE 2.59 System for EOCP 2.9

EOCP 2.10

Consider the following systems

1. $y(n) + ky(n-1) = x(n)$
2. $y(n) + ky(n-1) + y(n-2) = x(n)$
3. $y(n) - ky(n-1) = x(n)$
4. $y(n) + y(n-1) + ky(n-2) = x(n)$
5. $y(n) + .1y(n-1) + .3y(n-2) + y(n-3) = x(n)$

a) For what values of k are the systems stable?
b) Use MATLAB to find the step, impulse and the sinusoidal responses for all systems with suitable k.

EOCP 2.11

Consider the difference equation

$$y(n) - yk(n-1) + 0.5y(n-2) = 0.5x(n)$$

1. Is the system stable? For what k values?
2. Draw the block diagram representation of the system.
3. Use MATLAB to find the step and the impulse responses for a certain k.
4. Find the impulse response analytically with k of your choice.
5. Find the step response analytically assuming zero initial conditions with suitable k.
6. Solve the difference equation by iterations for n = 0, 1, 2, 3 and 5 if $x(n) = u(n)$ for suitable k.

EOCP 2.12

Consider the system represented by the difference equation

$$y(n) - 6y(n-1) + 9ky(n-2) = 19x(n)$$

1. Use MATLAB to find the step and the impulse responses for a k value that makes the system stable.
2. Use MATLAB to find the output $y(n)$ if $x(n) = 2u(n) + 5nu(n)$.
3. Draw the block diagram for the system.

EOCP 2.13

Consider the following systems

1. $y(n) + .2y(n-1) = x(n)$
2. $y(n) + .2y(n-1) + .1y(n-2) = x(n)$

Approximate the impulse response using cross-correlation. Compare with the actual impulse response.

EOCP 2.14

Consider the following outputs of the linear discrete systems

1. $y(n) = (.3)nu(n)$
2. $y(n) = (.3)^n u(n) + (.6)^n u(n)$

Approximate the impulse responses using correlations.

3

The Fourier Series and the Fourier Transform of Discrete Signals

3.1 Introduction

According to Joseph Fourier, a discrete periodic signal can be represented as a sum of complex exponentials or sinusoids. Also, a nonperiodic discrete signal and a finite-duration discrete signal can be represented as a finite sum of complex exponentials. This last representation for finite-duration discrete signals is called the Fourier transform of discrete signals. There are many advantages for this representation. The most important one is that if the discrete signal is put in the frequency domain, every single frequency in the signal will be clearly identified. Consider the plot of the discrete signal

$$y(n) = \cos(2\pi n) + 200\cos(4\pi n) + 0.1\cos(10\pi n)$$

as shown in Figure 3.1. By looking at the plot you would hardly think that this signal is composed of many sinusoids. You will probably reason that this signal contains no sinusoids at all. In the frequency domain, if we look at the magnitude of the signal, we see something similar to Figure 3.2.

You can clearly see the three frequencies in the figure. You may also argue that these signals of small magnitudes, like the term $0.1\cos(10\pi n)$, can be neglected. You may be wrong, especially if this term is the term that we are interested in. Before we present discussion about the discrete Fourier series and the discrete Fourier transform, we provide a very useful review on complex numbers.

3.2 Review of Complex Numbers

This is a good place to give a brief review of complex numbers because the chapters ahead, as well as this chapter, are heavily involved with their arithmetic manipulation.

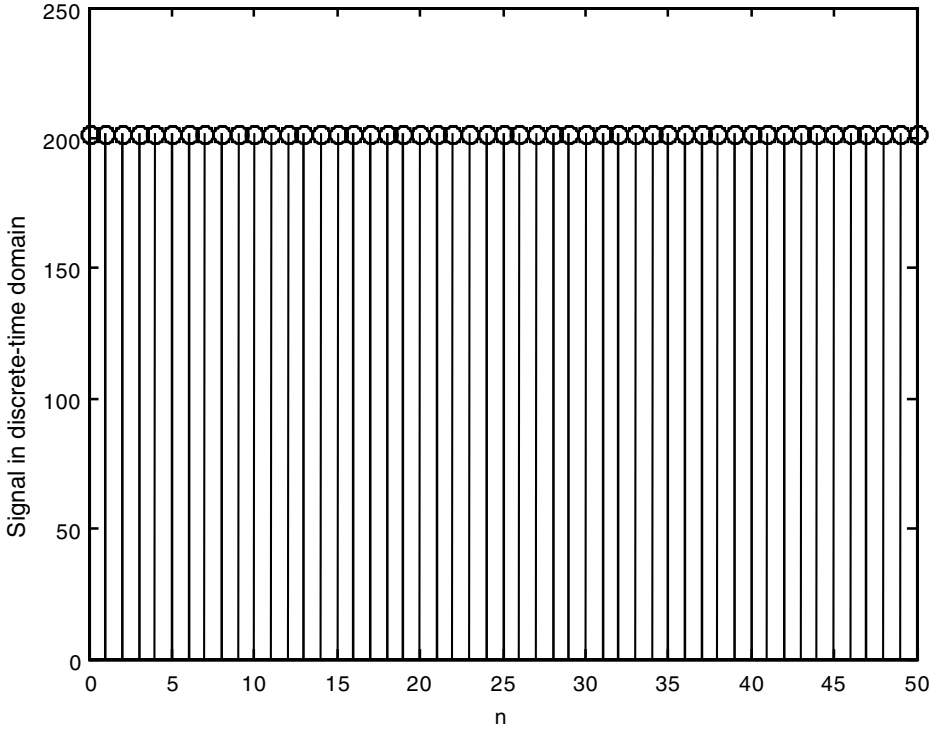

FIGURE 3.1 A discrete sinusoidal signal.

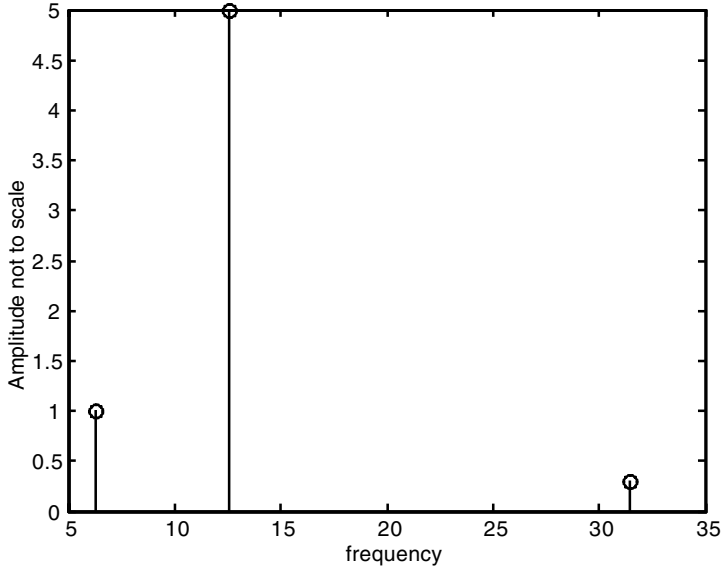

FIGURE 3.2 A discrete signal in the frequency domain.

3.2.1 Definition

By definition, the complex number j is the square root of –1. In general, the complex number $C = P + jQ$ consists of two parts; the real part, P, and the imaginary part, Q. C can be represented in many forms as we will see later. If we are multiplying or dividing complex numbers we would prefer to use the polar form. If we are adding or subtracting complex numbers we would rather use the rectangular form. The reason for that is the ease each form provides in the corresponding calculation.

Consider two complex numbers C_1 and C_2 where $C_1 = P_1 + jQ_1$ and $C_2 = P_2 + jQ_2$. These complex numbers are given in the rectangular form.

3.2.2 Addition

When we add two complex numbers we add their real parts and their imaginary parts together to form the addition

$$C_1 + C_2 = (P_1 + P_2) + j(Q_1 + Q_2) \tag{3.1}$$

3.2.3 Subtraction

When we subtract two complex numbers we subtract their real parts and their imaginary parts to form the subtraction

$$C_1 - C_2 = (P_1 - P_2) + j(Q_1 - Q_2) \tag{3.2}$$

3.2.4 Multiplication

Let us now consider the two complex numbers in polar form. In polar form C_1 and C_2 are represented as

$$C_1 = M_1 e^{j\theta_1} \text{ and } C_2 = M_2 e^{j\theta_2}$$

where

$$M_1 = \sqrt{P_1^2 + Q_1^2}$$

$$M_2 = \sqrt{P_2^2 + Q_2^2}$$

and

$$\theta_1 = \tan^{-1}\left[\frac{Q_1}{P_1}\right]$$

$$\theta_2 = \tan^{-1}\left[\frac{Q_2}{P_2}\right]$$

We can also represent the complex numbers C_1 and C_2 as $M_1 \angle \theta_1$ and $M_2 \angle \theta_2$, where M_1, M_2, θ_1 and θ_2 are as given above. The complex number $M_1 \angle \theta_1$ is read as a complex number with magnitude M_1 and phase angle θ_1. Complex numbers are easily multiplied in polar form as

$$M_1 e^{j\theta_1} M_2 e^{j\theta_2} = M_1 M_2 e^{j(\theta_1+\theta_2)} = M_1 M_2 \angle(\theta_1 + \theta_2) \tag{3.3}$$

If we have more than two complex numbers in polar form to be multiplied we use the same procedure. We multiply their magnitudes and add their phase angles to form the new product.

3.2.5 Division

To divide two complex numbers we divide their magnitudes and subtract their phase angles.

$$\begin{aligned} C_1/C_2 &= \left[M_1 e^{j\theta_1}\right] / \left[M_2 e^{j\theta_2}\right] \\ &= \left[M_1/M_2\right] e^{j(\theta_1-\theta_2)} = \left[M_1/M_2\right] \angle(\theta_1 - \theta_2) \end{aligned} \tag{3.4}$$

3.2.6 From Rectangular to Polar

Consider the complex number $C_1 = P_1 + jQ_1$. This representation in the rectangular form is shown in Figure 3.4. To convert to polar form we write C_1 as $C_1 = M_1 e^{j\theta_1} = M_1 \angle \theta_1$ where

$$M_1 = \sqrt{P_1^2 + Q_1^2} \quad \text{and} \quad \theta_1 = \tan^{-1}\left[\frac{Q_1}{P_1}\right]$$

3.2.7 From Polar to Rectangular

Consider the complex number $C_1 = M_1 e^{j\theta_1} = M_1 \angle \theta_1$ in polar form as seen in Figure 3.3. To convert to its equivalent rectangular form we write C_1 as $C_1 = P_1 + jQ_1$, where

$$M_1^2 = P_1^2 + Q_1^2 \quad \text{and} \quad \tan(\theta_1) = \left[\frac{Q_1}{P_1}\right]$$

These two equations can be solved simultaneously for P_1 and Q_1.

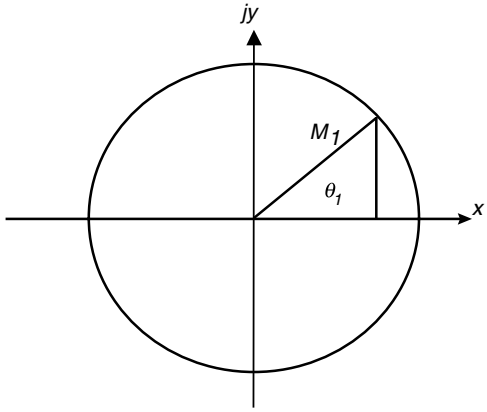

FIGURE 3.3 Rectangular form.

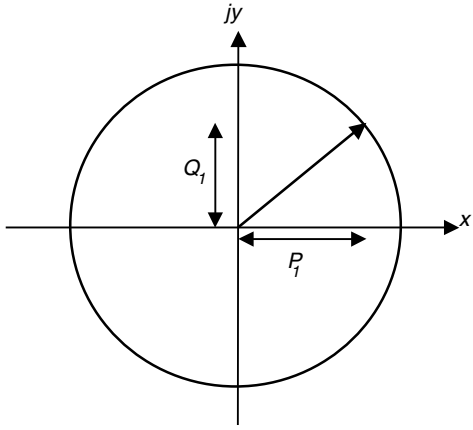

FIGURE 3.4 Polar form.

3.3 The Fourier Series of Discrete Periodic Signals

We are given the discrete signal $x(n)$ that might be the result of sampling a continuous signal $x(t)$ at $t = nT_s$ sec. This signal is periodic of period N samples if

$$x(n) = x(n+N)$$

for all integer values n. The fundamental period is the smallest N that satisfies the above condition. The period in seconds would be NT_s where T_s is the

sampling period. In this case the digital frequency is $\theta = 2\pi/NT_s$. Now consider the complex exponential set

$$\phi_m(n) = e^{jm\theta nT_s} = e^{j2\pi mn/N} \qquad m = 0, \pm 1, \pm 2, \ldots \tag{3.5}$$

where m is referred to as the frequency index and n is the discrete-time index. Notice that this complex exponential set is periodic with period N for all integers n since

$$\phi_m(n+N) = e^{jm\theta(n+N)T_s} = e^{j2\pi m} e^{j2\pi mn/N} = 1\, e^{j2\pi mn/N} = \phi_m(n) \tag{3.6}$$

Notice also that since $\phi_m\ (n + N) = \phi_m(n)$, there is only one set of complex functions of length N that is unique in the overall set. Unique set means that the members in the set are independent; no one function or complex exponential in the set can be written as a linear combination of the others. In the set $\phi_m(n)$ of length N we see that

$$\sum_{n=0}^{N-1} \phi_m(n) = \sum_{n=0}^{N-1} e^{j2\pi mn/N} = \begin{cases} N & m = 0, \pm N, \pm 2N, \ldots \\ 0 & m \neq 0, \pm N, \pm 2N, \ldots \end{cases} \tag{3.7}$$

since using the geometric series sum we have

$$\sum_{n=0}^{N-1} e^{j2\pi mn/N} = \sum_{n=0}^{N-1} \left(e^{j2\pi m/N}\right)^n = \frac{1 - e^{(j2\pi m/N)N}}{1 - e^{j2\pi m/N}} = \begin{cases} N & m = 0, \pm N, \pm 2N, \ldots \\ 0 & m \neq 0, \pm N, \pm 2N, \ldots \end{cases} \tag{3.8}$$

We can also see that

$$\sum_{n=0}^{N-1} e^{j2\pi mn/N} e^{-j2\pi kn/N} = \sum_{n=0}^{N-1} \left(e^{j2\pi n(m-k)/N}\right) = \begin{cases} N & m = k, \pm k + N, \pm k + 2N, \ldots \\ 0 & m \neq k, \pm k + N, \pm k + 2N, \ldots \end{cases} \tag{3.9}$$

This is the orthogonality condition. The periodic signal $x(n)$ can be approximated as the linear combination of the unique set $\phi(m)$ with the N complex exponentials as

$$x(n) = \sum_{n=0}^{N-1} c_m e^{j2\pi mn/N} \tag{3.10}$$

To find the Fourier series coefficients we multiply the above approximation by $e^{-j2\pi kn/N}$ and then sum over the period N to get

$$c_m = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi mn/N} \tag{3.11}$$

using the geometric series sum and the orthogonality condition we established earlier. Try to see it for yourself.

Writing the approximation to the periodic signal $x(n)$ allows us to clearly see the frequency contents of the signal through the coefficients c_m. These coefficients are complex numbers, yet they have a magnitude and a phase. It can be easily seen that the c_m set is periodic. We write

$$c_{m+N} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi(m+N)n/N} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi n}e^{-j2\pi nm/N} = c_m \tag{3.12}$$

Also, if $x(n)$ is real, then the Fourier series magnitude coefficients have even symmetry and the phase of the coefficients have odd symmetry. We finish this topic with an example. Consider the periodic discrete signal $x(n)$ = {1 0 1 1 0 1 1 0 1 …} with T_s = 0.1 sec as the sampling interval used to obtain $x(n)$. The period N is clearly 3. Thus the Fourier series coefficients are calculated as in the following:

$$c_0 = \frac{1}{3}\sum_{n=0}^{2} x(n)e^{-j2\pi 0n/3} = \frac{1}{3}\left[x(0)+x(1)+x(2)\right] = \frac{2}{3}$$

$$c_1 = \frac{1}{3}\sum_{n=0}^{2} x(n)e^{-j2\pi n/3} = \frac{1}{3}\left[x(0)+x(1)e^{-j2\pi/3}+x(2)e^{-j4\pi/3}\right] = \frac{1}{3}\left[0.5+j.886\right]$$

$$c_2 = \frac{1}{3}\sum_{n=0}^{2} x(n)e^{-j4\pi n/3} = \frac{1}{3}\left[x(0)+x(1)e^{-j4\pi/3}+x(2)e^{-j8\pi/3}\right] = \frac{1}{3}\left[0.5-j.886\right]$$

These are the frequency components in the signal $x(n)$. These components are located at $m\theta = m2\pi/NT_s = 20m\pi/3$ rad/sec for m = 0, 1, and 2. Thus these frequencies are located at 0, $20\pi/3$ and $40\pi/3$ rads per sample. This is something very difficult to see given only the discrete periodic signal $x(n)$ = {1 0 1 1 0 1 1 0 1 …} with T_s = 0.1 sec. We will come back to the Fourier series coefficients in Chapter 7 and see how can we utilize this development to estimate the average power, the average value and the energy in signals. We will also learn how to compute these Fourier series coefficients using the computer and the very well known algorithm, the fast Fourier transform.

3.4 The Discrete System with Periodic Inputs: The Steady-State Response

To start the process, let us consider the input $x(n)$ to be of the form

$$x(n) = X\cos(\theta n)$$

The term $\cos(\theta n)$ can also be written as

$$\cos(\theta n) = \frac{1}{2}\left[e^{j\theta n} + e^{-j\theta n}\right]$$

Now we can use superposition and apply the input $e^{j\theta n}$ and $e^{-j\theta n}$ one at a time. Let us take first $x(n) = e^{j\theta n}$ as the input to the system represented by $h(n)$, the impulse response. The output of the system in this case is

$$y(n) = x(n) * h(n) = h(n) * x(n) = \sum_{m=-\infty}^{+\infty} h(m)x(n-m) = \sum_{m=-\infty}^{+\infty} h(m)e^{j\theta(n-m)}$$

$$y(n) = \sum_{m=-\infty}^{+\infty} h(m)e^{j\theta n}e^{-j\theta m} = e^{j\theta n}\sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}$$

Given that n was taken in the range $\infty \le n \le \infty$, the output here is the steady-state output that remains after all the transients die. Finally, with $x(n) = e^{j\theta n}$ we have

$$y_{ss}(n) = x(n)\sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}$$

where $y_{ss}(n)$ is the steady-state response and $x(n) = e^{j\theta n}$. Let us define $H(e^{j\theta})$ as

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m} \tag{3.13}$$

then

$$y_{ss}(n) = x(n)H\left(e^{j\theta}\right) \tag{3.14}$$

Example 3.1

Consider the impulse response for a discrete system as $h(n) = \delta(n) + \delta(n-1)$ and let the input to the system be $x(n) = e^{j2\pi n}$ for $-\infty \le n \le +\infty$. Find the steady-state output.

Solution

From the relation $y_{ss}(n) = x(n)H(e^{j\theta})$ with $x(n) = e^{j2\pi n}$ and $H(e^{j\theta})$ as

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} \left(\delta(m) + \delta(m-1)\right)e^{-j\theta m} = \left(\delta(0) + \delta(-1)\right)e^{-j\theta(0)} + \left(\delta(1) + \delta(0)\right)e^{-j\theta(1)}$$

$$H\left(e^{j\theta}\right) = 1 + e^{-j\theta}$$

Thus the steady-state output becomes

$$y_{ss}(n) = e^{j2\pi n}\left(1 + e^{-j\theta}\right) \qquad 0 \le n \le \infty$$

The input is applied at $\theta = 2\pi$. So

$$H\left(e^{j\theta}\right) = 1 + e^{-j2\pi} = 1 + \cos 2\pi - j\sin 2\pi = 2$$

and

$$y_{ss}(n) = 2e^{j2\pi n} \qquad -\infty \le n \le +\infty$$

Example 3.2

For the same system as in Example 3.1, and with the input $x(n) = 10 + \cos(n\pi)$, what is the steady-state output?

Solution

To use $y_{ss}(n) = e^{j\theta n}H(e^{j\theta})$, we must put all the terms in $x(n)$ in exponential form and then we can use the superposition principle to find

$$y_{ss1}(n) \text{ due to } x_1(n) = 10 = 10e^{jn(0)}$$

$$y_{ss2}(n) \text{ due to } x_2(n) = \frac{1}{2}e^{j\pi n}$$

$$y_{ss3}(n) \text{ due to } x_3(n) = \frac{1}{2}e^{-j\pi n}$$

where

$$\cos(n\pi) = \frac{e^{jn\pi} + e^{-jn\pi}}{2}$$

From Example 3.1 we have $H(e^{j\theta}) = 1 + e^{-j\theta}$ and for $x_1(n) = 10e^{j\theta(0)}$, we have

$$H\left(e^{j\theta}\right) = 2$$

$$y_{ss1}(n) = 10e^{j\theta n}(2) = 20$$

For

$$x_2(n) = \frac{e^{j\pi n}}{2} \text{ with } \theta = \pi$$

we have

$$H\left(e^{j\theta}\right) = 1 + e^{-j\pi} = 1 + \cos\pi - j\sin\pi = 0$$

and

$$y_{ss2}(n) = \frac{e^{j\pi n}}{2}(0) = 0$$

For

$$x_3(n) = \frac{e^{-j\pi n}}{2} \text{ with } \theta = -\pi$$

we have

$$H\left(e^{-j\theta}\right) = 1 + e^{-j\theta} = 1 + e^{j\pi} = 1 + \cos\pi + j\sin\pi = 0$$

and

$$y_{ss3}(n) = \frac{e^{-j\pi n}}{2}(0) = 0$$

Therefore, the total solution is the combination

$$y_{ss}(n) = y_{ss1}(n) + y_{ss2}(n) + y_{ss3}(n) = 20 + 0 + 0 = 20 \qquad -\infty \le n \le +\infty$$

Example 3.3

Find the steady-state output for the system

$$h(n) = (.5)^n u(n)$$

with the input $x(n) = 10$.

Solution

We will find $H(e^{j\theta})$ first.

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m} = \sum_{m=0}^{\infty} (.5)^m e^{-j\theta m} = \sum_{m=0}^{\infty} \left((.5)e^{-j\theta}\right)^m = \frac{1}{1-(.5)e^{-j\theta}}$$

The input is $x(n) = 10 = 10\, e^{jn0}$. So $\theta = 0$ and

$$H\left(e^{j\theta}\right) = \frac{1}{1-(.5)e^{-j0}} = \frac{1}{1-0.5} = 2$$

and the steady-state response is

$$y_{ss}(n) = 10(2) = 20 \quad -\infty \le n \le +\infty$$

3.4.1 The General Form for $y_{ss}(n)$

Consider the input $x(n)$ as

$$x(n) = X\cos(\theta n + \varphi) = \frac{X}{2}\left[e^{j(\theta n+\varphi)} + e^{-j(\theta n+\varphi)}\right]$$

when θ is the radian frequency and φ is the phase shift. If $x(n)$ is applied to a linear time-variant system, the steady-state output using Equation (3.14) will be

$$y_{ss}(n) = \frac{X}{2}e^{j\varphi}e^{j\theta n}H\left(e^{j\theta}\right) + \frac{X}{2}e^{-j\varphi}e^{-j\theta n}H\left(e^{-j\theta}\right)$$

Let us write $H(e^{j\theta})$ in terms of its magnitude and phase as $Me^{j\alpha}$ where M is the magnitude of $H(e^{j\theta})$ and α is its phase shift. The steady-state solution is then

$$y_{ss}(n) = \frac{X}{2}e^{j\varphi}e^{j\theta n}Me^{j\alpha} + \frac{X}{2}e^{-j\varphi}e^{-j\theta n}Me^{-j\alpha} = \frac{XM}{2}\left[e^{j\varphi}e^{j(\theta n+\alpha)} + e^{-j\varphi}e^{-j(\theta n+\alpha)}\right]$$

By putting exponents together we get

$$y_{ss}(n) = \frac{XM}{2}\left[e^{j(\theta n+\alpha+\varphi)} + e^{-j(\theta n+\alpha+\varphi)}\right]$$

and finally the steady-state response is written as

$$y_{ss}(n) = XM\cos(\theta n + \varphi + \alpha) \tag{3.15}$$

Example 3.4

Let $h(n) = \delta(n) + \delta(n-1)$ and let $x(n) = 10\cos\left(n\frac{\pi}{2} + \frac{1}{2}\right)$. Find the steady-state response $y_{ss}(n)$.

Solution

To find the steady-state response we need to use the formula in Equation (3.15). Thus we need values for the parameters in the formula. The magnitude of the input is $X = 10$. M is the magnitude of $H(e^{j\theta}) = 1 + e^{-j\theta}$ as shown in previous examples. The input frequency is $\theta = \pi/2$, and $H(e^{j\theta})$ is

$$H\left(e^{j\frac{\pi}{2}}\right) = 1 + e^{-j\frac{\pi}{2}} = 1 + \cos\left(\frac{\pi}{2}\right) - j\sin\left(\frac{\pi}{2}\right) = 1 - j$$

The magnitude and the phase of the system at the frequency of the input are

$$M = \sqrt{1+1} = \sqrt{2}$$

$$\alpha = \tan^{-1}\left(\frac{-1}{1}\right) = -45^{\circ} = -\frac{\pi}{4}$$

The steady-state response is then given by

$$y_{ss}(n) = 10\left(\sqrt{2}\right)\cos\left(\frac{\pi}{2}n + \frac{1}{2} - \frac{\pi}{4}\right) \qquad -\infty \le n \le +\infty$$

3.5 The Frequency Response of Discrete Systems

The frequency response of a discrete system is a set of output values when each value is obtained for a particular input $x(n)$ at a unique frequency value. At each frequency of the input, the output will have a change in its magnitude and phase.The frequency response $H(e^{j\theta})$ was derived in the process of finding the steady-state response in which we have found that

$$y_{ss}(n) = x(n)H\left(e^{j\theta n}\right)$$

if $x(n)$ is $e^{j\theta n}$ and we have defined the frequency response as

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m} \tag{3.16}$$

So if we have the impulse response of the system, we can use the above equation to find the frequency response. Also we can calculate the frequency response $H(e^{j\theta})$ as a function of θ from the difference equation directly.

Consider the first-order difference equation

$$y(n) + ay(n-1) = x(n)$$

Let the input be periodic of the form

$$x(n) = e^{j\theta n}$$

We have seen that the output will be

$$y_{ss}(n) = e^{j\theta n}H\left(e^{j\theta}\right)$$

Therefore, if we substitute this output in the difference equation, we will get

$$e^{j\theta n}H\left(e^{j\theta}\right) + ae^{j\theta(n-1)}H\left(e^{j\theta}\right) = e^{j\theta n}$$

If we cancel out the $e^{j\theta n}$ by dividing by $e^{j\theta n}$ (note that $e^{j\theta n}$ is never zero), we will have

$$H\left(e^{j\theta}\right)\left(1 + ae^{-j\theta}\right) = 1$$

or

$$H\left(e^{j\theta}\right) = \frac{1}{1 + ae^{-j\theta}}$$

which is the frequency response of the system.

We can now generalize to the general difference equation case

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k) \tag{3.17}$$

Similar to what we did before and by letting $x(n) = e^{j\theta n}$ and $y(n) = e^{j\theta n}H(e^{j\theta})$, we will get

$$e^{j\theta n}H\left(e^{j\theta}\right) - \sum_{k=1}^{N} a_k e^{j\theta(n-k)}H\left(e^{j\theta}\right) = \sum_{k=0}^{M} b_k e^{j\theta(n-k)}$$

We can factorize $H(e^{j\theta})$ and cancel the $e^{j\theta n}$ to get

$$H(e^{j\theta})\left[1 - \sum_{k=1}^{N} a_k e^{-j\theta k}\right] = \sum_{k=0}^{M} b_k e^{-j\theta k}$$

and finally the general frequency response is

$$H(e^{j\theta}) = \frac{\sum_{k=0}^{M} b_k e^{-j\theta k}}{1 - \sum_{k=1}^{N} a_k e^{-j\theta k}} \tag{3.18}$$

Example 3.5

Find the frequency response for the system

$$y(n) + .5y(n-1) + 6y(n-2) = x(n) + x(n-1)$$

Solution

With $x(n) = e^{j\theta n}$ and $y(n) = e^{j\theta n}\,H(e^{j\theta})$ the difference equation can be written as

$$e^{j\theta n}H\left(e^{j\theta}\right) + .5e^{j\theta(n-1)}H\left(e^{j\theta}\right) + 6e^{j\theta(n-2)}H\left(e^{j\theta}\right) = e^{j\theta n} + e^{j\theta(n-1)}$$

or

$$H\left(e^{j\theta}\right)\left[1 + .5e^{-j\theta} + 6e^{-j2\theta}\right] = 1 + e^{-j\theta}$$

The frequency response is then

$$H\left(e^{j\theta}\right) = \frac{1 + e^{-j\theta}}{1 + .5e^{-j\theta} + 6e^{-j2\theta}} = \frac{e^{j2\theta} + e^{j\theta}}{e^{j2\theta} + .5e^{j\theta} + 6}$$

3.5.1 Properties of the Frequency Response

We will discuss two properties here.

3.5.1.1 The Periodicity Property

We have seen the frequency response function $H(e^{j\theta})$ previously and it is presented here as

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}$$

For periodicity, $H(e^{j\theta})$ must be the same as $H(e^{j(\theta+2\pi)})$

$$H(e^{j(\theta+2\pi)}) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j(\theta+2\pi)m} = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}e^{-j2\pi m}$$

You can easily see that

$$e^{-j2\pi m} = \cos(2\pi m) - j\sin(2\pi m) = 1 - j(0) = 1$$

Thus

$$H\left(e^{j(\theta+2\pi)}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m} = H\left(e^{j\theta}\right)$$

This property is important since in one period we can learn all about the signal that is being transformed.

3.5.1.2 The Symmetry Property

To study this property we will consider $H(e^{j\theta})$ for real values. We have

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}$$

and

$$H\left(e^{-j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{j\theta m}$$

The magnitude of $e^{-j\theta m}$ is the same as the magnitude of $e^{j\theta n}$, thus

$$\left|H\left(e^{j\theta}\right)\right| = \left|H\left(e^{-j\theta}\right)\right|$$

This indicates that the magnitude of $H(e^{j\theta})$ has even symmetry. Also the phase of $e^{-j\theta m}$ is $-\theta m$ and that of $e^{j\theta n}$ is θm. Therefore the phase of $H(e^{j\theta})$ is the negative of the phase of $H(e^{-j\theta})$. This indicates that the phase of $H(e^{j\theta})$ has odd symmetry.

In general, from the frequency response for a given discrete system, you can tell if the system is able to pass a certain frequency range or reject another.

Example 3.6

Let

$$H\left(e^{j\theta}\right) = 1 - e^{-j\theta}$$

With respect to passing or rejecting frequencies, what kind of system is this?

Solution

$H(e^{j\theta}) = 1 - e^{-j\theta} = 1 - (\cos\theta - j\sin\theta) = 1 - \cos\theta + j\sin\theta$. The magnitude response is

$$\left|H\left(e^{j\theta}\right)\right| = \sqrt{\left(\sin^2\theta\right) + (1-\cos\theta)^2}$$

The plot of the magnitude of $H(e^{j\theta})$ vs. θ is shown in Figure 3.5. Notice that the system attenuates low-frequency signals but at frequencies approximately between $\pi/2$ and π this system does not attenuate as it does at frequencies near zero. Hence this system passes high frequencies and prevents low frequencies from passing.

Example 3.7

Consider the system with the frequency response function

$$H\left(e^{j\theta}\right) = \frac{1}{2 - 0.1e^{-j\theta}}$$

What frequencies does this system pass and what frequencies does it not?

Solution

The magnitude of the frequency response is

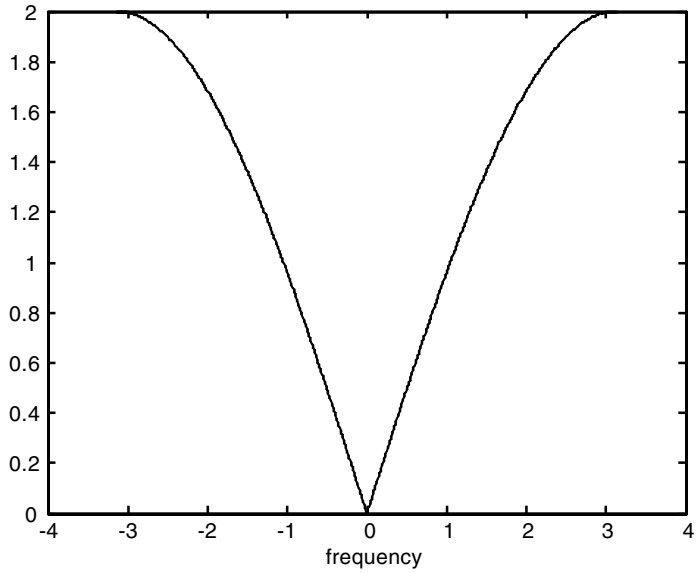

FIGURE 3.5 Magnitude plot for Example 3.6.

$$\left|H\left(e^{j\theta}\right)\right| = \left|\frac{1}{2-.1e^{-j\theta}}\right| = \left|\frac{1}{2-.1\cos\theta+.1j\sin\theta}\right| = \frac{1}{\sqrt{(2-.1\cos\theta)^2+(.1\sin\theta)^2}}$$

The plot of $|H(e^{j\theta})|$ is shown in Figure 3.6. It is clear from the plot that this system passes low frequencies and rejects high frequencies.

3.6 The Fourier Transform of Discrete Signals

The Fourier transform of a discrete signal $x(n)$ is $X(\theta)$ where $\theta = wT_s$. w is the analogue frequency of the continuous signal and T_s is the sampling period. The Fourier transform of discrete signals is defined as

$$X(\theta) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n} \tag{3.19}$$

and to find $x(n)$ given $X(\theta)$ we use the relation

$$x(n) = \frac{1}{2\pi}\int_{2\pi} X(\theta)e^{j\theta n}d\theta \tag{3.20}$$

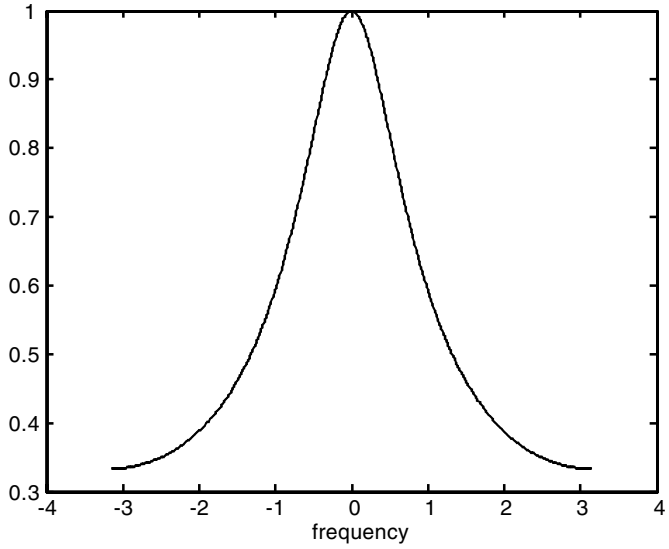

FIGURE 3.6 Magnitude plot for Example 3.7.

Note that $X(\theta)$ is a function of the continuous variable θ, the digital frequency, while $x(n)$ is a function of the discrete variable n.

We have seen that the frequency response function $H(e^{j\theta})$ was needed to calculate the steady-state response for a given discrete system. It also contains information about the system itself. The Fourier transform of discrete signals can be applied to signals that are periodic and not periodic. It will make the solution to difference equations much easier as we will see later.

Example 3.8

What is the Fourier transform of

$$x(n) = (.5)^n u(n)$$

Solution

The Fourier transform of $x(n)$ is $X(\theta)$ and it is

$$X(\theta) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\theta n} = \sum_{n=0}^{\infty} (.5)^n e^{-j\theta n} = \sum_{n=0}^{\infty} \left(.5e^{-j\theta}\right)^n = \left[\frac{1}{1-.5e^{-j\theta}}\right]$$

where here we used the geometric series sum

$$\sum_{n=0}^{\infty} a^n = \frac{1}{1-a} \qquad |a| < 1 \tag{3.21}$$

So $x(n) \leftrightarrow X(\theta)$ or we write the pairs $(.5)^n u(n) \leftrightarrow \dfrac{1}{1-.5e^{-j\theta}}$

Example 3.9

What is the Fourier transform of the impulse signal $A\delta(n)$?

Solution

For $x(n) = A\delta(n)$ we have

$$X(\theta) = \sum_{-\infty}^{\infty} A\delta(n)e^{-j\theta n} = A\delta(0)e^{-j\theta(0)} = A$$

Note that $\delta(n)$ is valid only at $n = 0$. Therefore,

$$x(n) = A\delta(n) \leftrightarrow X(\theta) = A$$

3.7 Convergence Conditions

The Fourier transform of $x(n)$ is given again as

$$X(\theta) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n}$$

where $X(\theta)$ is an infinite series and that series must converge for $X(\theta)$ to exist. For $X(\theta)$ to exist, it is necessary that $x(n)$ be summable. This means that

$$\sum_{n=-\infty}^{+\infty} |x(n)| < \infty$$

However, some signals such as the step signal are not summable. We will see how to deal with such signals later in this chapter.

3.8 Properties of the Fourier Transform of Discrete Signals

Table 3.2 lists some properties of the Fourier transform of discrete signals. We will prove some of them here.

3.8.1 The Periodicity Property

The Fourier transform of the discrete signal $x(n)$ is

$$X(\theta) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n}$$

Let $\theta = \theta + 2\pi$. Then

$$X(\theta + 2\pi) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n}e^{-j2\pi n}$$

But $e^{-j2n\pi} = \cos(2\pi n) - j\sin(2\pi n) = 1 - 0 = 1$. Thus,

$$X(\theta + 2\pi) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n} = X(\theta)$$

and $X(\theta)$ is periodic in θ with the period of 2π.

3.8.2 The Linearity Property

The Fourier transform of $x(n) = a_1x_1(n) + a_2x_2(n)$ is

$$X(\theta) = \sum_{n=-\infty}^{n=\infty} \left[a_1x_1(n) + a_2x_2(n)\right]e^{-j\theta n}$$

$$X(\theta) = \sum_{n=-\infty}^{\infty} a_1x_1(n)e^{-j\theta n} + \sum_{n=-\infty}^{\infty} a_2x_2(n)e^{-j\theta n}$$

which clearly results in

$$X(\theta) = a_1X_1(\theta) + a_2X_2(\theta)$$

3.8.3 The Discrete-Time Shifting Property

If $x(n)$ is shifted by n_0, then the Fourier transform of $x(n - n_0)$ is

$$\sum_{n=-\infty}^{+\infty} x\left(n-n_0\right)e^{-j\theta n}$$

If we let $n - n_0 = m$, then the Fourier transform of $x(n - n_0)$ becomes

$$\sum_{m=-\infty}^{+\infty} x(m)e^{-j\theta\left(m+n_0\right)} = e^{-j\theta n_0}\sum_{m=-\infty}^{\infty} x(m)e^{-j\theta m} = e^{-j\theta n_0}X(\theta)$$

The Fourier pairs are then

$$x\left(n-n_0\right) \leftrightarrow e^{-j\theta n_0}X(\theta)$$

3.8.4 The Frequency Shifting Property

The Fourier transform of $x(n)e^{j\theta_0 n}$ where θ_0 is the frequency shift is

$$\sum_{n=-\infty}^{+\infty}\left[e^{j\theta_0 n}x(n)e^{-j\theta n}\right] = \sum_{n=-\infty}^{+\infty} x(n)e^{-j(\theta-\theta_0)n} = X\left(\theta-\theta_0\right)$$

Therefore,

$$e^{j\theta_0 n}x(n) \leftrightarrow X\left(\theta-\theta_0\right)$$

3.8.5 The Reflection Property

Consider the reflected signal $x(–n)$ of $x(n)$. Its Fourier transform is

$$\sum_{n=-\infty}^{+\infty} x(-n)e^{-j\theta n}$$

Let $–n = m$. Then the transform becomes

$$\sum_{m=-\infty}^{+\infty} x(m)e^{j\theta m} = \sum_{m=-\infty}^{+\infty} x(m)e^{-jm(-\theta)} = X(-\theta)$$

Thus

$$x(-n) \leftrightarrow X(-\theta)$$

3.8.6 The Convolution Property

In real-time, with an input $x(n)$ and a system transfer function $h(n)$, the output of the system is then $y(n)$ and is given by

$$y(n) = x(n) * h(n) = \sum_{m=-\infty}^{+\infty} x(m)h(n-m) \tag{3.22}$$

Sometimes this summation becomes very complex to evaluate. Now let us take the Fourier transform on both sides to get

$$Y(\theta) = f\left[x(n) * h(n)\right] = f\left[\sum_{m=-\infty}^{\infty} x(m)h(n-m)\right]$$

where f indicates "the Fourier transform of." But according to the defining equation of the Fourier transform we have

$$f\left[\sum_{m=-\infty}^{\infty} x(m)h(n-m)\right] = \sum_{n=-\infty}^{+\infty}\left[\sum_{m=-\infty}^{\infty} x(m)h(n-m)\right]e^{-j\theta n}$$

$$= \sum_{m=-\infty}^{+\infty} x(m)\sum_{n=-\infty}^{+\infty} h(n-m)\,e^{-j\theta n}$$

where the last term above is obtained using the interchange of summation property.

Now let $k = n - m$ to get

$$Y(\theta) = \sum_{m=-\infty}^{+\infty} x(m)\sum_{k=-\infty}^{\infty} h(k)e^{-j\theta(k+m)} = \sum_{m=-\infty}^{+\infty} x(m)e^{-j\theta m}\sum_{k=-\infty}^{+\infty} h(k)e^{-j\theta k}$$

Finally the important relation results in the equation for the convolution in the frequency domain

$$Y(\theta) = X(\theta)H(\theta) \tag{3.23}$$

This result means that convolution in real-time is multiplication in the frequency domain, an easy-to-manipulate complex algebra. This result can be written as

$$x_1(n) * x_2(n) \leftrightarrow X_1(\theta)X_2(\theta) \tag{3.24}$$

Example 3.10

Consider the discrete system with $h(n) = \delta(n) + \delta(n-1)$ and an input $x(n) = (.5)^n\, u(n)$. Find the output $y(n)$ using the Fourier transform method.

Solution

The transform of $h(n)$, using Table 3.1 and the linearity and the shifting properties in Table 3.2 is

$$H(\theta) = 1 + e^{-j\theta}$$

and the Fourier transform of $x(n)$ is

$$X(\theta) = \frac{1}{1-(.5)e^{-j\theta}}$$

The output using the Fourier transform is then

$$Y(\theta) = X(\theta)H(\theta) = \left[1 + e^{-j\theta}\right]\left[\frac{1}{1-.5e^{-j\theta}}\right] = \frac{1}{1-.5e^{-j\theta}} + \frac{e^{-j\theta}}{1-.5e^{-j\theta}}$$

We can bring the above equation into the discrete domain. The first term is $(.5)^n\, u(n)$. We can use the shifting property in Table 3.2 for the second term and get $(.5)^{n-1}u(n-1)$. The output in the discrete domain is

$$y(n) = (.5)^n u(n) + (.5)^{n-1}u(n-1) \qquad n \geq 0$$

Example 3.11

Let us pass the input $x(n) = (.5)^n u(n)$ through two systems with

$$h_1(n) = h_2(n) = \delta(n) + \delta(n-1)$$

Find $y(n)$ using the Fourier transform method.

TABLE 3.1

Fourier Transform Pairs

$x(n)$	$X(\theta)$
$A\delta(n)$	A
$Au(n)$	$A\left[\frac{1}{1-e^{-j\theta}}+\sum_{\Delta=-\infty}^{\infty}\pi\delta(\theta-2\pi n)\right]$
A	$2\pi A\sum_{n=-\infty}^{+\infty}\delta(\theta-2\pi n)$
$a^n u(n)\ \|a\|<1$	$\frac{1}{1-ae^{-j\theta}}$
$na^n u(n)\ \|a\|<1$	$\frac{ae^{j\theta}}{\left(e^{j\theta}-a\right)^2}$
$e^{j\theta_0 n}$	$2\pi\sum_{n=-\infty}^{+\infty}\delta(\theta-\theta_0-2\pi n)$
$\cos(\theta_0 n)$	$\pi\sum_{n=-\infty}^{+\infty}\left[\delta(\theta-\theta_0-2\pi n)+\delta(\theta+\theta_0-2\pi n)\right]$
$\sin(\theta_0 n)$	$\frac{\pi}{j}\sum_{n=-\infty}^{+\infty}\delta(\theta-\theta_0-2\pi n)-\delta(\theta+\theta_0-2\pi n)$

TABLE 3.2

Fourier Transform Properties

Real-time	Fourier domain
$a_1x_1(n)+a_2x_2(n)$	$a_1X_1(\theta)+a_2X_2(\theta)$
$x(n-n_0)$	$e^{-j\theta n_0}\,X(\theta)$
$e^{j\theta_0 n}\,x(n)$	$X(\theta-\theta_0)$
$nx(n)$	$j\frac{d}{d\theta}[X(\theta)]$
$x_1(n)*x_2(n)$	$X_1(\theta)\,X_2(\theta)$
$x_1(n)x_2(n)$	$\frac{1}{2\pi}\int_{-\infty}^{+\infty}X_1(m)X_2(\theta-m)dm$
$x(-n)$	$X+\pi(-\theta)$
$\sum_{n=-\infty}^{\infty}\|x(n)\|^2$	$\frac{1}{2\pi}\int_{-\pi}^{+\pi}\|X(\theta)\|^2 d\theta$

Solution

As in Example 3.10,

$$X(\theta)=\frac{1}{1-.5e^{-j\theta}}$$

and

$$H_1(\theta) = H_2(\theta) = 1 + e^{-j\theta}$$

If we call the output of the first system $y_1(n)$ then

$$Y_1(\theta) = X(\theta)H_1(\theta) = \left[\frac{1}{1-.5e^{-j\theta}}\right]\left[1+e^{-j\theta}\right] = \frac{1}{1-.5e^{-j\theta}} + \frac{e^{-j\theta}}{1-.5e^{-j\theta}}$$

The output of the second system will have the output of the first system as its input. Therefore, the output of the second system $y(n)$ is

$$Y(\theta) = Y_1(\theta)H_2(\theta) = \left[\frac{1}{1-0.5e^{-j\theta}} + \frac{e^{-j\theta}}{1-.5e^{-j\theta}}\right]\left[1+e^{-j\theta}\right]$$

$$= \frac{1}{1-0.5e^{-j\theta}} + \frac{e^{-j\theta}}{1-0.5e^{-j\theta}} + \frac{e^{-j\theta}}{1-.5e^{-j\theta}} + \frac{e^{-2j\theta}}{1-.5e^{-j\theta}}$$

By using Tables 3.1 and 3.2 we get

$$y(n) = (.5)^n u(n) + 2(.5)^{n-1}u(n-1) + (.5)^{n-2}u(n-2) \qquad n \geq 0$$

3.9 Parseval's Relation and Energy Calculations

We have seen in Chapter 1 that the total energy in the signal $x(n)$ is

$$E(n) = \sum_{n=-\infty}^{+\infty} |x(n)|^2$$

The Parseval's relation for discrete signals says that this energy can be computed using the Fourier transform as

$$E(n) = \sum_{n=-\infty}^{+\infty} |x(n)|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} |X(\theta))|^2 d\theta$$

Example 3.12

Find the energy in the signal

$$x(n) = \delta(n) + \delta(n-1)$$

Solution

We have seen that

$$E(n) = \sum_{n=-\infty}^{+\infty} |x(n)|^2 = (1)^2 + (1)^2 = 2$$

We can use the Parseval's theorem to find the energy as

$$E(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\left(1 + e^{-j\theta}\right)\right|^2 d\theta$$

$$E(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|(1 + \cos\theta - j\sin\theta)\right|^2 d\theta = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left((1+\cos\theta)^2 + \sin^2\theta\right) d\theta$$

If we simplify we arrive at

$$E(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} 1 d\theta + \frac{1}{2\pi}\int_{-\pi}^{\pi} 2\cos\theta \; d\theta + \frac{1}{2\pi}\int_{-\pi}^{\pi} \cos^2\theta \; d\theta + \frac{1}{2\pi}\int_{-\pi}^{\pi} \sin^2\theta \; d\theta$$

$$E(n) = \frac{1}{2\pi}(\pi + \pi) + \frac{2}{2\pi}\left[\sin(\pi) - \sin(-\pi)\right] + \frac{1}{2\pi}(\pi + \pi) = 1 + 0 + 1 = 2$$

3.10 Numerical Evaluation of the Fourier Transform of Discrete Signals

We should keep in mind that the Fourier transform of $x(n)$ is $X(\theta)$ which is a complex function in θ. MATLAB can be used to calculate the complex values for X at each θ value. The magnitude of X, the phase of X, the real part of X, and the imaginary part of X can be calculated and plotted. We also know that $X(\theta)$ is periodic of period 2π. Since $X(\theta)$ is symmetric when $x(n)$ is real, we can plot $|X(\theta)|$ in the interval $[0,\pi]$ and know all about $x(n)$ in that range.

Example 3.13

Find the Fourier transform of $x(n) = (.1)^n \; n \geq 0$. Plot the magnitude and the phase of $X(\theta)$, the Fourier transform of $x(n)$.

Solution

The Fourier transform is given by

$$X(\theta) = \sum_{m=-\infty}^{+\infty} x(m)e^{-j\theta m} = \sum_{m=0}^{+\infty} (.1)^m e^{-j\theta m} = \sum_{m=0}^{\infty} \left(0.1e^{-j\theta}\right)^m = \frac{1}{1-0.1e^{-j\theta}}$$

To use MATLAB to plot $|X(\theta)|$ and the phase of $X(\theta)$ vs. θ we need to put $X(\theta)$ in the form

$$X(\theta) = \frac{a_0 + a_1 e^{-j\theta} + a_2 e^{-2j\theta} + a_3 e^{-3j\theta} + \ldots}{1 + b_1 e^{-j\theta} + b_2 e^{-2j\theta} + b_3 e^{-3j\theta} + \ldots} \tag{3.25}$$

We then can use the MATLAB function

```
X= freqz(n,d,df)
```

The function `freqz` receives the numerator vector `n`, the denominator vector `d` and the digital frequency vector, and sends back the complex values of $X(\theta)$ in the vector X. Then we can use the MATLAB functions `abs` and `angle` to find the magnitude and the angle of the frequency response. In our example

$$X(\theta) = \frac{1}{1-.1e^{-j\theta}}$$

where `n = [1]; d = [1 – .1]`. We will use 401 frequency points in the range from 0 to π radians. The MATLAB script is shown next.

```
df = 0:pi/400:pi; % 401 frequency points
n = [1]
d = [1 -.1]
x = freqz(n, d, df)
subplot(2, 1, 1);
plot(df, abs(x));
ylabel('Magnitude plot');
subplot(2, 1, 2);
plot(df, angle(x));
ylabel('Phase plot'); xlabel('Frequency');
```

The plots are shown in Figure 3.7.

Example 3.14

Find the Fourier transform of $x(n) = \delta(n) + \delta(n-1) + \delta(n-2) + \delta(n-3)$. Plot the magnitude and the phase of $X(\theta)$.

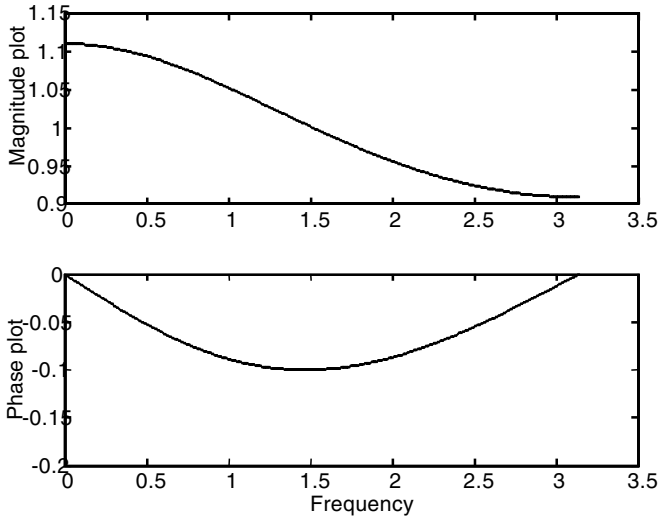

FIGURE 3.7 Frequency response for Example 3.13.

Solution

Since $x(n)$ is finite and defined only at the $n = 0, 1, 2$ and 3, we can still use the MATLAB function `freqz` and we can use the following script to plot the magnitude and phase of $X(\theta)$ at 401 frequency points.

```
df = 0:pi/400:pi; % 401 frequency points
n = [1 1 1 1]
d = [1]
x = freqz(n, d, df)
subplot(2, 1, 1);
plot(df, abs(x));
ylabel('Magnitude plot');
subplot(2, 1, 2);
plot(df, angle(x));
ylabel('Phase plot'); xlabel('Frequency');
```

The plots are in Figure 3.8.

However, we can take a different approach and implement the defining equation for $X(\theta)$ directly. We have

$$X(\theta) = \sum_{m=-\infty}^{+\infty} x(m)e^{-j\theta m}$$

Assume that $x(n)$ is valid for $n_1 \le n \le n_2$ and we want to calculate $X(\theta)$ at $p + 1$ points. We are also interested in the frequency range from 0 to π. In this case, the frequency spacing is taken as

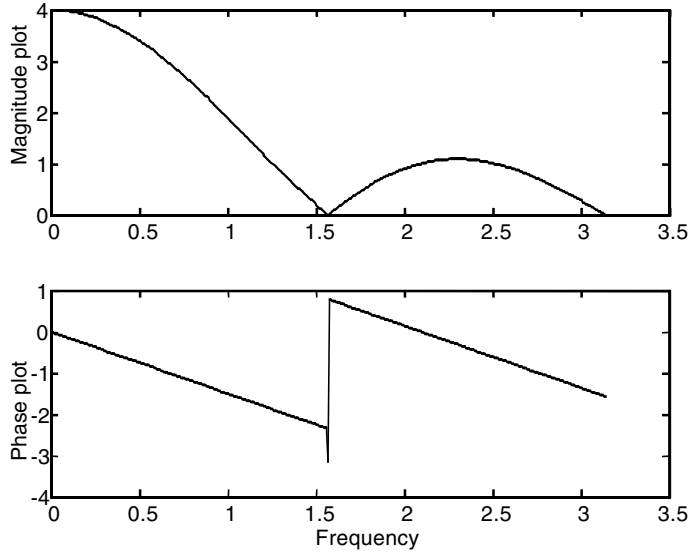

FIGURE 3.8 Frequency response for Example 3.14.

$$df = (\pi/p)m \text{ for } m = 0, 1, 2, \ldots p$$

The frequency response becomes

$$X(m) = \sum_{r=1}^{N} e^{-j(\pi/p)mn_r} x(n_r) \text{ for } m = 0, 1, \ldots p \tag{3.26}$$

where N is the number of samples for $x(n)$. Note that if $x(n_r)$ is a column vector having N rows and 1 column, and $X(m)$ is also a column vector having $(p + 1)$ rows and 1 column, then the summation

$$\sum_{r=1}^{N} e^{-j(\pi/p)mn_r}$$

should be a matrix of $p + 1$ rows and N columns. This requires the vector m to be a column vector with $p + 1$ rows and 1 column and the n_r vector to be a row vector with 1 row and N columns. In MATLAB, if we enter data as row vectors then Equation (3.26) can be written as

$$X = e^{-j(\pi/p)m'n} x' \tag{3.27}$$

where $'$ indicates transpose of a matrix or a vector in MATLAB. To implement this method for the example at hand we will write the script

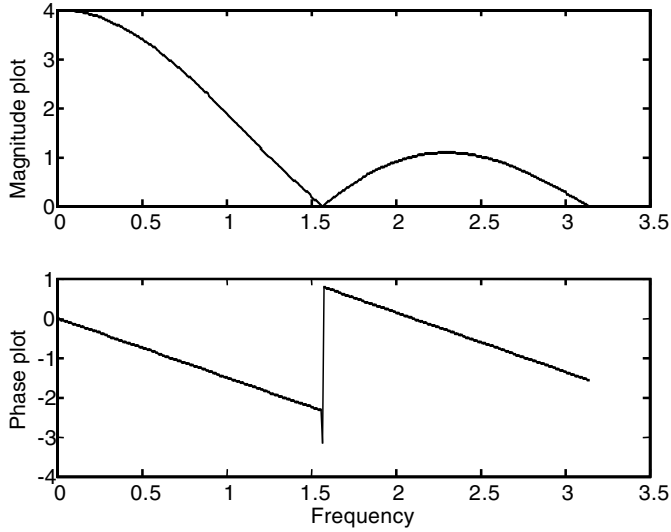

FIGURE 3.9 Frequency response for Example 3.14.

```
m =[0:400]  % 401 points for the transform
n = [0 1 2 3];  % the index for which x(n) is defined.
x = [1 1 1 1];% the values of x(n).
df = (pi/400)*m;
X = ((exp (-j*pi/400)).^(m'*n))*x'; % ' indicates transpose
subplot (2, 1, 1);
plot (df,abs(X))
ylabel('Magnitude plot');
subplot(2, 1, 2);
plot(df, angle(X));
ylabel('Phase plot'); xlabel('Frequency');
```

and the plots are shown in Figure 3.9. Note that this example was straightforward. The method described in the example will be very useful, especially if you are to find the Fourier transform of $x(n) = n(.1)^n \cos(n)$ for $0 \le n \le 100$ or for $0 \le n \le 1000$ or for a bigger range. We will look at an example of this form in the EOCE section later in the chapter.

3.11 Some Insights: Why Is This Fourier Transform?

3.11.1 The Ease in Analysis and Design

With the Fourier transform we are able to identify the frequency contents of the input signal $x(n)$, both the magnitude and the phase spectrum. It may

not be possible to predict what frequencies such $x(n)$ contains, especially if they are given as a plot. Knowing the frequency contents of the input signal gives us more insights into the way we analyze and design linear systems. We will know what frequencies will pass and what frequencies will not pass through a particular system. Some convolution sums are very difficult to evaluate in discrete real-time. With the help of the Fourier transform things become much easier.

3.11.2 Sinusoidal Analysis

If the system $H(\theta)$ is subject to a sinusoidal input signal

$$x(n) = X\cos(\theta t + \varphi)$$

the steady-state output, $y_{ss}(n)$, of the system can be evaluated as

$$y_{ss}(n) = |x(n)||H(\theta)|\cos(\theta n + \varphi + \alpha)$$

or

$$y_{ss}(n) = XM\cos(\theta n + \varphi + \alpha)$$

where X is the magnitude of the input $x(n)$, M is the magnitude of the frequency response of the discrete system $H(\theta)$, φ is the phase of $H(\theta)$ at a particular given θ and α is the phase of $H(\theta)$.

This says that the output of a linear time-invariant system, if subject to a sinusoidal input, will have a steady-state solution equal to the magnitude of the input signal multiplied by the magnitude of the transfer function of the system evaluated at the frequency of the input signal and shifted by the phase angle of the transfer function evaluated at the input frequency as well.

Note also that the frequency of the output is the same as the frequency of the input. This means that the system is linear. If the system is not linear, the output frequency will *differ* from the input frequency.

3.12 End of Chapter Examples

EOCE 3.1

Consider the discrete system given below

$$h(n) = .1\delta(n) + .2\delta(n-2) + .5\delta(n-3)$$

Plot the frequency response, the phase and magnitude of $H(\theta)$.

Solution

The impulse response $h(n)$ is defined only at $n = 0$, $n = 2$ and $n = 3$. With

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}$$

we have

$$H\left(e^{j\theta}\right) = .1e^{-j\theta(0)} + .2e^{-j2\theta} + .5e^{-j3\theta} = .1 + .2e^{-2j\theta} + .5e^{-3j\theta}$$

The plot of the phase and magnitude for $H(e^{j\theta})$ can be obtained using MATLAB and we will take the range from 0 to 4π to show that $H(e^{j\theta})$ is periodic in θ of period 2π. The script is

```
m=0:400;
df = (m/400)*pi*4; % Range from 0 to 4pi
n = [.1 0 .2 .5]
d = [1]
x = freqz(n, d, df)
subplot(2, 1, 1);
plot(df, abs(x));
ylabel('Magnitude plot');
subplot(2, 1, 2);
plot(df, angle(x));
ylabel('Phase plot'); xlabel('Frequency');
```

The plot is shown in Figure 3.10.

EOCE 3.2

Consider the difference equation

$$y(n) + .1y(n-1) + .2y(n-2) = x(n)$$

Find the frequency response and plot its magnitude and phase.

Solution

With $x(n) = e^{j\theta n}$ and $y(n) = e^{j\theta n}H(e^{j\theta})$ we have from the given equation that

$$e^{j\theta n}H\left(e^{j\theta}\right) + .1e^{j\theta(n-1)}H\left(e^{j\theta}\right) + .2e^{j\theta(n-2)} = e^{j\theta n}$$

When we factor out $H(e^{j\theta})$ we will get

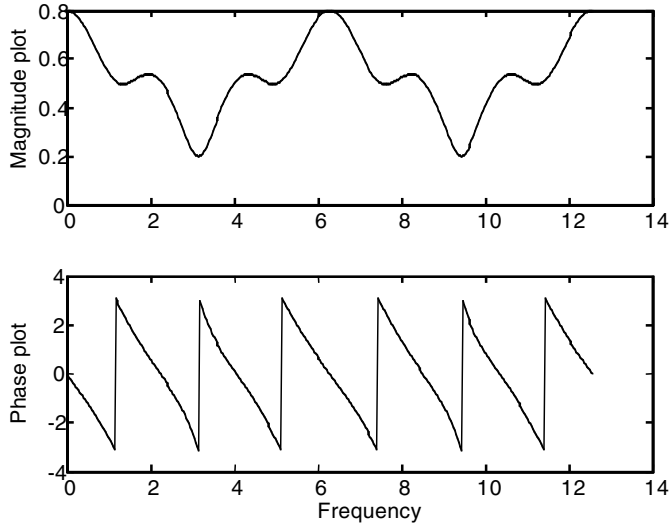

FIGURE 3.10 Frequency response for EOCE 3.1.

$$H\left(e^{j\theta}\right) = \left(1 + .1e^{-j\theta} + .2e^{-2j\theta}\right) = 1$$

and

$$H\left(e^{j\theta}\right) = \frac{1}{1 + .1e^{-j\theta} + .2e^{-2j\theta}}$$

Next we use MATLAB to plot the magnitude and the phase of this frequency response in the range from 0 to 2π radians.

```
m=0:400;
df = (m/400)*2*pi; % Range from 0 to 2pi
n = [1]
d = [1 .1 .2]
x = freqz(n, d, df)
subplot(2, 1, 1);
plot(df, abs(x));
ylabel('Magnitude plot');
subplot(2, 1, 2);
plot(df, angle(x));
ylabel('Phase plot'); xlabel('Frequency');
```

The plot is shown in Figure 3.11.

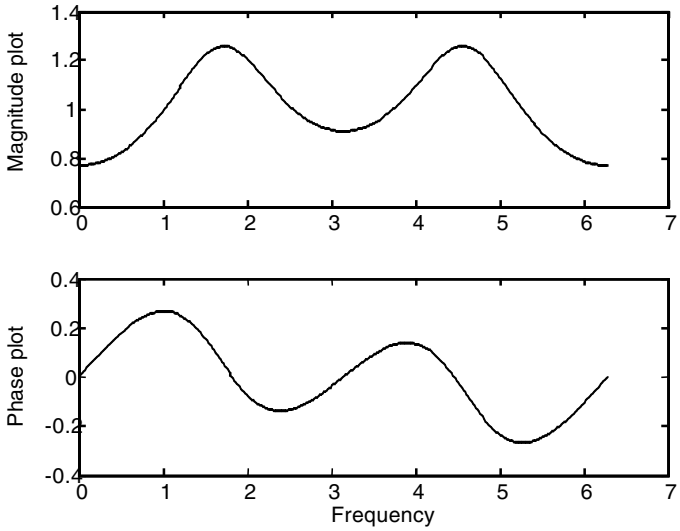

FIGURE 3.11 Frequency response for EOCE 3.2.

EOCE 3.3

Find the frequency response of the system with

$$h(n) = n(.1)^n \sin(n) \text{ for } 0 \le n \le 100$$

Solution

We know that

$$H\left(e^{j\theta}\right) = \sum_{m=-\infty}^{+\infty} h(m)e^{-j\theta m}$$

and this is not a difference equation so we can use the MATLAB function `freqz` to find the frequency response. It is also not easy to find $H(e^{j\theta})$ analytically. So we use the method we established earlier in the chapter. In this case n_1 is 0 and n_2 is 100. We will take 400 points, so p is 400. The frequency resolution is taken as

$$df = \frac{\pi}{p} m \qquad m = 0, 1, 2, \ldots p$$

We will use MATLAB to implement $H = e^{-j(\pi/p)m'n}h'$ as we did earlier for the Fourier transform approximation. The MATLAB script is

```
m = [0:400]; n = [0:100]; p =400;
h = n.*(0.6.^n).*sin(n);
```

```
df = (pi/p)*m;
H = ((exp (-j*pi/400)).^(m'*n))*h';
subplot(2, 1, 1);  plot(df, abs(H));
ylabel('Magnitude plot');
subplot(2, 1, 2);  plot(df, angle(H));
ylabel('Phase plot');
xlabel('Range from 0 to pi');
```

The plots are in Figure 3.12.

EOCE 3.4

Consider the discrete system described by the impulse response

$$h(n) = (.9)^n u(n)$$

Find that steady-state response of the system when

$$x(n) = 2\cos\left(\frac{\pi}{2}n\right) + 4\sin\left(\frac{\pi}{2}n\right)$$

Solution

First we need to find $H(e^{j\theta})$, then we can use the formula

$$y_{ss}(n) = \left|H\left(e^{j\theta}\right)\right|\left|x(n)\right|\cos(\theta n + \varphi + \alpha)$$

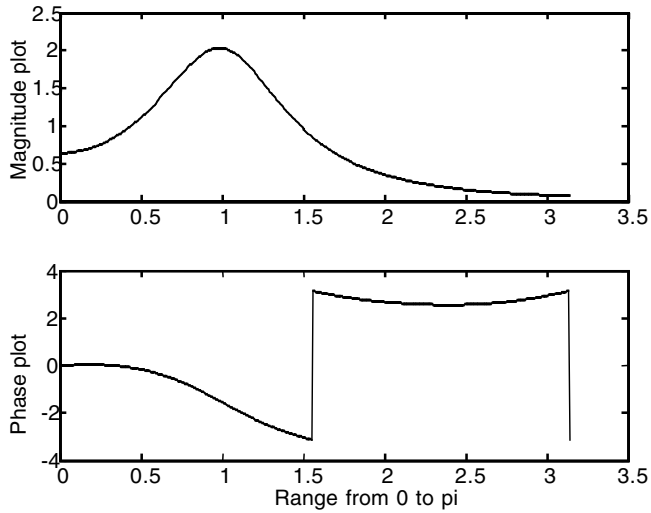

FIGURE 3.12 Frequency response for EOCE 3.3.

to find the steady-state response. The frequency response is

$$H\left(e^{j\theta}\right)=\sum_{m=0}^{\infty}(.9)^m e^{-j\theta m}=\sum_{m=0}^{\infty}\left((.9)e^{-j\theta}\right)^m=\frac{1}{1-.9e^{-j\theta}}=\frac{1}{1-.9\cos\theta+.9\sin\theta}$$

The input can be divided into two separate inputs; $x_1(n)$ and $x_2(n)$, where

$$x_1(n)=2\cos\left(\frac{\pi}{2}n\right)\ \text{ and }\ x_2(n)=4\sin\left(\frac{\pi}{2}n\right)$$

Then we can use superposition to find $y_{ss}(n) = y_{ss1}(n) + y_{ss2}(n)$ where $y_{ss1}(n)$ is the output due to $x_1(n)$ and $y_{ss2}(n)$ is the output due to $x_2(n)$.

For $x_1(n)$= 2cos(π/2 n) we have

$$\left|x_1(n)\right|=2,\ \theta=\frac{\pi}{2},\text{and}\ \varphi=0$$

With

$$\left|H\left(e^{j\theta}\right)\right|=\frac{1}{\sqrt{(1-.9\cos\theta)^2+(.9\sin\theta)^2}}$$

and for $\theta = \pi/2$ the magnitude of the frequency response becomes

$$\left|H\left(e^{j\frac{\pi}{2}}\right)\right|=\frac{1}{\sqrt{(1-.9(0))^2+(.9(1))^2}}=\frac{1}{\sqrt{1+.81}}=\frac{1}{\sqrt{2.81}}$$

and the phase is $0 - \tan^{-1}(.9/1) = 0.7328$. The steady-state solution is then

$$y_{ss1}(n)=2\left(\frac{1}{\sqrt{2.81}}\right)\cos\left(\frac{\pi}{2}n+0-\tan^{-1}\left(\frac{.9}{1}\right)\right)\quad -\infty\le n\le+\infty$$

For $x_2(n) = 4\sin(\pi/2)$, we need to write both inputs using the same reference. We write this second input as

$$x_2(n)=4\cos\left(\frac{\pi}{2}n-\frac{\pi}{2}\right)$$

The only difference now is in the phase shift of the input and its magnitude.

$$\left|x_2(n)\right|=4,\text{and}\ \varphi=-\frac{\pi}{2}$$

The output is then

$$y_{ss2}(n) = 4\left(\frac{1}{\sqrt{2.81}}\right)\cos\left(\frac{\pi}{2}n - \frac{\pi}{2} - \tan^{-1}\left(\frac{.9}{1}\right)\right) \quad -\infty \le n \le +\infty$$

Finally

$$y_{ss}(n) = y_{ss1}(n) + y_{ss2}(n) =$$
$$= \left(\frac{1}{\sqrt{2.81}}\right)\left[2\cos\left(\frac{\pi}{2}n + 0 - 0.7328\right) + 4\cos\left(\frac{\pi}{2}n - \frac{\pi}{2} - 0.7328\right)\right]$$

EOCE 3.5

The Fourier transform of $\alpha x_1(n) + \beta x_2(n)$ is $\alpha X_1(\theta) + \beta X_2(\theta)$. Use MATLAB to show this property.

Solution

Consider the two signals

$$x_1(n) = 2\delta(n) + 3\delta(n-1) - 4\delta(n-3)$$

and

$$x_2(n) = 2\delta(n) + 3\delta(n-3)$$

Let $\alpha = 2$ and $\beta = 3$. Now consider the following MATLAB script to obtain the proof .

```
m= 0: 400; k=(pi/400)*m;
n=0:2; x1=[2 3 4]; x2=[2 0 3];
x = 2*x1 + 3*x2; % the sum of the signals
X = ((exp (-j*pi/400)).^(m'*n))*x';
X1 = ((exp (-j*pi/400)).^(m'*n))*x1';
X2 = ((exp (-j*pi/400)).^(m'*n))*x2';
plot(k, abs(X)-abs(2*X1+3*X2));
xlabel('Frequency');
ylabel('Error in approximation');
```

The plot is shown in Figure 3.13.

EOCE 3.6

Use MATLAB to prove the time-shifting property of the Fourier transform of discrete signals.

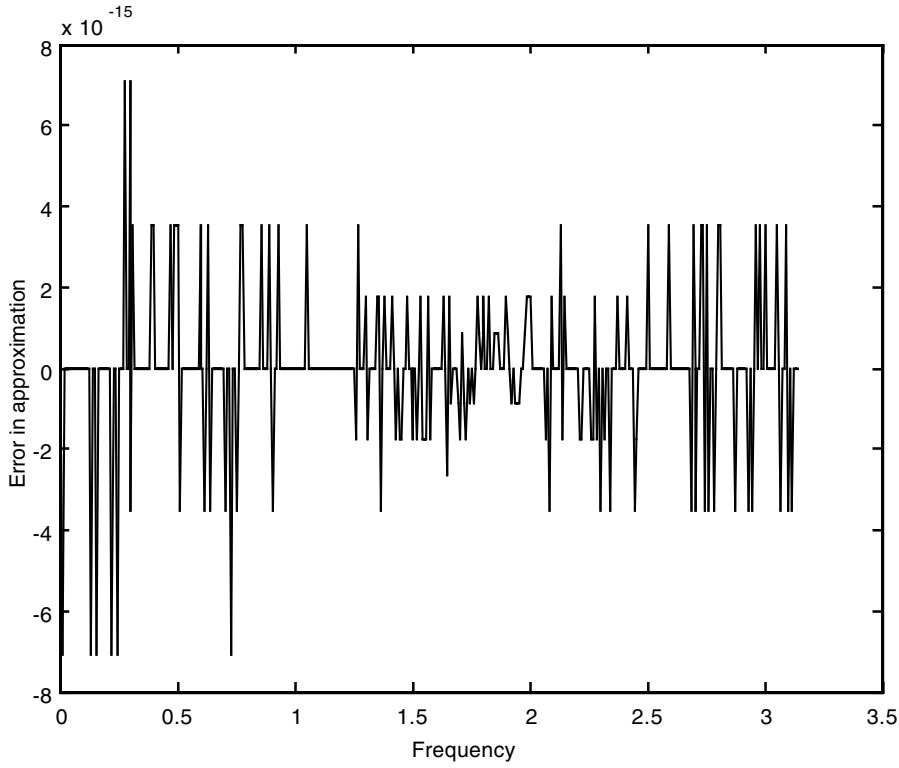

FIGURE 3.13 Error for EOCE 3.5.

Solution

Consider the signal

$$x(n) = \delta(n) + 2\delta(n-1) + 3\delta(n-2)$$

We know that by shifting $x(n)$ 3 units, for example, only the index n for $x(n)$ will change. The MATLAB script is

```
m =0:400;
k = (pi/400)*m;
n = [0 1 2];
x = [1 2 3];
X = ((exp (-j*pi/400)).^(m'*n))*x';
% next we shift X, the transfer of x(n), by 3 units.
X = (exp(-j*3).^k').*X;
%let us shift x in real-time by 3 units
```

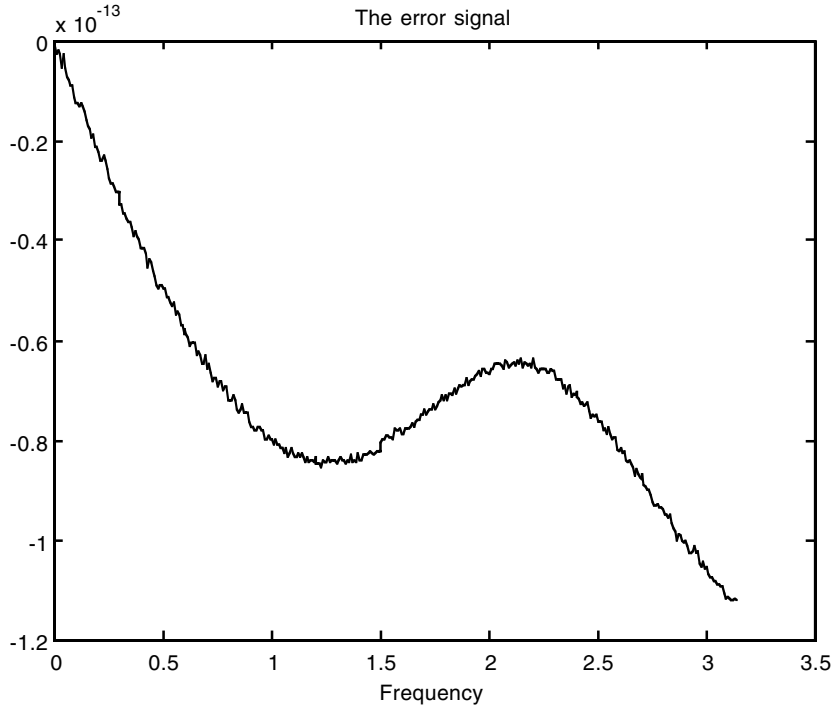

FIGURE 3.14 Error for EOCE 3.6.

```
x_shifted = x;  % no change in value
n_shifted = n+3; %index shift only
X_shifted = ((exp(-j*pi/400)).^(m'*n_shifted))*x_shifted';
plot(k, abs(X_shifted)-abs(X)); title('The error signal');
xlabel('Frequency');
```

The plot is shown in Figure 3.14.

EOCE 3.7

Consider the following frequency response

$$H\left(e^{j\theta}\right) = \frac{1}{1-.9e^{-j\theta}}$$

1. What is the steady-state output if the input to the system is

$$x(n) = \frac{1}{2}\cos\left(\frac{\pi}{4}n\right)$$

2. Use MATLAB to find the response to this input.

Solution

1. The frequency response at the input frequency is

$$H\left(e^{j\frac{\pi}{4}}\right)=\frac{1}{1-.9\frac{\sqrt{2}}{2}+.9\frac{\sqrt{2}}{2}j}$$

and the magnitude at the input frequency is

$$\left|H\left(e^{j\frac{\pi}{4}}\right)\right|=\frac{1}{\sqrt{\left(1-.9\frac{\sqrt{2}}{2}\right)^2+\left(.9\frac{\sqrt{2}}{2}\right)^2}}=1.36$$

The angle is

$$0-\tan^{-1}\left(\frac{\frac{.9\sqrt{2}}{2}}{1-.9\frac{\sqrt{2}}{2}}\right)=-1.05$$

and the steady-state solution is

$$y_{ss}(n)=\left(\frac{1}{2}\right)(1.36)\cos\left(\frac{\pi}{4}n+(-1.05)\right)$$

2. We can use MATLAB and write the following script:

```
n =0:10;  % 11 samples only
x = (1/2)*cos(pi/4*n);
yssanalyt=0.5*1.36*cos(pi/4*n-1.05);
b = [1]; %numerator vector in the frequency response
a = [1 -.9]; %denominator vector in the frequency response
ymatlab = filter(b, a, x);
subplot(3, 1, 1), stem (n,x); ylabel('Input signal')
subplot(3, 1, 2), stem (n, yssanalyt); ylabel('Output
 analytical')
subplot(3, 1, 3), stem (n, ymatlab); ylabel('Output using
 MATLAB');
xlabel('n');
```

The plots are shown in Figure 3.15. You can see that as n increases in the solution using MATLAB for the total response, the outputs in the steady-state solution and the MATLAB total response do match.

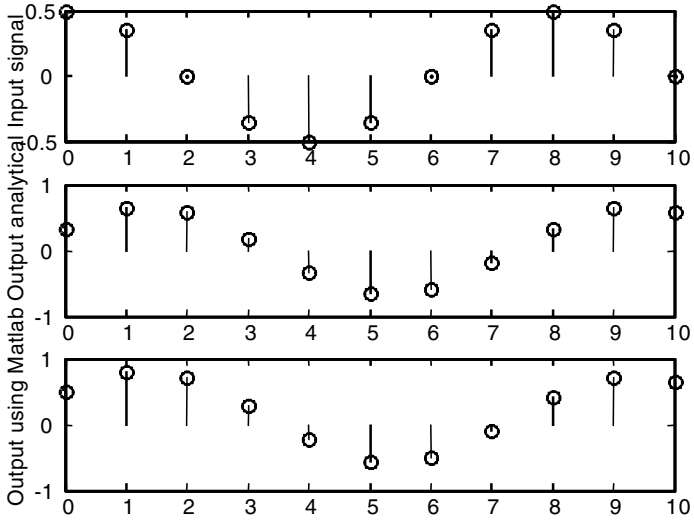

FIGURE 3.15 Signals for EOCE 3.7.

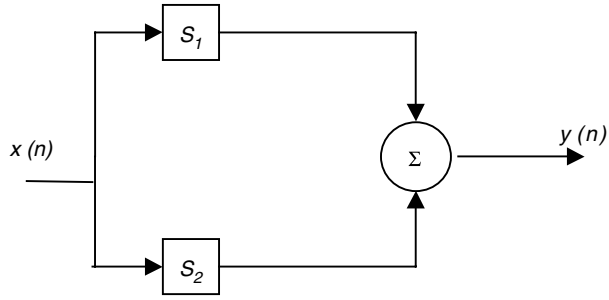

FIGURE 3.16 System for EOCE 3.8.

EOCE 3.8

Consider the system shown in Figure 3.16. Let S_1 be represented by

$$y_1(n) + .1y_1(n-1) = x(n)$$

and S_2 be represented by

$$y_2(n) + .1y_2(n-1) + 2y(n-2) = x(n)$$

Find the output using MATLAB if

$$x(n) = \frac{1}{2}\cos\left(\frac{\pi}{4}n - \pi\right)$$

Solution

The frequency response of system 1 is obtained as

$$H_1\left(e^{j\theta}\right) = \frac{1}{1+.1e^{-j\theta}}$$

and the frequency response of system 2 is obtained as

$$H_2\left(e^{j\theta}\right) = \frac{1}{1+.1e^{-j\theta}+2e^{-j2\theta}}$$

Systems 1 and 2 are connected in parallel and $y(n)$ can be calculated as the output of the whole system. Thus

$$H_1\left(e^{j\theta}\right) + H_2\left(e^{j\theta}\right) = \frac{1+.1e^{-j\theta}+2e^{-j2\theta}+1+.1e^{-j\theta}}{\left(1+.1e^{-j\theta}\right)\left(1+.1e^{-j\theta}+2e^{-j2\theta}\right)}$$

$$= \frac{2+.2e^{-j\theta}+2e^{-j2\theta}}{1+.2e^{-j\theta}+2.01e^{-j2\theta}+.2e^{-j3\theta}}$$

The MATLAB script used to find the output is

```
n =0:20;  % 21 samples only
x = (1/2)*cos(pi/4*n-pi);
b = [2 .2 2]; %numerator vector in the frequency response
a = [1 .2 2.01 .2]; %denominator vector in the frequency
 response
y = filter(b, a, x);
subplot(2, 1, 1), stem (n,x); ylabel('Input signal')
subplot(2, 1, 2), stem (n, y); ylabel('Output using MATLAB')
xlabel('n');
```

The plot is shown in Figure 3.17.

EOCE 3.9

Consider the system in Figure 3.18. System S_1 is represented by

$$y_1(n) + y_1(n-1) = x(n)$$

and S_2 is represented by

$$y_2(n) - y_2(n-1) = y_1(n)$$

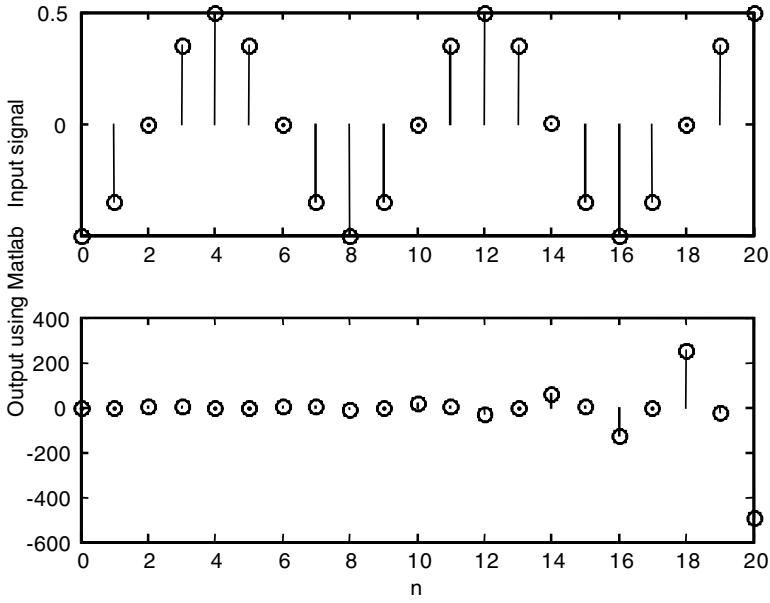

FIGURE 3.17 Signals for EOCE 3.8.

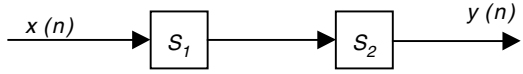

FIGURE 3.18 System for EOCE 3.9.

Find the steady-state response $y(n)$ if $x(n)$ is

$$x(n) = .5\cos\left(n\frac{\pi}{4}\right)$$

Find also the response to this input using MATLAB. Note that using MATLAB you will get the total response. The steady-state response is obtained by observing the MATLAB output for large n.

Solution

Since S_1 and S_2 are connected in series, the steady-state $y_{ss}(n)$ can be thought of as the output to the system $H_1(e^{j\theta})\,H_2(e^{j\theta})$ with the input $x(n)$.

$$H_1\left(e^{j\theta}\right) = \frac{1}{1+e^{-j\theta}} \quad \text{and} \quad H_2\left(e^{j\theta}\right) = \frac{1}{1-e^{-j\theta}}$$

The whole system is now

$$H\left(e^{j\theta}\right) = H_1\left(e^{j\theta}\right)H_2\left(e^{j\theta}\right) = \frac{1}{1+e^{-j\theta}}\frac{1}{1-e^{-j\theta}} = \frac{1}{1-e^{-2j\theta}} = \frac{1}{1-\cos 2\theta + j\sin 2\theta}$$

The frequency response at the input frequency is given by

$$H\left(e^{j\frac{\pi}{4}}\right) = \frac{1}{1-\cos\left(2\frac{\pi}{4}\right)+j\sin\left(2\frac{\pi}{4}\right)} = \frac{1}{1+j}$$

The magnitude of the frequency response is $1/\sqrt{2}$ and its phase is $(0 - \tan^{-1}(1/1)) = -\pi/4$. Thus the steady-state response for the whole system is

$$y_{ss}(n) = \left(\frac{1}{2}\right)\left(\frac{1}{\sqrt{2}}\right)\cos\left(n\frac{\pi}{4}-\frac{\pi}{4}\right) = \frac{1}{2\sqrt{2}}\cos\left(\frac{\pi}{4}(n-1)\right)$$

We see that the output is the input reduced in magnitude by $1/\sqrt{2}$ and shifted by $-\pi/4$.

We can use MATLAB to plot $y_{ss}(n)$ and the response to the given input. We write the script

```
n =0:10;  % 11 samples only
x = (1/2)*cos(pi/4*n);
yssanalyt=(1/2*sqrt(2))*cos(pi/4*n-pi/4);
b = [1]; %numerator vector in the frequency response
a = [1 0 -1]; %denominator vector in the frequency response
ymatlab = filter(b, a, x);
subplot(3, 1, 1), stem (n,x); ylabel('Input signal')
subplot(3, 1, 2), stem (n, yssanalyt); ylabel('Output
 analytical')
subplot(3, 1, 3), stem (n, ymatlab); ylabel('Output using
 MATLAB');
xlabel('n');
```

The plots are shown in Figure 3.19.

EOCE 3.10

Two systems $h_1(n)$ and $h_2(n)$ are connected in series with an input $x(n) = \delta(n)$. Find the output $y(n)$ for $n \geq 0$ if $h_1(n) = (1/2)^n u(n)$ and $= h_2(n) = (1/4)^n u(n)$.

Solution

In real-time $y(n) = (h_1(n) * h_2(n)) * x(n)$. But we can use the Fourier transform to write

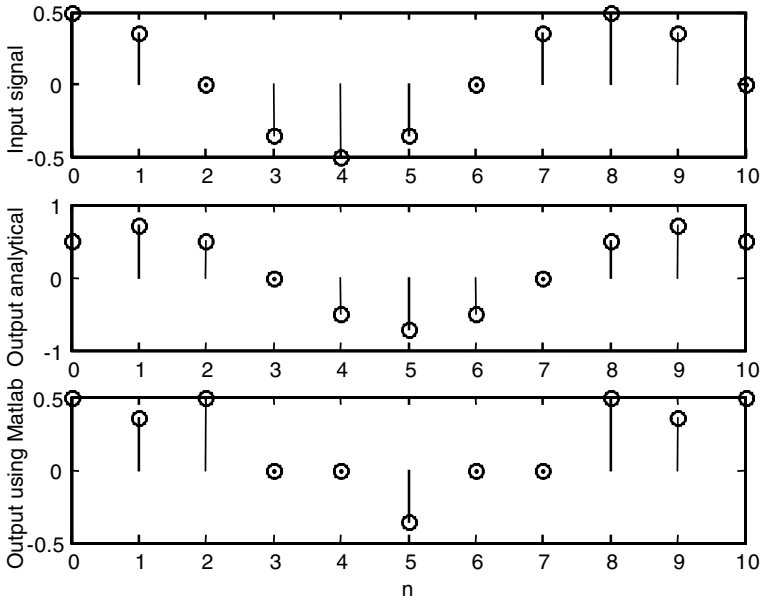

FIGURE 3.19 Plots for EOCE 3.9.

$$Y(\theta) = X(\theta)H_1(\theta)H_2(\theta)$$

With $X(\theta) = 1$, the first system is

$$H_1(\theta) = \frac{1}{1-.5e^{-j\theta}}$$

and the second system is

$$H_2(\theta) = \frac{1}{1-\frac{1}{4}e^{-j\theta}}$$

The output then is given as

$$Y(\theta) = (1)\left(\frac{1}{1-.5e^{-j\theta}}\right)\left(\frac{1}{1-\frac{1}{4}e^{-j\theta}}\right) = \frac{2}{1-\frac{1}{2}e^{-j\theta}} - \frac{1}{1-\frac{1}{4}e^{-j\theta}}.$$

Using Table 3.1 we get

$$y(n) = 2\left(\frac{1}{2}\right)^n - \left(\frac{1}{4}\right)^n \quad n \geq 0$$

with

$$H(\theta) = \frac{1}{1 - \frac{3}{4}e^{-j\theta} + \frac{1}{8}e^{-j2\theta}}$$

Let us plot $y(n)$ analytically and using MATLAB with the following script:

```
n =0:20;
x = zeros(1,length(n));x(1)=1; % this is the impulse input
b = [1];
a = [ 1 -3/4 1/8];
yanalyt=2*(1/2).^n-(1/4).^n;
ymatlab = filter(b, a, x);
subplot (3, 1, 1) ;stem(n,x);  ylabel('Input signal')
subplot (3, 1, 2); stem(n, yanalyt); ylabel('Output
 analytically')
subplot (3, 1, 3) ;stem(n, ymatlab); ylabel('Output using
 MATLAB')
xlabel('n');
```

The plots are shown in Figure 3.20.

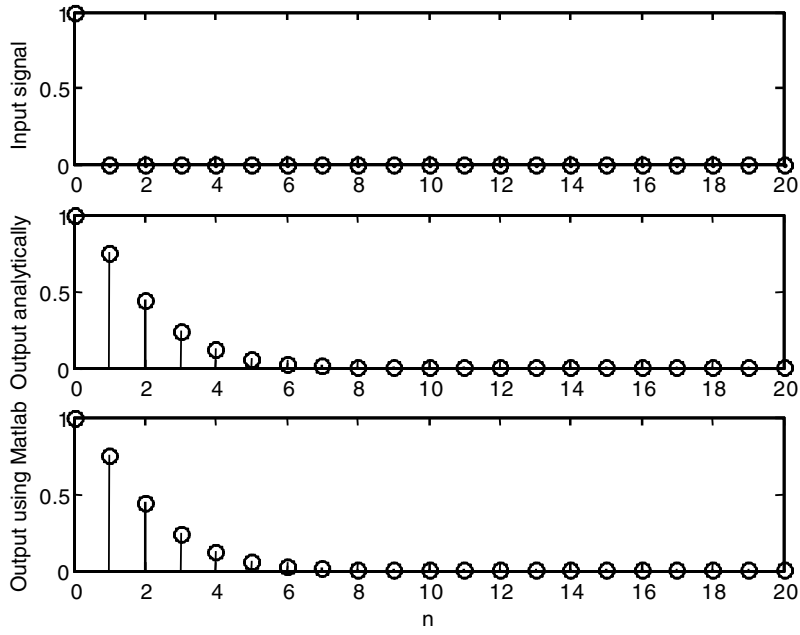

FIGURE 3.20 Plots for EOCE 3.10.

3.13 End of Chapter Problems

EOCP 3.1

Perform the following calculations and give the result in the form $a + jb$.

1. $\dfrac{1-j}{j}-1$
2. $\dfrac{1-je^{-j\frac{\pi}{2}}}{1+j}$
3. $\dfrac{1}{j}-\dfrac{1}{j-1}+j$
4. $\dfrac{e^{\pi}-e^{-j\pi}}{j}+1$
5. $e^{j\frac{\pi}{2}}+\dfrac{e^{j\pi}-1}{1-j}$

EOCP 3.2

Put the complex numbers in EOCP 3.1 in the polar form.

EOCP 3.3

Find the frequency response for each of the following systems.

1. $h(n)=\delta(n)+\delta(n-1)$
2. $h(n)=\delta(-n+2)+\delta(-n+1)+\delta(n)+\delta(n-1)+\delta(n-2)$
3. $h(n)=a(b)^n u(n)$
4. $h(n)=a(b)^n u(n)-a(b)^{n-1}u(n-1)$
5. $h(n)=u(n-4)$
6. $h(n)=a(b)^n u(n)-a(b)^{n-4}u(n-4)$
7. $h(n)=(a)^n\cos(n\pi)u(n)$
8. $h(n)=(a)^n\cos(n\pi)u(n)-(a)^{n-1}\cos(n\pi-\pi)u(n-1)$
9. $h(n)=e^{j\frac{\pi}{2}}(a)^n\sin(n\pi)u(n)$
10. $h(n)=j\sin\left(n\dfrac{\pi}{2}\right)u(n)-j\cos\left((n-1)\dfrac{\pi}{2}\right)u(n-1)$

EOCP 3.4

For a = .5, and b = .4, use MATLAB to plot the magnitude and phase of the frequency responses found in EOCP 3.3.

EOCP 3.5

Use MATLAB to find the steady-state response of each system in EOCP 3.3 to the inputs

1. $x(n) = \cos\left(\frac{\pi}{2}n\right)u(n)$
2. $x(n) = \cos\left(\frac{\pi}{2}n\right)u(n) - \cos\left(\frac{\pi}{2}(n-1)\right)u(n-1)$

EOCP 3.6

Find the frequency response for each of the following systems.

1. $y(n) + ay(n-1) = x(n)$
2. $y(n) + ay(n-1) = x(n) + x(n-1)$
3. $y(n) + ay(n-1) + by(n-2) = x(n)$
4. $y(n) + ay(n-2) = x(n)$
5. $y(n) + ay(n-3) = x(n) + x(n-3)$
6. $y(n) - ay(n-2) = x(n-3)$
7. $y(n) - ay(n-3) - by(n-4) = x(n)$
8. $y(n) - ay(n-6) = x(n-3)$
9. $y(n) + ay(n-1) = x(n) + x(n-1) + x(n-3)$
10. $y(n) = x(n) + x(n-1) + x(n-3) + x(n-4)$

EOCP 3.7

Use MATLAB to plot the magnitude and phase of the systems in EOCP 3.6 for $a = .1$ and $b = .5$.

EOCP 3.8

Use MATLAB to find the steady-state responses for the system in EOCP 3.6 with a and b as given in EOCE 3.7 if

1. $x(n) = 10$
2. $x(n) = \sin\left(\frac{2\pi}{3}n\right)u(n)$

EOCP 3.9

Find the Fourier transform of the following signals.

1. $3\delta(n) + 3\delta(n-1)$
2. $(.5)^n u(n) + e^{jn}$
3. $(.5)^n u(n-1) + e^{j(n-1)}$

4. $(.5)^n u(n) - (.5)^{n-1}u(n-1)$
5. $(n-1)(.5)^{n-1}\, u(n-1)$
6. $3\,\delta(n) * 3\delta(n-1)$
7. $(.5)^{n-1}u(n-1) * e^{j(n-1)}$
8. $(.5)^n u(n) * 3\delta(n) * 3\delta(n-1)$
9. $(.5)^{n-1}u(n-1) * x(n) * \delta(n)$
10. $(.5)^n u(n) + 3\delta(n) * 3\delta(n-1)$

EOCP 3.10

Use the Fourier transform to find the output $y(n)$ when the input is $x(n) = (.5)^n u(n)$ for the following systems.

1. $h(n) = \delta(n) + \delta(n-1)$
2. $h(n) = \delta(n-1) + \delta(n-2)$
3. $h(n) = (.5)^n u(n)$
4. $h(n) = (.5)^n u(n) - (.5)^{n-1}u(n-1)$
5. $h(n) = (.3)^{n-5}u(n-5) + \delta(n)$

EOCP 3.11

Consider the following systems shown in Figures 3.21 through 3.25. Find $y(n)$ if $x(n) = (.5)^n u(n)$. Use the Fourier transform method. We are given that

$$h_1(n) = \delta(n) + \delta(n-1)$$

$$h_2(n) = (.5)^n u(n)$$

$$h_3(n) = (.3)^{n-5}u(n-5) + \delta(n) = h(n)$$

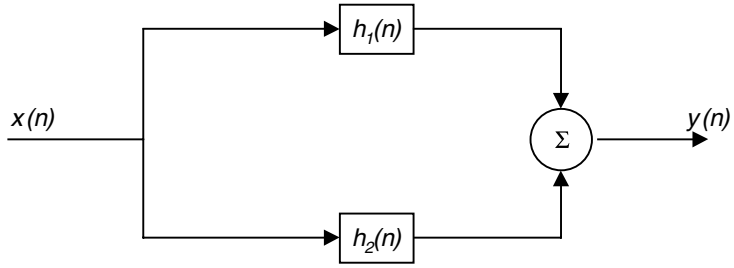

FIGURE 3.21 Signal for EOCP 3.11.

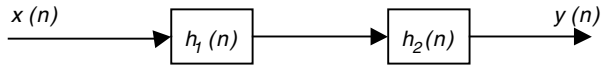

FIGURE 3.22 Signal for EOCP 3.11.

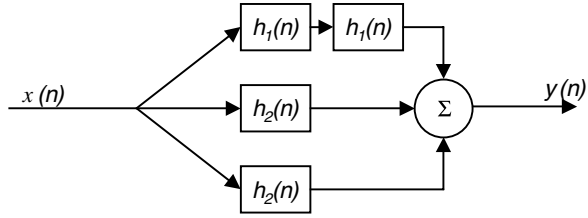

FIGURE 3.23 Signal for EOCP 3.11.

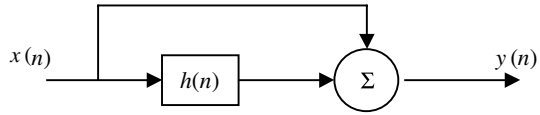

FIGURE 3.24 Signal for EOCP 3.11.

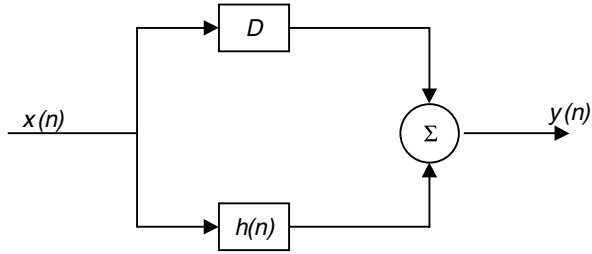

FIGURE 3.25 Signal for EOCP 3.11.

EOCP 3.12

Consider the signal in Figure 3.26. Find the Fourier transform of this signal.

EOCP 3.13

Consider the system

$$y(n) = x(n) - x(n-1)$$

1. Find the steady-state output if $x(n) = A\cos(n\theta)$.
2. Find the steady-state response when

$$x(n) = x_1(n) + x_2(n)$$

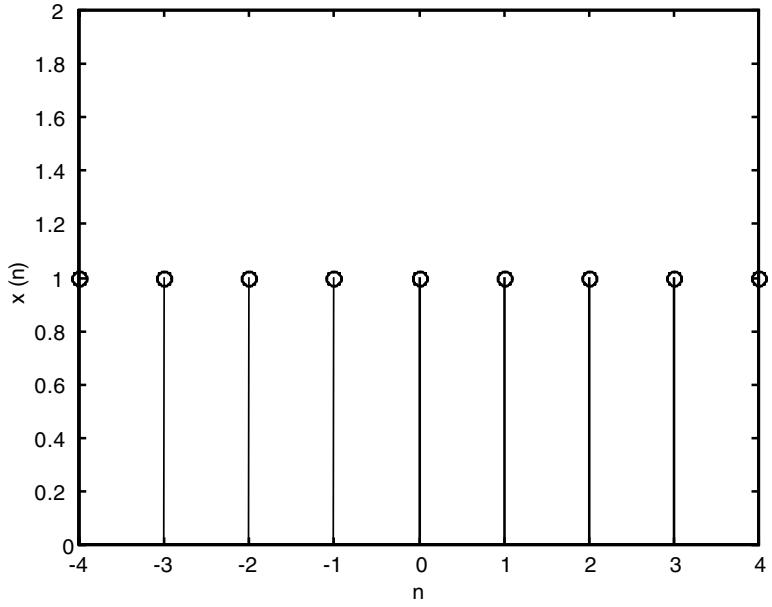

FIGURE 3.26 Signal for EOCP 3.12.

where

$$x_1(n) = B\cos(n\theta) \text{ and } x_2(n) = C\cos(100n\theta)$$

3. What do you think that this system is doing?

EOCP 3.14

Consider the system

$$y(n) - y(n-1) = \frac{x(n) + x(n-1)}{2}$$

1. Find the frequency response of this system.
2. Find the steady-state response when

$$x(n) = \left(A\cos\left(\frac{n\pi}{3}\right)u(n)\right)$$

3. What do you say about this system?

EOCP 3.15

Consider the systems

$$H\left(e^{j\theta}\right) = 10 + 2e^{-j\theta} + 3e^{-3j\theta}$$

and

$$H\left(e^{j\theta}\right) = \frac{1}{1 + .5e^{-j\theta} - 5e^{-j2\theta}}$$

1. Find the difference equation that describes the two systems.
2. What is $h(n)$, the impulse response for both?
3. Use MATLAB to find the steady-state response if

$$x(n) = 10\cos\left(\frac{n\pi}{3}\right)u(n)$$

4. Are both systems stable?
5. Use MATLAB to plot the magnitude response of each system.
6. What kind of system is each?

EOCP 3.16

Consider the following discrete signals with T_s = 0.1 sec.

1. $x(n) = \sin\left(\frac{4}{3}\pi n\right)$
2. $x(n) = \sin\left(\frac{8}{3}\pi n\right)$
3. $x(n) = \{1\ \ 1\ \ 1\}$ a periodic signal with $N = 3$.

a) Find the period for the first two signals.
b) Find the Fourier series coefficients.
c) Where are these frequency components located?

4

The z-Transform and Discrete Systems

4.1 Introduction

The *z*-transform is a frequency domain representation that makes solution, design and analysis of discrete linear systems simpler. It also gives some insights about the frequency contents of signals where these insights are hard to see in real-time systems. There are other important uses for the *z*-transform but we will concentrate only on the issues described in this introduction.

4.2 The Bilateral *z*-Transform

The *z*-transform of the signal $x(n)$ is given by

$$X(z) = \sum_{n=-\infty}^{+\infty} x(n) z^{-n} \tag{4.1}$$

where z is the complex variable. If we try to expand Equation (4.1) we get

$$X(z) = \ldots + x(-2)z^{2} + x(-1)z^{1} + x(0)z^{0} + x(1)z^{-1} + x(2)z^{-2} + \ldots \tag{4.2}$$

You can see that in Equation (4.1) the power of z indicates the position of the samples in the signal $x(n)$. This notice is very important. Consider that

$$x(n) = x(0)\ \delta(n)$$

where this signal has the strength $x(0)$ and is located only at $n = 0$. The *z*-transform of $x(n)$, $X(z)$, is then

$$X(z) = \sum_{n=-\infty}^{+\infty} x(n)z^{-n} = \sum_{n=-\infty}^{+\infty} x(0)\delta(n)z^{-n} = x(0)\ z^0 = x(0)$$

Similarly, if

$$x(n) = x(-1)\ \delta(n+1) + x(0)\ \delta(n) + x(1)\ \delta(n-1)$$

then we can see that this signal has values only at $n = -1$, $n = 0$ and $n = 1$. Thus

$$X(z) = x(-1)\ z^1 + x(0)\ z^0 + x(1)\ z^{-1} = x(-1)\ z^1 + x(0) + x(1)\ z^{-1}$$

In general, if

$$x(n) = x(p)\ \delta\left(n - n_0\right)$$

is a signal that is available only at $n = n_0$, then $X(z)$ is

$$X(z) = x(p)(z)^{-n_0}$$

Example 4.1

Consider the signal in Figure 4.1. What is the z-transform of $x(n)$?

Solution

$x(n)$ can be written as

$$x(n) = 2\delta(n+2) - 1\delta(n+1) + 2\delta(n) - 1\delta(n-1) + 2\delta(n-2)$$

and its z-transform is given by Equation (4.1) as

$$X(z) = \sum_{n=-\infty}^{+\infty} \left(x(n)\right)z^{-n}$$

Substituting in the above equation we get

$$X(z) = 2z^2 - 1z^1 + 2z^0 - z^{-1} + 2z^{-2} = 2z^2 - z^1 + 2 - z^{-1} + 2z^{-2}$$

Notice that z^{-2} represents a delay of $2T_s$ units of time and z^3 represents an advance of $3T_s$ units of time, where T_s is the sampling interval for the signal $x(n)$.

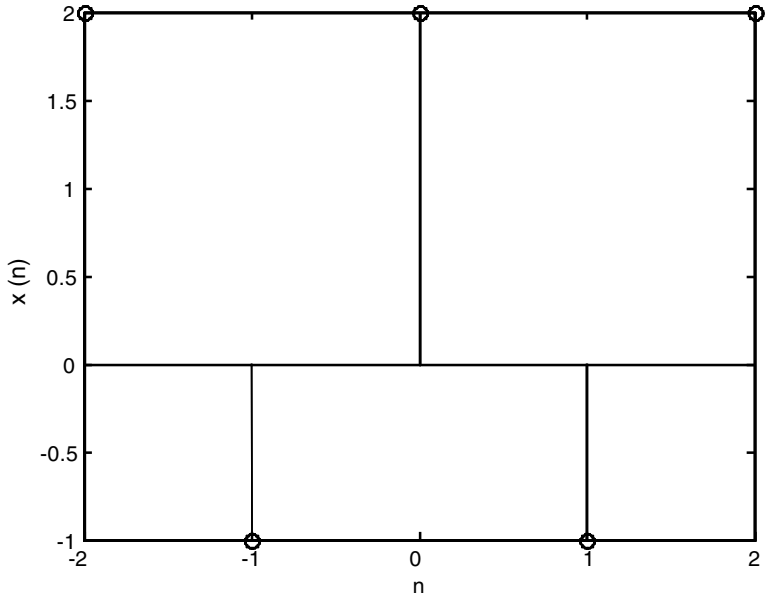

FIGURE 4.1 Signal for Example 4.1.

4.3 The Unilateral *z*-Transform

The unilateral z-transform is the transform of the signal $x(n)$ for $n \geq n_0$. We will take $n_0 = 0$ in this discussion. We will use the notation

$$x(n) \leftrightarrow X(z)$$

to indicate that we can get $X(z)$ from $x(n)$ and we can get $x(n)$ from $X(z)$ as well.

Example 4.2

Find the z-transform of $x(n) = A\delta(n)$

Solution

Using the definition of the z-transform we write

$$X(z) = \sum_{n=0}^{+\infty} x(n)z^{-n} = \sum_{n=0}^{+\infty} A\delta(n)z^{-n}$$

But $\delta(n)$ is defined only at $n = 0$. So $X(z) = Az^{-0} = A$ and we write

$$A\delta(n) \leftrightarrow A$$

Similarly we have

$$A\delta(n - n_0) \leftrightarrow Az^{-n_0}$$

Example 4.3

Find the z-transform of $x(n) = Au(n)$, the unit step discrete signal.

Solution

With Equation (4.1) we have

$$X(z) = \sum_{n=-\infty}^{+\infty} x(n)z^{-n}$$

Since $u(n)$ starts at $n = 0$ and is available only for $n \geq 0$, $X(z)$ becomes

$$X(z) = \sum_{n=0}^{\infty} Az^{-n} = A\sum_{n=0}^{\infty} \left(z^{-1}\right)^n = \frac{A}{1 - z^{-1}}$$

where the last result is a direct application of the geometric series sum. Thus we write

$$Au(n) \leftrightarrow \frac{A}{1 - z^{-1}}$$

and similarly

$$Au(n - n_0) \leftrightarrow \frac{Az^{-n_0}}{1 - z^{-1}}$$

Example 4.4

Find the z-transform of the signal $x(n) = Aa^n$ for $n \geq 0$.

Solution

Using the defining equation of the z-transform, we write using the unilateral case

$$X(z)=\sum_{n=0}^{+\infty}x(n)z^{-n}=\sum_{n=0}^{+\infty}Aa^{n}z^{-n}=A\sum^{\infty}\left(az^{-1}\right)^{n}=\frac{A}{1-az^{-1}}$$

Therefore, we write

$$Aa^{n}u(n)\leftrightarrow\frac{A}{1-az^{-1}}$$

and

$$Aa^{n-p}u(n-p)\leftrightarrow\frac{Az^{-p}}{1-az^{-1}}$$

Example 4.5

Find the z-transform of the complex exponential discrete signal

$$x(n)=Aa^{n}e^{j\theta n}u(n)$$

Solution

Using the defining equation again we write

$$X(z)=\sum_{n=0}^{+\infty}Aa^{n}e^{j\theta n}u(n)z^{-n}=A\sum_{n=0}^{+\infty}\left(ae^{j\theta}z^{-1}\right)^{n}=\frac{A}{1-ae^{j\theta}z^{-1}}$$

Using the notation we established we write

$$Aa^{n}e^{j\theta n}u(n)\leftrightarrow\frac{A}{1-ae^{j\theta}z^{-1}}$$

Example 4.6

Find the z-transform of the signal

$$x(n)=Aa^{n}\cos(\theta n)u(n)$$

Solution

Using the defining equation we get

$$X(z)=\sum_{n=0}^{+\infty}x(n)z^{-n}=\sum_{n=0}^{+\infty}Aa^{n}\cos(\theta n)u(n)z^{-n}=\frac{A}{2}\sum_{n=0}^{\infty}a^{n}\left(e^{j\theta n}+e^{-j\theta n}\right)z^{-n}$$

By rearranging terms and using the geometric series sum we arrive at

$$X(z)=\frac{A}{2}\sum_{n=0}^{\infty}\left(ae^{j\theta}z^{-1}\right)^n+\frac{A}{2}\sum_{n=0}^{\infty}\left(ae^{-j\theta}z^{-1}\right)^n=\frac{A}{2}\left[\frac{1}{1-ae^{j\theta}z^{-1}}+\frac{1}{1-ae^{-j\theta}z^{-1}}\right]$$

After simplification we get

$$Aa^n\cos(\theta n)u(n)\leftrightarrow\frac{A\left(1-az^{-1}\cos\theta\right)}{1-az^{-1}\left(e^{j\theta}+e^{-j\theta}\right)+a^2z^{-2}}$$

4.4 Convergence Considerations

The z-transform of $x(n)$ is given by

$$X(z)=\sum_{n=0}^{+\infty}x(n)z^{-n}$$

where z is a complex number. This complex number can be written in polar form as

$$z=re^{j\theta}$$

With z in polar form, $X(z)$ becomes

$$X\left(re^{j\theta}\right)=\sum_{n=0}^{+\infty}x(n)\left(re^{j\theta}\right)^{-n}=\sum_{n=0}^{+\infty}x(n)r^{-n}e^{-j\theta n}\tag{4.3}$$

So we can see that the z-transform of $x(n)$ is the Fourier transform of $x(n)r^{-n}$. If $r=1$ then $z=e^{j\theta}$ and $|z|=\left|e^{j\theta}\right|=1$, which is a circle of unity magnitude radius.

In Equation (4.3) and if we consider the unilateral case, the series must converge for the z-transform to exist. This will happen if $x(n)r^{-n}$ is absolutely summable. Mathematically we require

$$\sum_{n=0}^{+\infty}\left|x(n)r^{-n}\right|<\infty$$

The region where the z-transform converges is called the region of convergence (ROC) and is usually an annular region.

Example 4.7

What is the ROC of the z-transform of $x(n) = Au(n)$?

Solution

We have seen that

$$x(n) = Au(n) \leftrightarrow \frac{A}{1 - z^{-1}} = \frac{A}{z - 1}$$

The ROC is $|z| > 1$, which is the region exterior to the unit circle in the z-plane.

Example 4.8

What is the ROC of the z-transform of $x(n) = A\, a^n\, u(n)$?

Solution

We have seen that

$$Aa^n u(n) \leftrightarrow \frac{A}{1 - az^{-1}} = \frac{Az}{z - a}$$

The ROC is then $|z| > |a|$.

Example 4.9

What is the ROC of the z-transform of $x(n) = Aa^n e^{j\theta}\, u(n)$?

Solution

We have seen that

$$Aa^n e^{j\theta n} u(n) \leftrightarrow \frac{A}{1 - ae^{j\theta} z^{-1}} = \frac{Az}{z - ae^{j\theta}}$$

for which the ROC is $|z| > |ae^{j\theta}| = |a|$.

Example 4.10

What is the ROC of the z-transform of $x(n) = Aa^n \cos(\theta n)\, u(n)$?

Solution

We have seen that

$$Aa^n \cos(\theta n)a(n) \leftrightarrow \frac{A}{2}\left[\frac{1}{1-ae^{j\theta}z^{-1}} + \frac{1}{1-ae^{-j\theta}z^{-1}}\right] = \frac{A}{2}\left[\frac{z}{z-ae^{j\theta}} + \frac{z}{z-ae^{-j\theta}}\right]$$

The ROC for both terms is $|z| > |a|$ since $|e^{-j\theta}| = |e^{j\theta}| = 1$.

Example 4.11

Find the ROC of the z-transform of $x(n) = (.5)^n\, u(n) + (.4)^n\, u(n)$.

Solution

From the properties of the z-transform (which will be discussed later), we have

$$X(z) = X_1(z) + X_2(z)$$

where $X_1(z)$ and $X_2(z)$ are the z-transforms for $(.5)^n\, u(n)$ and $(.4)^n\, u\ (n)$, respectively.

$$X(z) = \frac{1}{1-.5z^{-1}} + \frac{1}{1-.4z^{-1}} = \frac{z}{z-.5} + \frac{z}{z-.1} = \frac{z(z-.4)+z(z-.5)}{(z-.5)(z-.4)}$$

The ROC is given by $|z| > .5$ and $|z| > .4$. Thus we conclude that the ROC is $|z| > .4$. This is the annular region outside the circle of radius .4 in magnitude.

Example 4.12

Find the ROC of the z-transform of $x(n) = (.5)^n\, u(n) + (.9)^n\, u(-n-1)$.

Solution

The first signal, $x(n) = (.5)^n\, u(n)$, has the transform

$$X_1(z) = \sum_{n=0}^{\infty} (.5)^n z^{-n} = \frac{1}{1-.5z^{-1}} = \frac{z}{z-.5}$$

with the ROC $|z| > .5$. The second signal $x_2(n) = (.9)^n\, u(-n-1)$ has the z-transform

$$X_2(z) = \sum_{n=-\infty}^{-1} (.9)^n z^{-n}$$

Let as add 1 and subtract 1 from the right side of the equation above so that we make the summation stop at $n = 0$. We will get

$$X_2(z) = \sum_{n=-\infty}^{n=0} (.9)^n z^{-n} - 1$$

To start the summation from $n = 0$ to $n = \infty$, we replace n by $-n$ under the summation to get

$$X_2(z) = \sum_{n=0}^{\infty} (.9)^{-n} z^n - 1 = \sum_{n=0}^{\infty} \left(.9^{-1} z^1\right)^n - 1 = \frac{1}{1 - .9^{-1} z} - 1 = -\frac{1}{1 - .9z^{-1}}$$

For the second signal we require that $|(.9)^{-1} z^1| < 1$. This implies an ROC given by $|z| < .9$. Thus to find the ROC of $X(z)$ we require that $|z| > .5$ and $|z| < .9$. This means that the ROC is $.5 < |z| < .9$.

4.5 The Inverse *z*-Transform

The inverse z-transform can be obtained analytically as

$$x(n) = \frac{1}{2\pi j} \oint_C X(z) z^{n-1} dz \tag{4.4}$$

with $j = \sqrt{-1}$ and c is a counter-clockwise closed path in the z-plane.

To avoid integration in the z-plane to find $x(n)$ from $X(z)$, we can use other ways to find $x(n)$ given $X(z)$ with the help of Tables 4.1 and 4.2.

4.5.1 Partial Fraction Expansion

We will assume that our signals $x(n)$ are defined for $n \geq 0$. The best way to illustrate the method is to give an example.

Example 4.13

Consider the signal in the z-domain

$$X(z) = \frac{z}{(z-1)(z-2)}$$

What is $x(n)$?

TABLE 4.1

Selected z-Transform Pairs

$x(n)$	$X(z)$	ROC
$A\delta(n)$	A	Entire z-plane
$Au(n)$	$\frac{Az}{z-1}$	$\lvert z\rvert > 1$
$nu(n)$	$\frac{z}{(z-1)^2}$	$\lvert z\rvert > 1$
$a^n u(n)$	$\frac{z}{z-a}$	$\lvert z\rvert > \lvert a\rvert$
$na^n u(n)$	$\frac{az}{(z-a)^2}$	$\lvert z\rvert > \lvert a\rvert$
$na^n u(n)$	$\frac{az}{(z-a)^2}$	$\lvert z\rvert > \lvert a\rvert$
$n^2 u(n)$	$\frac{z(z+1)}{(z-1)^3}$	$\lvert z\rvert > 1$
$n^2 a^n u(n)$	$\frac{az(z+a)}{(z-a)^3}$	$\lvert z\rvert > \lvert a\rvert$
$A\ sin(\theta n)u(n)$	$\frac{Az\sin\theta}{z^2-2z\cos\theta+1}$	$\lvert z\rvert > \lvert a\rvert$
$A\ cos(\theta n)u(n)$	$\frac{z(z-\cos\theta)}{z^2-2z\cos\theta+1}$	$\lvert z\rvert > 1$
$a^n\ sin(\theta n)u(n)$	$\frac{az\sin\theta}{z^2-2az\cos\theta+a^2}$	$\lvert z\rvert > \lvert a\rvert$

TABLE 4.2

z-Transform Properties

Discrete Time–Domain	z-Domain
$a_1x_1(n) + a_2x_2(n)$	$a_1X_1(z) + a_2X_2(z)$
$x(n - n_0)u(n - n_0)$	$z^{-n_0}\ X(z)\ \ n_0 \geq 0$
$a^n x(n)$	$X(z/a)$
$n\ x(n)$	$-z\frac{dX(z)}{dz}$
$x(n/p)$	$X(z^p)$ for positive p
$x_1(n)^*\ x_2(n)$	$X_1(z)\ X_2(z)$
$x(0)$	$\lim_{z\to\infty} X(z)$
$x(\infty)$	$\lim_{z\to 1}(z-1)X(z)$ for known $x(\infty)$

Solution

$X(z)$ can be written as

$$X(z) = \frac{A}{z-1} + \frac{B}{z-2}$$

where A and B are constants. After determining A and B, we will use the entries in Table 4.1 and the properties in Table 4.2 to find $x(n)$. However, the entry

$$a^n u(n) \leftrightarrow \frac{z}{z-a}$$

requires a z in the numerator. Therefore we need to divide $X(z)$ by z then do the partial fraction expansion. After we are done, we will again multiply the results by z to get $X(z)$. So we will write

$$\frac{X(z)}{z} = \frac{z}{z(z-1)(z-a)} = \frac{1}{(z-1)(z-2)} = \frac{A}{z-1} + \frac{B}{z-2}$$

The constants are determined as

$$A = \left.\frac{1}{z-2}\right|_{z=1} = -1$$

$$B = \left.\frac{1}{z-1}\right|_{z=2} = 1$$

Therefore,

$$\frac{X(z)}{z} = \frac{-1}{z-1} + \frac{1}{z-2}$$

The z-transform is then

$$X(z) = \frac{-z}{z-1} + \frac{z}{z-2}$$

Now we can use Table 4.1 to get $x(n)$ with the help of entry 1 in Table 4.2. We will get

$$x(n) = -(1)^n u(n) + (2)^n u(n) = -u(n) + (2)^n u(n)$$

with

$x(0) = -1 + 1 = 0$
$x(1) = -1 + 2 = 1$
$x(2) = -1 + 4 = 3$
$x(3) = -1 + 8 = 7$

4.5.2 Long Division

In most cases $X(z)$ can be put in a rational fraction form as a ratio of two polynomials in z, the numerator and the denominator. Then we can use long division to find the first few values of $x(n)$. This method is good to check the results of the closed form for $x(n)$. The two polynomials should be put in descending power of z.

Example 4.14

Find the first three values of $x(n)$ for

$$X(z) = \frac{z}{(z-1)(z-2)}$$

Solution

We can multiply out the denominator to get

$$X(z) = \frac{z}{z^2 - 3z + 2}$$

Now we arrange X(z) as in the following

$$\begin{array}{r} z^{-1} + 3z^{-2} + 7z^{-3} + \cdots \\ z^2 - 3z + 2 \overline{)\, z} \\ + \\ -z + 3 - 2z^{-1} = 3 - 2z^{-1} \\ + \\ -3 + 9z^{-1} - 6z^{-2} = 7z^{-1} - 6z^{-2} \\ + \\ -7z^{-1} + 21z^{-2} - 14z^{-3} = 15z^{-2} - 14z^{-3} \\ \vdots \end{array}$$

The result is read as

$$X(z) = \frac{z}{z^2 - 3z + 7} = 0z^0 + (1)z^{-1} + 3z^{-2} + 7z^{-3} + \cdots$$

This indicates that

$$x(0) = 0,\ x(1) = 1,\ x(2) = 3 \text{ and } x(3) = 7$$

as was obtained before with the closed form.

4.6 Properties of the *z*-Transform

Next we will discuss some of the important properties of the *z*-transform. We assume that the signals start at $n \geq 0$.

4.6.1 Linearity Property

The *z*-transform of $x(n) = a_1 x_1(n) + a_2 x_2(n)$ is

$$\begin{aligned} X(z) &= \sum_{n=0}^{\infty} \big(a_1 x_1(n) + a_2 x_2(n)\big) z^{-n} \\ &= a_1 \sum_{n=0}^{\infty} x_1(n) z^{-n} + a_2 \sum_{n=0}^{\infty} x_2(n) z^{-n} \\ &= a_1 X_1(z) + a_2 X_2(z) \end{aligned}$$

Therefore, we write

$$x(n) = a_1\, x_1(n) + a_2\, x_2(n) \leftrightarrow a_1\, X_1(z) + a_2\, X_2(z)$$

4.6.2 Shifting Property

The *z*-transform of $x(n\text{-}n_0)\ u(n\text{-}n_o)$ is

$$\sum_{n=0}^{\infty} x\big(n - n_0\big) u\big(n - n_0\big) z^{-n}$$

Since $u(n - n_0) = 1$ for $n \geq n_0$. we have the z-transform of $x(n - n_0)\ u(n - n_o)$ written as

$$\sum_{n=n_0}^{\infty} x\left(n-n_0\right)z^{-n}$$

Let $m = n - n_0$, then

$$\sum_{n=n_0}^{\infty} x\left(n-n_0\right)z^{-n} = \sum_{m=0}^{\infty} x(m)z^{-\left(m+n_0\right)} = z^{-n_0}\sum_{m=0}^{\infty} x(m)z^{-m} = z^{-n_0}X(z)$$

So we write

$$x\left(n-n_0\right)u\left(n-n_0\right) \leftrightarrow z^{-n_0}X(z)$$

The z-transform of $x(n - n_0)$ is

$$\sum_{n=0}^{\infty} x\left(n-n_0\right)z^{-n}$$

Let $n - n_0 = m$. Then the transform of $x(n - n_0)$ is

$$\sum_{m=-n_0}^{\infty} x(m)z^{-(m+n_0)} = \sum_{m=-n_0}^{\infty} x(m)z^{-m}z^{-n_0} = \left[\sum_{m=-n_0}^{-1} x(m)z^{-m}z^{-n_0} + \sum_{m=0}^{\infty} x(m)z^{-m}z^{-n_0}\right]$$

$$\sum_{m=-n_0}^{\infty} x(m)z^{-(m+n_0)} = \left[\sum_{m=-n_0}^{-1} x(m)z^{-m}z^{-n_0} + X(z)z^{-n_0}\right]$$

Therefore, we write

$$x\left(n-n_0\right) \leftrightarrow \left[\sum_{m=-n_0}^{-1} x(m)z^{-m}z^{-n_0} + X(z)z^{-n_0}\right]$$

The reason for this derivation is to take into account the initial conditions for $y(n)$ since they are always given for $n < 0$.

The z-transform of $x(n + n_0)$ is

$$\sum_{n=0}^{\infty} x\left(n+n_0\right)z^{-n}$$

Let $n + n_0 = m$. Then the transform of $x(n + n_0)$ is

$$\sum_{m=n_0}^{\infty} x(m)z^{-(m-n_0)} = \sum_{m=n_0}^{\infty} x(m)z^{-m}z^{n_0} = \left[\sum_{m=0}^{\infty} x(m)z^{-m}z^{n_0} - \sum_{m=0}^{m=n_0-1} x(m)z^{-m}z^{n_0}\right]$$

$$= z^{n_0}X(z) - \sum_{m=0}^{m=n_0-1} x(m)z^{-m}z^{n_0}$$

Therefore, we write

$$x\left(n+n_0\right) \leftrightarrow z^{n_0}X(z) - \sum_{m=0}^{m=n_0-1} x(m)z^{-m}z^{n_0}$$

The reason for this derivation will be apparent when we talk about state-space systems in later chapters. This also takes into consideration nonzero initial conditions.

4.6.3 Multiplication by e^{-an}

The z-transform of $e^{-an}u(n)x(n)$ is

$$\sum_{n=0}^{\infty} e^{-an}x(n)z^{-n} = \sum_{n=0}^{\infty} x(n)\left(e^{+a}z^{+1}\right)^{-n}$$

Thus we have

$$e^{-an}x(n) \leftrightarrow X\left(e^{a}z\right)$$

4.6.4 Convolution

The z-transform of

$$x_1(n) * x_2(n) = \sum_{k=-\infty}^{+\infty} x_1(k)x_2(n-k)$$

if $x_1(n)$ and $x_2(n)$ start at $n \geq 0$ is obtained using the z-transform defining summation equation

$$\sum_{n=0}^{\infty}\left[\sum_{k=0}^{\infty} x_1(k)x_2(n-k)\right]z^{-n}$$

which is also given as

$$\sum_{k=0}^{\infty} x_1(k)\sum_{n=0}^{\infty} x_2(n-k)z^{-n}$$

Let $n - k = m$ in the inner summation. Then $m = -k$ will be the lower summation. But $x_2(m)$ is defined only for $m \geq 0$. Therefore,

$$\sum_{k=0}^{\infty} x_1(k)\sum_{m=0}^{\infty} x_2(m)z^{-m-k} = \sum_{k=0}^{\infty} x_1(k)z^{-k}\sum_{m=0}^{\infty} x_2(m)z^{-m} = X_1(z)X_2(z)$$

From this we write

$$x_1(n) * x_2(n) \leftrightarrow X_1(z)X_2(z)$$

This is the convolution equation in real-time related to the convolution in the frequency domain. We see that convolution, a sometimes difficult operation, in real-time is simply complex multiplication in the z-domain.

4.7 Representation of Transfer Functions as Block Diagrams

Consider the general third-order transfer function

$$\frac{Y(z)}{X(z)} = \frac{az^3 + bz^2 + cz + d}{z^3 + ez^2 + fz + g}$$

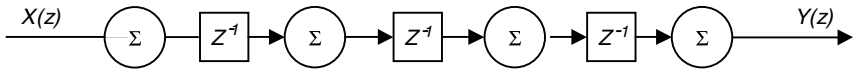

FIGURE 4.2 Block diagram step 1.

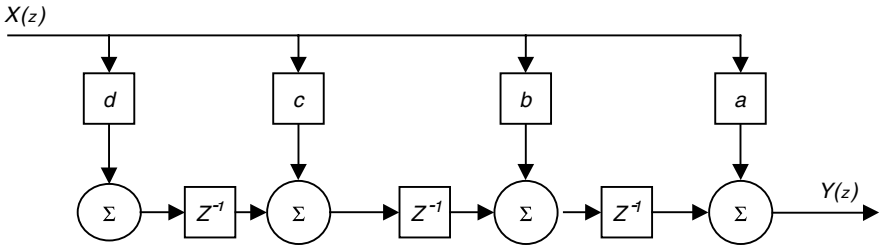

FIGURE 4.3 Block diagram step 2.

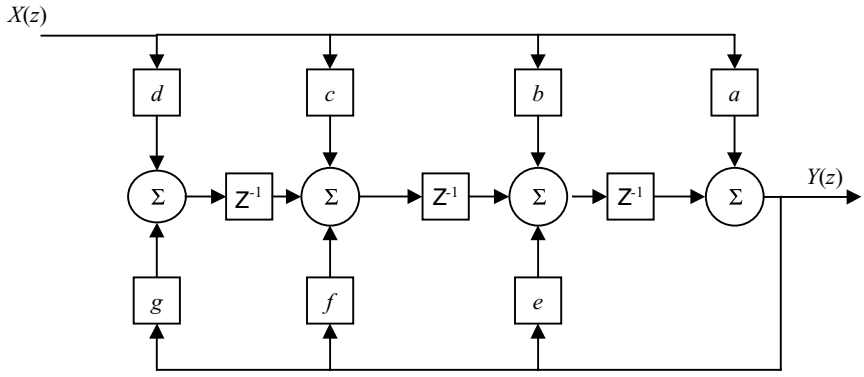

FIGURE 4.4 Block diagram step 3.

The block diagram representation for this transfer function is obtained using the following steps.

1. The system is third order and hence we need three delay elements. Each delay will be preceded by a summer and followed by a summer. The initial diagram is shown in Figure 4.2, where z^{-1} represents a delay element.
2. Next we feed forward, a to the summer at the output $Y(z)$, b to the summer before the third delay, c to the summer before the second delay and d to the summer before the first delay. The modified picture is shown in Figure 4.3.
3. Now we feed backward, g to the summer before the first delay, f to the summer before the second delay, and e to the summer before the third delay. The final block is shown in Figure 4.4.

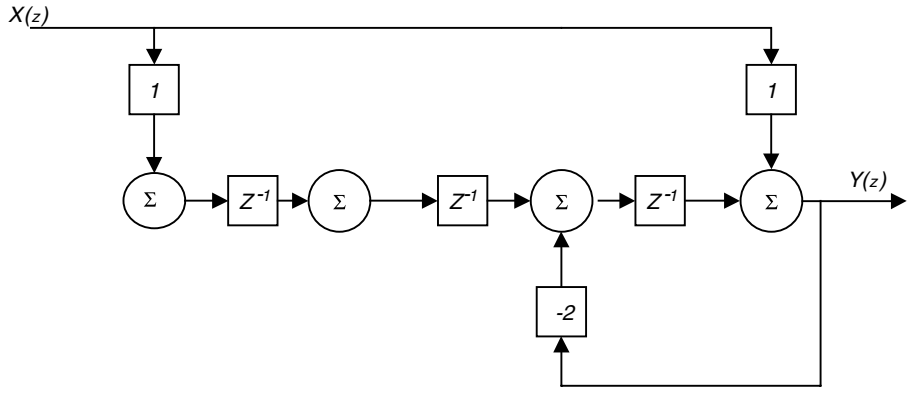

FIGURE 4.5 Block diagram for Example 4.15.

Example 4.15

Draw the block diagram for the system described by the following transfer function

$$\frac{Y(z)}{X(z)} = \frac{z^3 + 1}{z^3 + 2z^2}$$

Solution

First we write the system transfer function as

$$\frac{Y(z)}{X(z)} = \frac{z^3 + 0z^2 + 0z + 1}{z^3 + 2z^2 + 0z + 0}$$

We see that we need three delay elements. The block is shown in Figure 4.5. Note that we have only one feedback line to the summer that precedes the third delay. Also we have only two forward paths as seen in the figure.

4.8 $x(n)$, $h(n)$, $y(n)$, and the z-Transform

Throughout this book we have been using $x(n)$ to represent the input, $h(n)$ to represent the impulse response and $y(n)$ to represent the output. In discrete real-time the output is given using the convolution

$$y(n) = x(n) * h(n)$$

But as we saw earlier, convolution is multiplication is the transform domain. Thus we write $Y(z) = X(z)\,H(z)$. We then can use Table 4.1 and Table 4.2 to get back to the real-time signal $y(n)$.

Example 4.16

If $x(n) = u(n)$ is an input to the system with $h(n) = (.5)^n\,u(n)$, what is the output $y(n)$?

Solution

The output in the z-domain is

$$Y(z) = X(z)\,H(z)$$

The input in the z-domain is $X(z) = z/z - 1$ and the impulse response is $H(z) = z/z - .5$. Thus, using the convolution property of the z-transform we write

$$Y(z) = \frac{z}{z-1}\frac{z}{z-.5}$$

We will find $Y(z)/z$ first to make use of Table 4.1.

$$\frac{Y(z)}{z} = \frac{z}{(z-1)(z-5)} = \frac{A}{z-1} + \frac{B}{z-.5}$$

The constants A and B are calculated as

$$A = \left.\frac{z}{z-1/2}\right|z=1 = \frac{1}{1-1/2} = 2$$

$$B = \left.\frac{z}{z-1}\right|z=1/2 = \frac{1/2}{1/2-1} = -1$$

Therefore,

$$\frac{Y(z)}{z} = \frac{2}{z-1} - \frac{1}{z-.5}$$

and

$$Y(z) = \frac{2z}{z-1} - \frac{z}{z-.5}$$

To get back $y(n)$ we use Table 4.1 and write

$$y(n) = 2u(n) - .5^n\, u(n)$$

4.9 Solving Difference Equation Using the z-Transform

We have seen before that the z-transform of $x(n - n_0)$ is

$$x\left(n-n_0\right) \leftrightarrow \left[\sum_{m=-n_0}^{-1} x(m)z^{-m}z^{-n_0} + X(z)z^{-n_0}\right]$$

This relation is important when we z-transform difference equation.

Example 4.17

Consider the system

$$y\,(n) - .5y(n-1) = x(n)$$

with $y(-1) = 0$ and $x(n) = u(n)$. Find $y(n)$ for $n \geq 0$.

Solution

We will z-transform the given difference equation term by term.

$$y(n) \leftrightarrow Y(z)$$

$$y(n-1) \leftrightarrow \sum_{m=-1}^{-1} y(m)z^{-1}z^{-m} + z^{-1}Y(z) = y(-1)z^{-1}z^{+1} + z^{-1}Y(z)$$

Therefore, the given equation becomes

$$Y(z) - .5\left[y(-1)z^0 + z^{-1}Y(z)\right] = X(z) = \frac{z}{z-1}$$

or

$$Y(z) - .5z^{-1}Y(z) = \frac{z}{z-1}$$

Solving for $Y(z)$ we get

$$Y(z) = \frac{z^2}{(z-1)(z-.5)}$$

By doing partial fraction expansion on $Y(z)$ we get

$$Y(z) = \frac{2z}{z-1} - \frac{z}{z-.5}$$

and

$$y(n) = 2u(n) - (.5)^n u(n) \quad n \geq 0$$

Example 4.18

Use the z-transform to find the impulse response of the system

$$y(n) - y(n-2) = x(n)$$

Solution

We z-transform the equation above term by term to get

$$y(z) - \left[y(-2)z^0 + y(-1)z^{-1} + z^{-2}y(z)\right] = X(z)$$

With

$$X(z) = 1 \leftrightarrow \delta(n) = x(n)$$

and $y(-2) = y(-1) = 0$ (calculating the impulse response) we have

$$Y(z)\left(1 - z^{-2}\right) = X(z)$$

or

$$Y(z) = \frac{1}{1-z^{-2}} = \frac{z^2}{z^2-1}$$

and

$$\frac{Y(z)}{z} = \frac{z}{z^2-1} = \frac{A}{z-1} + \frac{B}{z+1}$$

The constants are

$$A = \frac{z}{z+1}\bigg|_{z=1} = \frac{1}{2}$$

$$B = \frac{z}{z-1}\bigg|_{z=-1} = \frac{1}{2}$$

Thus

$$Y(z) = \frac{1/2\,z}{z-1} + \frac{1/2\,z}{z+1}$$

and the output $y(n) = h(n)$ is

$$y(n) = 1/2\left[u(n) + (-1)^n u(n)\right]$$

4.10 Convergence Revisited

Consider the following transfer function in the z-domain

$$H(z) = \frac{z^2 + z + 1}{z^2 + 2z + 1}$$

Let us first find $h(n)$ using long division by first putting the numerator and the denominator of the transfer function in the descending powers of z. In this case we will get

$$H(z) = 1z^0 - z^{-1} + 2z^{-2} - 3z^{-3} + 4z^{-4} + \ldots$$

and the impulse response $h(n)$ in this case is

$$h(n) = \delta(n) - \delta(n-1) + 2\delta(n-2) - 3\delta(n-3) + 4\delta(n-4) + \ldots$$

This impulse response is causal since $h(n)$ has zero values for $n < 0$. However, if we put $H(z)$ in the form

$$H(z) = \frac{1 + z + z^2}{1 + 2z + z^2}$$

we will get

$$H(z) = 1 - z + 2z^2 - 3z^3 + 4z^4 + \ldots$$

and $h(n)$ in this case is

$$h(n) = \delta(n) - \delta(n+1) + 2\delta(n+2) - 3\delta(n+3) + 4\delta(n+4) + \ldots$$

The signal is noncausal because it is zero for $n > 0$ and nonzero or $n \leq 0$.

Notice that

$$H(z) = \frac{1 + z + z^2}{1 + 2z + z^2} = \frac{z^2 + z + 1}{z^2 + 2z + 1}$$

But we have two signals

$$h(n) = \delta(n) - \delta(n+1) + 2\delta(n+2) - 3\delta(n+3) + 4\delta(n+4) + \ldots$$

$$h(n) = \delta(n) - \delta(n-1) + 2\delta(n-2) - 3\delta(n-3) + 4\delta(n-4) + \ldots$$

There is a reason for what we see here. Consider the signal $h(n) = a^n\, u(n)$ with its z-transform

$$H(z) = \sum_{n=-\infty}^{+\infty} a^n u(n) z^{-n} = \sum_{n=0}^{\infty} a^n z^{-n} = \frac{1}{1 - az^{-1}} = \frac{z}{z - a}$$

This $H(z)$ has the ROC $|z| > |a|$ since we require that $|az^{-1}| < 1$ for the series to be summable. Notice also that $H(z)$ has its pole at $z = a$ and that $H(z)$ converges outside the circle of radius of magnitude a.

Consider next the signal $h(n) = a^n u(-n - 1)$ with its z-transform given by

$$H(z) = \sum_{-\infty}^{+\infty} a^n u(-n-1) z^{-n} = \sum_{n=-\infty}^{-1} a^n z^{-n} = \sum_{n=-\infty}^{n=0} a^n z^{-n} - 1 = \sum_{n=0}^{n=\infty} a^{-n} z^n - 1$$

where we changed the sign on n when we changed the limit to start at $n = 0$ and end at $n = \infty$. $H(z)$ then is written as

$$H(z) = \frac{1}{1 - a^{-1}z} - 1 = \frac{1 - \left(1 - a^{-1}z\right)}{1 - a^{-1}z} = \frac{a^{-1}z}{1 - a^{-1}z} = -\frac{z}{z - a}$$

But the ROC is now different since in this case we required that $|a^{-1}z| < 1$ for the series to converge. This means that $|z/a| < 1$ or $|z| < |a|$.

From what has been discussed it is clear that to find $h(n)$ from $H(z)$, the ROC must be given. To generalize, let us consider that $H(z)$ has N poles and M zeros. If $H(z)$ has a ROC for which

$$|z| > |p_i|$$

when p_i is the i^{th} pole that is farthest from the pole at the origin, then in this case the system is causal. But if the ROC is

$$|z| < |p_i|$$

where p_i is the i^{th} pole that is closest to the pole at the origin, then in this case the system is noncausal. Let us look at some examples.

Example 4.19

Consider the system

$$H(z) = \frac{(2z-3)z}{(z-1)(z-2)}$$

with ROC $|z| > 2$. Find $h(n)$.

Solution

The transfer function can be written as

$$H(z) = \frac{(2z-3)z}{(z-1)(z-2)} = \frac{z}{z-1} + \frac{z}{z-2}$$

We can see that the ROC of $H(z)$ is outside the rings $|z| = 1$ and $|z| = 2$ and hence $h(n) = (1)^n\, u(n) + (+2)^n\, u(n)$

Example 4.20

Consider the same $H(z)$ as in Example 4.19 but the ROC now is $|z| < 1$. What is $h(n)$?

Solution

We can see in this case that the ROC is inside both rings $|z| = 1$ and $|z| = 2$. Therefore

$$h(n) = -(1)^n\, u(-n-1) + -(2)^n\, u(-n-1)$$

The minus sign here is added because the system is noncausal.

4.11 The Final Value Theorem

The final value of the signal $x(n)$ as $n \to \infty$ can be obtained using the z-transform. The derivation is omitted since it is somewhat involved. We have

$$x(\infty) = \lim_{n\to\infty} x(n) = \lim_{z\to 1}(z-1)X(z) \tag{4.5}$$

if $x(n)$ has a final value. $x(n)$ will have a final value if all the poles of $X(z)$ are within the unit circle. This is to say that for all the poles z_i, $|z_i| > 1$.

Example 4.21

Consider the system

$$h(n) = (.5)^n u(n)$$

Find $h(\infty)$ or the final value of $h(n)$.

Solution

We can see here, since the expression for $h(n)$ is simple, that $h(\infty) = (.5)^\infty u(\infty) = 0$. This is clear because $(.5)^\infty$ will approach zero faster. But we can use the final value theorem since the pole for $H(z) = z/(z - .5)$ is within the unit circle. Therefore

$$h(\infty) = \lim_{z\to 1}(z-1)\left(\frac{z}{z-.5}\right) = (1-1)\frac{1}{1-.5} = 0$$

4.12 The Initial-Value Theorem

The initial-value theorem is used to find the initial value $x(0)$ for the signal $x(n)$. From the z-transform of $x(n)$ we write

$$X(z) = x(0) + x(1)z^{-1} + z(2)z^{-2} + \ldots$$

If we take the limit of $X(z)$ as z approaches infinity we will have

$$\lim_{z\to\infty} X(z) = x(0) + \lim_{z\to\infty}\frac{x(1)}{z} + \lim_{z\to\infty}\frac{x(2)}{z^2} + \ldots$$

So we have

$$\lim_{z\to\infty} X(z) = x(0) \tag{4.6}$$

as the initial-value theorem.

Example 4.22

If a certain system has the impulse response

$$h(n) = (.5)^n\, u(n)$$

what would be $h(0)$, the initial value for $h(n)$?

Solution

It is clear that $h(0) = (.5)^0\, u(0) = 1$. Using the initial value theorem, we have

$$h(0) = \lim_{z\to\infty} H(z) = \lim_{z\to\infty} \frac{z}{z-.5} = 1.$$

4.13 Some Insights: Poles and Zeroes

The transfer function, $H(z)$, of a linear time-invariant system is a very important representation. It tells us many things about the stability of the system, the poles, the zeros and the shape of the transients of the output of the system. Using $H(z)$ we can find the steady-state response of the system and the particular solution of the system all in one shot.

4.13.1 The Poles of the System

The poles of the system are the roots of the denominator, the algebraic equation in the variable z, of the transfer function $H(z)$

$$H(z) = \frac{N(z)}{D(z)}$$

$D(z)$ is a polynomial in z of order equal to the order of the system. The roots of the denominator $D(z)$ are called the poles of the system. These are the same poles we discussed in Chapter 2. We called them then the eigenvalues of the system. $D(z)$ is actually the characteristic equation of the system or, as referred to before, the auxiliary equation of the system.

4.13.2 The Zeros of the System

The roots of the numerator $N(z)$ are called the zeros of the system.

4.13.3 The Stability of the System

The poles of the system determine its stability. If the poles are *all* within the unit circle, then the system at hand is stable and the transients will die as time progresses. The stability of the system is determined by the poles and not the zeros. If one of the poles is outside the unit circle, then the system is not stable. You may have zeros that are outside the unit circle, but the location of the zeros has no effect on the stability of the system.

Given $H(z)$, the roots of the denominator will determine the general shape of the output $y(n)$ which, in this case, is $h(n)$ because the input $x(n)$ is the impulse $\delta(n)$. If $D(z)$ has two roots (second-order system) called α_1 and α_2 then the output will have the general form

$$y(n) = h(n) = c_1(\alpha_1)^n + c_2(\alpha_2)^n$$

where the constant c's are to be determined. The exponential terms will determine the shape of the transients. If one of the α's is outside the unit circle, the output will grow without bounds. If the two α's are within the unit circle, the output will die as time progresses. The α's are the eigenvalues or the poles of the system.

The transfer function $Y(z)/X(z)$ is called $H(z)$ if the input $X(z)$ is 1 ($x(n) = \delta(n)$). The transfer function $Y(z)/X(z)$ is very important as we will see later in the design of linear time-invariant systems

4.14 End of Chapter Exercises

EOCE 4.1

Find the z-transform of the signals

1. $x(n) = (1/3)^n\, u(n)$
2. $x(n) = -(1/2)^n\, u(-n-1)$
3. $x(n) = (1/2)^n\, u(n) - (1/2)^n\, u(-n-1)$

and indicate their ROC.

Solution

For the first signal $x(n) = (1/3)^n\, u(n)$ the z-transform is

$$X(z) = \sum_{n=-\infty}^{+\infty} \left(\frac{1}{3}\right)^n u(n) z^{-n} = \sum_{n=0}^{\infty} \left(\frac{1}{3} z^{-1}\right)^n = \frac{1}{1 - \frac{1}{3} z^{-1}} = \frac{z}{z - \frac{1}{3}} \quad \text{for } \left|\frac{1}{3} z^{-1}\right| < 1$$

or $|1/3| < |z|$. So the ROC in $|z| > 1/3$.

For the second signal, $x(n) = -(1/2)^n\, u(-n-1)$, the z-transform is given by

$$X(z) = \sum_{n=-\infty}^{+\infty} -\left(\frac{1}{2}\right)^n u(-n-1) z^{-n} = -\sum_{n=-\infty}^{-1} \left(\frac{1}{2}\right)^n z^{-n}$$

$$X(z) = -\sum_{n=1}^{n=\infty} \left(\frac{1}{2}\right)^{-n} z^n = 1 - \sum_{n=0}^{\infty} \left(\left(\frac{1}{2}\right)^{-1} z\right)^n = 1 - \frac{1}{1 - \left(\frac{1}{2}\right)^{-1} z} = \frac{z}{z - \frac{1}{2}}$$

with the ROC $|z| < 1/2$.

For the last signal, $x(n) = (1/3)^n\, u(n) - (1/2)^n\, u(-n-1)$, the z-transform is

$$X(z) = \sum_{n=-\infty}^{+\infty} \left(\frac{1}{3}\right)^n u(n) z^{-n} - \sum_{n=-\infty}^{+\infty} \left(\frac{1}{2}\right)^n u(-n-1) = \frac{z}{z - \frac{1}{3}} + \frac{z}{z - \frac{1}{2}}$$

with ROC now as $|z| > 1/3$ and $|z| < 1/2$. By combining these conditions we get

$$\frac{1}{3} < |z| < \frac{1}{2}$$

for the ROC.

EOCE 4.2

What is the z-transform of the signal

$$x(n) = (n+2)(.5)^n u(n)$$

Solution

$x(n)$ can be written as

$$x(n) = n x_1(n) + 2 x_1(n)$$

where $x_1(n) = (.5)^n\, u(n)$. Thus we have

$$X_1(z) = \frac{z}{z-.5} \text{ with } |z| > .5$$

Next we can use the differentiation property in Table 4.2 to find $X(z)$ as

$$X(z) = -z\frac{d}{dz}X_1(z) + 2\frac{z}{z-.5} = -z\left[\frac{(z-.5)-z}{(z-.5)^2}\right] + 2\frac{z}{z-.5}$$

$$X(z) = \frac{.5z}{(z-.5)^2} + 2\frac{z}{z-.5} = \frac{.5z + 2z(z-.5)}{(z-.5)^2} = \frac{2z^2 - .5z}{(z-.5)^2}$$

with $|z| > .5$ as the ROC.

EOCE 4.3

What is $X(z)$ if

$$x(n) = \cos(n)u(n) + nu(n)$$

Solution

We can divide the given signal into two parts

$$X(z) = X_1(z) + X_2(z)$$

where $X_1(z)$ and $X_2(z)$ are the z-transforms of $\cos(n)u(n)$ and $nu(n)$, respectively. Therefore

$$X(z) = \frac{1 - \cos(1)z^{-1}}{1 - 2\cos(1)z^{-1} + z^{-2}} + \frac{z^{-1}}{\left(1 - z^{-1}\right)^2}$$

with $|z| > 1$ as the ROC. $X_2(z)$ can be obtained by finding the z-transform of $u(n)$ first, then taking the derivative with respect to z as

$$X_2(z) = \frac{-zd}{dz}\left(\frac{z}{z-1}\right) = -z\left(\frac{z-1-z}{(z-1)^2}\right) = \frac{z}{(z-1)^2} = \frac{z^{-1}}{\left(1-z^{-1}\right)^2}$$

EOCE 4.4

What is the z-transform of the signal

$$x(n) = x_1(n) * x_2(n)$$

with

$$x_1(n) = \delta(n) + 2\delta(n-1)$$

and

$$x_2(n) = \delta(n-1) + 3\delta(n-2)$$

Solution

By using the convolution property we write

$$X(z) = X_1(z)X_2(z) = \left(1 + 2z^{-1}\right)\left(z^{-1} + 3z^{-2}\right) = z^{-1} + 3z^{-2} + 2z^{-2} + 6z^{-3}$$

and $x(n)$ then is

$$x(n) = \delta(n-1) + 5\delta(n-2) + 6\delta(n-3)$$

We can also use MATLAB and convolve $x_1(n)$ with $x_2(n)$ and write the following script

```
x1 = [1 2]
x2 = [0 1 3]
x = conv (x1, x2)
```

to get

$$x = \{\underset{\uparrow}{0}\ 1\ 5\ 6\}$$

which indicates that

$$X(z) = 0z^0 + 1z^{-1} + 5z^{-2} + 6z^{-3}$$

EOCE 4.5

With

$$x_1(n) = \delta(n+1) + \delta(n) + \delta(n-1)$$

and

$$x_2(n) = \delta(n) + \delta(n-1)$$

what is the z-transform if $x(n) = x_1(n) * x_2(n)$?

Solution

Convolution in real-time is multiplication in the z-domain. Thus

$$X(z) = X_1(z)X_2(z) = \left(z + 1 + z^{-1}\right)\left(1 + z^{-1}\right)$$

$$= z + 1 + 1 + z^{-1} + z^{-1} + z^{-2} = z + 2 + 2z^{-1} + z^{-2}$$

and the inverse z-transform is

$$x(n) = \delta(n+1) + 2\delta(n) + 2\delta(n-1) + \delta(n-2).$$

We can also use MATLAB to do this but with some attention to the starting index of each signal. We use the MATLAB function `conv` if the two signals start at $n = 0$. In our case $x_1(n)$ starts at $n = -1$ and $x_2(n)$ starts at $n = 0$. In this case we have to fix the starting and ending indices to find $x(n)$. We write the following MATLAB script to do that.

```
x1 = [1 1 1];
n1 = [-1 0 1];
x2 = [1 1];
n2 = [0 1];
ns = n1(1) + n2(1); % the starting minimum
ne= n1(length (x1)) + n2(length(x2)); % the ending maximum
n = [ns : ne];
x = conv(x1, x2);
[n' x'] % to display the convolution result with the index n
```

We will get

```
–1  1
 0  2
 1  2
 2  1
```

and from this we have

$$x(n) = \delta(n+1) + 2\delta(n) + 2\delta(n-1) + \delta(n-2)$$

EOCE 4.6

When

$$x(n) = (.5)^n u(n) + (.3)^n u(n) + (.9)^n u(n)$$

what is $X(z)$ and its ROC?

Solution

Using the linearity property of the z-transform, we write

$$X(z) = \frac{z}{z-.5} + \frac{z}{z-.3} + \frac{z}{z-.9}$$
$$= \frac{z(z-.3)(z-.9) + z(z-.5)(z-.9) + z(z-.5)(z-.3)}{(z-.5)(z-.3)(z-.9)}$$

and the ROC is $|z| > .3$ and $|z| > .5$ and $|z| > .9$ all satisfied simultaneously. The ROC becomes $|z| > .9$.

EOCE 4.7

Consider the transforms

$$X_1(z) = 1 + z^{-1} + 3z^{-2}$$

and

$$X_2(z) = 1 + 3z^{-2}$$

What is $x(n) = x_1(n) * x_2(n)$?

Solution

$$x_1(n) = \delta(n) + \delta(n-1) + 3\delta(n-2)$$

and

$$x_2(n) = \delta(n) + 3\delta(n-2)$$

We can use MATLAB to find $x(n) = x_1(n) * x_2(n)$ by writing the script

```
x1=[1 1 3];
x2=[1 0 3];
x=conv(x1,x2);
```

to get

$x = \{ \underset{\uparrow}{1}\ 1\ 6\ 3\ 9\}$

and

$$x(n) = \delta(n) + \delta(n-1) + 6\delta(n-2) + 3\delta(n-3) + 9\delta(n-4)$$

We can also use the z-transform to arrive at this result.

$$X(z) = X_1(z)X_2(z) = \left(1 + z^{-1} + 3z^{-2}\right)\left(1 + 3z^{-2}\right)$$
$$= 1 + 3z^{-2} + z^{-1} + 3z^{-3} + 3z^{-2} + 9z^{-4}$$

and then the inverse z-transform is

$$x(n) = \delta(n) + \delta(n-1) + 6\delta(n-2) + 3\delta(n-3) + 9\delta(n-4)$$

EOCE 4.8

Use MATLAB, long division and partial function expansion to find $h(n)$ if

$$H(z) = \frac{1}{z^2 - 3z + 2} \text{ with ROC } |z| > 2$$

Using MATLAB, first we need to put $X(z)$ in ascending powers of z^{-1} and write

$$H(z) = \frac{z^{-2}}{1 - 3z^{-1} + 2z^{-2}}$$

Then we will use the MATLAB function residuez that has the form

```
[r  p  k] = residuez (num, den)
```

where `num` and `den` are the coefficients of the numerator and the denominator of the rational z-transformed function. `r` is a vector that contains the residues, `p` in the vector that contains the poles and `k` is the constant term that is nonzero if the degree of `num` is larger or equal to the degree of `den`, the numerator and the denominator of the rational z-transformed function. With this we write the script

```
num=[ 0 0 1];
den=[1 -3 2 ];
[r,p,k]=residuez(num,den)
```

to get

r = 0.5000 and –1.0000
p = 2 and 1
k = 0.5

This gives the transfer function

$$H(z) = \frac{1}{2} - \frac{z}{z-1} + \frac{\frac{1}{2}z}{z-2}$$

from which we get

$$h(n) = \frac{1}{2}\delta(n) - u(n) + \frac{1}{2}(2)^n u(n)$$

using Table 4.1. We can see that the first few terms of $h(n)$ are

$$h(0) = 1/2 - 1 + 1/2 = 0$$

$$h(1) = 0 - 1 + 1 = 0$$

$$h(2) = 0 - 1 + 2 = 1$$

We can also use MATLAB to find some terms of $h(n)$. To do that, we can use the MATLAB function `filter` with an impulsive input. We have

$$Y(z) = X(z)\,H(z)$$

If $X(z) = 1$, ($h(n) = \delta(n)$), then $Y(z) = H(z)$ and $y(n)$ is $h(n)$. To do that we write the script

```
num = [0 0 1];
den = [1 -3 2];
x = [1 zeros(1, 2)]; % the impulse input of 3 samples.
h = filter(num, den, x)% h(n) for only the first 3 samples
```

to get the same result we found earlier.

Using long division, we can divide the numerator by the denominator to get

$$H(z) = \frac{1}{z^2 - 3z + 2} = 0z^0 + 0z^{-1} + z^{-2} + 3z^{-3} + \ldots$$

which clearly indicates that $h(0) = 0$, $h(1) = 0$ and $h(2) = 1$ as we saw earlier.

Using partial fraction expansion, we write the transfer function as

$$H(z)/z = \frac{1}{z(z-1)(z-2)} = \frac{A}{z} + \frac{B}{z-1} + \frac{C}{z-2}$$

with the constants evaluated as

$$A = \frac{1}{(z-1)(z-2)}\bigg|z=0 = \frac{1}{2}$$

$$B = \frac{1}{z(z-2)}\bigg|z=1 = -1$$

$$C = \frac{1}{z(z-1)}\bigg|z=2 = \frac{1}{2}$$

So the transfer function becomes

$$H(z) = \frac{(1/2)z}{z} - \frac{-z}{z-1} + \frac{(1/2)z}{z-2}$$

and

$$h(n) = \frac{1}{2}\delta(n) - u(n) + \frac{1}{2}(2)^n u(n)$$

EOCE 4.9

Find $h(n)$ if $H(z)$ is

$$H(z) = \frac{1}{z^2 - .9z + .7} = \frac{z^{-2}}{1 - .9z^{-1} + .7z^{-2}}$$

Assume the resulting signal $h(n)$ is valid only for $n \geq 0$ and it is real.

Solution

Since this $H(z)$ is not similar to any form in Table 4.1, we need to use partial fraction expansion on $H(z)$. We do that using MATLAB and write the script

```
num=[0 0 1];
den=[1 -.9 .7];
[r p k]=residuez(num,den)
mag_r= abs(r)
phase_r=angle(r)
mag_p=abs(p)% magnitude of the poles
phase_p=angle(p)% phase angles of the poles
```

to get

r = –0.7143 – 0.4557i and –0.7143 + 0.4557i
p = 0.4500 + 0.7053i and 0.4500 – 0.7053i
k = 1.4286
mag_r = 0.8473 and 0.8473
phase_r = –2.5737 and 2.5737
mag_p = 0.8367 and 0.8367
phase_p = 1.0029 and –1.0029

Hence we can see that

$$H(z) = 1.428 + \frac{0.847e^{-j2.57}}{1-|0.8367|e^{j1.0029}z^{-1}} + \frac{0.847e^{j2.57}}{1-|0.8367|e^{-j1.0029}z^{-1}}$$

with the ROC as $|z| > 0.8367$. Now we can use Table 4.1 to get the inverse transform $h(n)$. We write

$$h(n) = 1.428\delta(n) + 0.847e^{-j2.57}(.8367)^n e^{j(1.0029)n} + 0.847e^{j2.57}(.8367)^n e^{-j(1.0029)n}$$

These two terms are complex conjugate terms. We know that the sum of two complex conjugate terms is two times the real part of the complex number. Thus we write

$$\begin{aligned} h(n) &= 1.428\delta(n) + 2\text{Real}\left(0.847e^{-j2.57}(.8367)^n e^{j(1.0029)n}\right) \\ &= 1.428\delta(n) + 2(0.847)(.8367)^n \text{Real}\left(e^{j(1.0029n-2.57)}\right) \end{aligned}$$

After some simplifications we get

$$h(n) = 1.428\delta(n) + \left[2(0.847)(.8367)^n \cos(1.0029n - 2.57)\right] u(n)$$

EOCE 4.10

Consider the system transfer function in the z-domain

$$H(z) = \frac{z}{z^3 - 6z^2 + 11z - 6}$$

with ROC $|z| > 2$. Find $h(n)$.

Solution

First we write $H(z)$ in the proper form so that we can use partial function to find $h(n)$. $H(z)$ then is

$$H(z) = \frac{z^{-2}}{1 - 6z^{-1} + 11z^{-2} - 6z^{-3}}$$

By putting $H(z)$ is partial fraction form we use MATLAB and write the script

```
num = [0 0 1 ]
den = [ 1 -6 11 -6 ]
[ r p k ] = residuez (num, den)
```

to get

r = 0.5000, –1.0000 and 0.5000
p = 3.0000, 2.0000 and 1.0000
k = [ ] to indicate zero value

Then the partial fraction expansion results in

$$H(z) = \frac{0.5}{1 - z^{-1}} + \frac{-1}{1 - 2z^{-1}} + \frac{0.5}{1 - 3z^{-1}}$$

To bring $H(z)$ back into real-time we need to look carefully at the ROC for $H(z)$. For the first term $0.5/(1 - z^{-1}) = 0.5z/(z - 1)$,the pole is at $z = 1$. The ROC of $H(z)$ is outside the ring $|z| = 2$ and hence outside the ring $|z| = 1$. In this case

$$\frac{0.5}{1 - z^{-1}} \leftrightarrow 0.5u(n)$$

The second term $-1/(1 - 2z^{-1}) = -z/(z - 2)$ has the pole $z = 2$. The ROC of $H(z)$ is outside the ring $|z| = 2$ and hence

$$\frac{-1}{1 - 2z^{-1}} \leftrightarrow -(2)^n u(n)$$

The last term $0.5/(1 - 3z^{-1}) = 0.5z/(z - 3)$ has a pole at $z = 3$. The ROC of $H(z)$ is outside the $|z| = 2$ circle and $z = 3$ is within this ROC. Thus

$$\frac{0.5}{1-3z^{-1}} \leftrightarrow -0.5(3)^n u(-n-1)$$

Therefore, the inverse transform is

$$h(n) = 0.5u(n) - (2)^n - 0.5(3)^n u(-n-1)$$

Note: When you use MATLAB to find $h(n)$ be careful to pay attention to the ROC of $H(z)$.

EOCE 4.11

Consider the following causal difference equation

$$y(n) - .7y(n-1) = x(n)$$

where $y(n)$ is the output and $x(n)$ is the input.

1. Find the transfer function $H(z)$.
2. Find the transfer function $H(e^{j\theta})$.
3. Find $h(n)$, the impulse response.
4. Find $y(n)$ if $x(n) = u(n)$.

Solution

1. With zero initial conditions we transform the difference equation term by term into the z-domain. We will have

   $$Y(z) - .7z^{-1}Y(z) = X(z)$$

   Taking $Y(z)$ as a common factor we get

   $$Y(z)\left[1 - .7z^{-1}\right] = X(z)$$

   and the transfer function is

   $$\frac{Y(z)}{X(z)} = \frac{1}{1-.7z^{-1}} = \frac{z}{z-.7}$$

   with ROC $|z| > .7$.
2. Since $H(z)$ converges for $|z| > .7$, the unit circle is inside the ROC and so we can find $H(e^{j\theta})$ from $H(z)$ as

$$H\left(e^{j\theta}\right) = H(z)\Big|_{z=e^{j\theta}} = \frac{e^{j\theta}}{e^{j\theta} - .7} = \frac{1}{1 - .7e^{-j\theta}}$$

3. $h(n)$ can be obtained directly from Table 4.1 as

$$h(n) = (.7)^n u(n)$$

4. With $h(n) = (.7)^n u(n)$ and $x(n) = u(n)$ we have $y(n) = x(n) * h(n)$. Or in z-domain we have

$$Y(z) = \frac{z}{z-1}\frac{z}{z-.7} = \frac{z^2}{z^2 - 1.7z + .7} = \frac{1}{1 - 1.7z^{-1} + .7z^{-2}}$$

At this point we can determine the ROC for $Y(z)$ as the intersection of region $|z| > 1$ and $|z| > .7$. Thus the ROC for $Y(z)$ is $|z| > .7$. We can use MATLAB to find $Y(z)$ in partial fraction form by writing the script

```
num = [ 1];
den = [ 1 -1.7 .7 ];
[r, p, k] = residuez (num, den)
```

to get

r = 3.3333 and –2.3333
p = 1.0000 and 0.7000

Thus

$$Y(z) = \frac{3.332}{1 - z^{-1}} + \frac{-2.3333}{1 - 0.7z^{-1}}$$

The inverse transform of the above equation gives

$$y(n) = 3.332(1)^n u(n) - 2.3333(0.7)^n u(n)$$

We can use MATLAB to plot the results obtained in this example. We will use the MATLAB function `freqz` as

```
[ H, theta ] = freqz(num, den, N, 'whole')
```

where `H` is the vector that contains all the frequency response values, `theta` the vector that contains the frequency values, `num` and `den` are the coefficients of the numerator and the denominator of the transfer function, and

`whole` indicates that `freqz` will evaluate `H` for the `N theta` values around the entire unit circle.

The MATLAB script is given next.

```
num = [ 1 0 ];
den = [ 1 -.7];
[ H, theta ] = freqz(num, den, 200, 'whole');
mag_H = abs (H);
angle_H = angle(H);
% Next we will generate an impulse input of length 20
ximp = [ 1 zeros(1, 19) ];
% Next we generate a step input of length 20
xstp = ones(1, 20);
yimp = filter(num, den, ximp);
ystp = filter(num, den, xstp);
subplot(2, 2, 1); plot(theta/pi, mag_H);
xlabel('Frequency is pi units');
ylabel('Magnitude');
subplot(2, 2, 2); plot(theta/pi, angle_H/pi);
xlabel('Frequency in pi units'); ylabel('Angle in pi units');
n=0:19;
subplot(2, 2, 3); plot(n, yimp);
xlabel('n'); ylabel('Impulse response');
subplot(2, 2, 4); plot(n,ystp);
xlabel('n'); ylabel('Step response');
```

The plots are as in Figure 4.6.

EOCE 4.12

Consider the system transfer function

$$H(z) = \frac{2z^2 - 12z}{(z - .3)(z + .2)(z - .3)}$$

Find $h(n)$ for the following ROCs and indicate stability.

1. $|z| > 3$
2. $|z| < .2$
3. $.2\ |z| < .3$
4. $.3 < |z| < 3$
5. $.2 < |z| < 3$

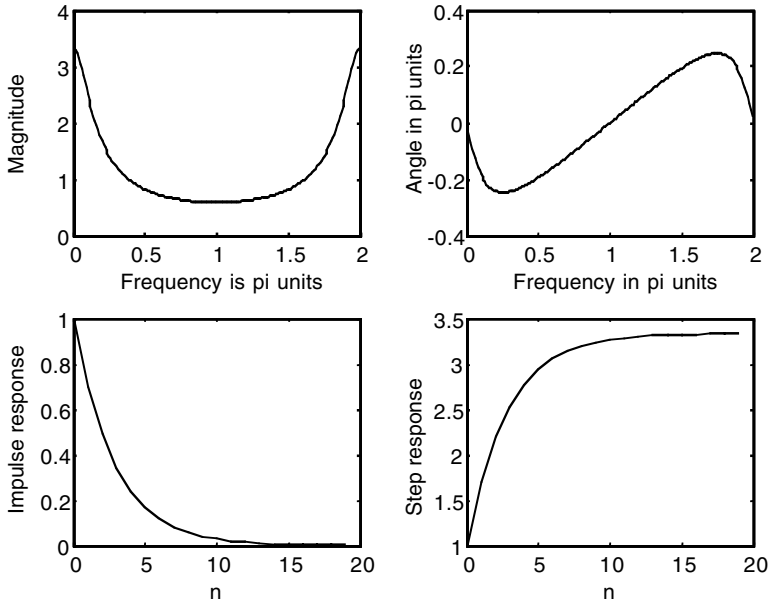

FIGURE 4.6 Signals for EOCE 4.11.

Solution

The poles of the system are at z =.3, z = –.2 and z = 3.

1. $|z| > 3$ is the ROC that is outside the ring $|z| = 3$. The poles at .3, –.2 and 3 are inside this ring, but the ring $|z| = 1$ is not inside this ROC. Therefore the system is unstable and causal with

$$h(n) = c_1(.3)^n u(n) + c_2(-.2)^n u(n) + c_3(3)^n u(n)$$

2. $|z| < .2$ is the ROC that is inside the ring $|z| = .2$. The $|z| = 1$ ring is not in this ROC. Also, none of the system poles are inside this ROC. Therefore, the system is unstable and noncausal with

$$h(n) = c_4(.3)^n u(-n-1) + c_5(-.2)^n u(-n-1) + c_6(3)^n u(-n-1)$$

3. $.2 < |z| < .3$ is the ROC that is between the two rings $|z| = .2$ and $|z| = .3$. The $|z| = 1$ ring is not in this ROC, hence instability. The ROC is inside $|z| = 3$ and $|z| = .3$, with

$$h(n) = c_6(.3)^n u(-n-1) + c_8(-.2)^n u(n) + c_9(3)^n u(-n-1)$$

4. $.3 < |z| < 3$ is the ROC that is between the two rings $|z| = .3$ and $|z| = 3$. The ring $|z| = 1$ is inside this ROC and hence the system is stable. The ROC is inside $|z| = 3$ and thus

$$h(n) = c_{10}(-.2)^n u(n) + c_{11}(.3)^n u(n) + c_{12}(3)^n u(-n-1)$$

5. $.2 < |z| < 3$ is the ROC that is between the two rings $|z| = .2$ and $|z| = 3$. The ring $|z| = 1$ is within this ROC. Thus the system is stable. The ROC is inside $|z| = 3$ and $|z| = .3$. Thus we have

$$h(n) = c_{13}(-.2)^n u(n) + c_{14}(.3)^n u(-n-1) + c_{15}(3)^n u(-n-1)$$

EOCE 4.13

MATLAB has a function called `filter` that can be used to find the response to a discrete system using initial conditions that are different from zero. The form of this function is

```
y = filter(num, den, x, y0)
```

where `y` is the output vector, `num` is the numerator coefficients vector, `den` is the denominator coefficients vector, `x` is the input vector, and `y0` is the initial conditions vector that is derived using the actual initial conditions given to you with the system under investigation. In real-time the N^{th} order difference equation can be represented as

$$y(n) + a_1 y(n-1) + \ldots + a_N y(n-N) = b_0 x(n) + b_1 x(n-1) + \ldots + b_L x(n-L)$$

The z-transform of the above equation is

$$Y(z) = \frac{b_0 + b_1 z^{-1} + \ldots + b_L z^{-L}}{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-N}}$$

The derived initial conditions for the filter function are computed as in the following.

$$\begin{aligned}
y_{00} &= -a_1 y(-1) - \ldots - a_N y(-N) \\
y_{01} &= -a_2 y(-1) - a_3 y(-2) - \ldots - a_N y(-N+1) \\
&\vdots \\
y_{0N} &= -a_N y(-1)
\end{aligned}$$

Consider the following system

$$y(n) - y(n-1) = x(n)$$

with $y(-1) = 1$ and $x(n) = \delta(n)$. Find $y(n)$ using the z-transform.

Solution

We can z-transform the difference equation term by term and write

$$Y(z) - \left[y(-1) + z^{-1}Y(z)\right] = X(z)$$

We can rearrange terms and write

$$Y(z)\left[1 - z^{-1}\right] = X(z) + y(-1)$$

We finally have

$$Y(z) = \frac{X(z)}{1 - z^{-1}} + \frac{y(-1)}{1 - z^{-1}}$$

With $X(z) = 1$, the output is

$$Y(z) = \frac{z}{z-1} + \frac{z}{z-1} = \frac{2z}{z-1}$$

Using Table 4.1 we get

$$y(n) = 2u(n)$$

Using MATLAB we first write $\dfrac{Y(z)}{X(z)}$ without initial conditions to get

$$\frac{Y(z)}{X(z)} = \frac{1}{1 - z^{-1}} = \frac{b_0}{1 + (a_1)z^{-1}}$$

The initial condition vector y_0 will have one component since the system is first order. So the derived initial condition is

$$y_{00} = -a_1 y(-1) = -1(1) = -1$$

Next we write the MATLAB script to find $y(n)$ for $x(n) = \delta(n)$.

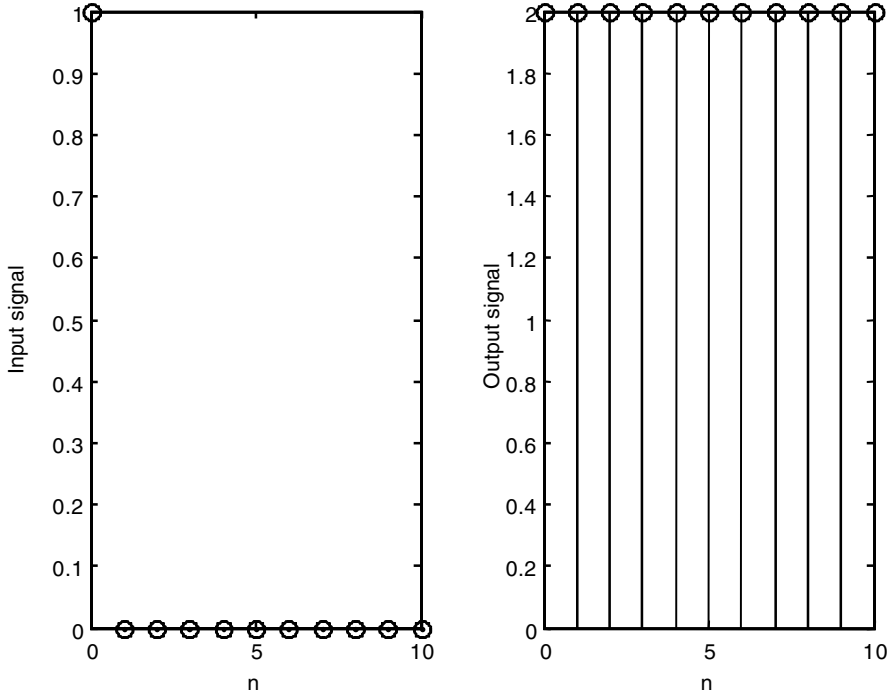

FIGURE 4.7 Signals for EOCE 4.13.

```
n = 0 : 10;
num = [ 1 ];
den = [ 1 -1 ];
y0=1;% this is the derived initial condition
x = [ 1 zeros(1,length(n)-1) ];
y=filter(num, den,x,y0);
subplot(1, 2, 1); stem(n, x); xlabel('n'), ylabel('Input
 signal');
subplot(1, 2, 2); stem(n, y); xlabel('n'), ylabel('Output
 signal');
```

The plots are shown in Figure 4.7.

We can also use MATLAB with the MATLAB function `filtic` to find the derived initial conditions for us. The general form of `filtic` is

```
ic=filtic(num,den,x0,y0)
```

where `ic` is the returned derived initial conditions. `num` and `den` are as discussed before. `x0` and `y0` are the initial condition vectors given with the problem for negative n. We can rework the MATLAB solution using the `filtic` function as in the script

```
n = 0 : 10;
num = [ 1 ];
den = [ 1 -1 ];
y0 = 1;% the given initial condition
ic=filtic(num,den,y0); %the derived initial condition
x = [ 1 zeros(1,(length(n)-1)) ];
y = filter(num, den, x, ic);
subplot(1, 2, 1); stem(n, x); xlabel('n'), ylabel('Input
 signal');
subplot(1, 2, 2); stem(n, y); xlabel('n'), ylabel('Output
 signal');
```

and the result is the same.

EOCE 4.14

Consider the following system

$$y(n) - .5y(n-1) - .1y(n-2) + .2y(n-3) = x(n)$$

with $x(n) = u(n)$ and $y(-1) = 1$, $y(-2) = 2$ and $y(-3) = 3$,

1. Find the output $y(n)$ using MATLAB.
2. Find the system transfer function.
3. Is the system stable?

Solution

1. The initial condition vector y_0 for the function `filter` is determined next.

$$y_{00} = -a_1 y(-1) - a_2 y(-2) - a_3 y(-3) = .5(1) + .1(2) - .2(3) = .1$$

$$y_{01} = -a_2 y(-1) - a_3 y(-2) = .1(1) - .2(2) = -.3$$

$$y_{02} = -a_3 y(-1) = -.2(1) = -.2$$

We can use MATLAB to determine these derived initial conditions by writing the following script

```
num = [ 1 ];
den = [ 1 -.5 -.1 .2 ];
y0 = [1 2 3];% the given initial condition
ic=filtic(num,den,y0); %the derived initial condition
```

The result is

ic = 0.1000 –0.3000 –0.2000

With these derived initial conditions we can write the following MATLAB script to plot the response due to the step input.

```
n = 0 : 15;
num = [ 1 ];
den = [ 1 -.5 -.1 .2 ];
y0 = [ 1 2 3];% the given initial condition
ic=filtic(num,den,y0); %the derived initial condition
x = [ ones(length(n)) ];
y = filter(num, den, x, ic);
subplot(1, 2, 1); stem(n, x); xlabel('n'), ylabel('Input
 signal');
subplot(1, 2, 2); stem(n, y); xlabel('n'), ylabel('Output
 signal');
```

The plots are shown in Figure 4.8.

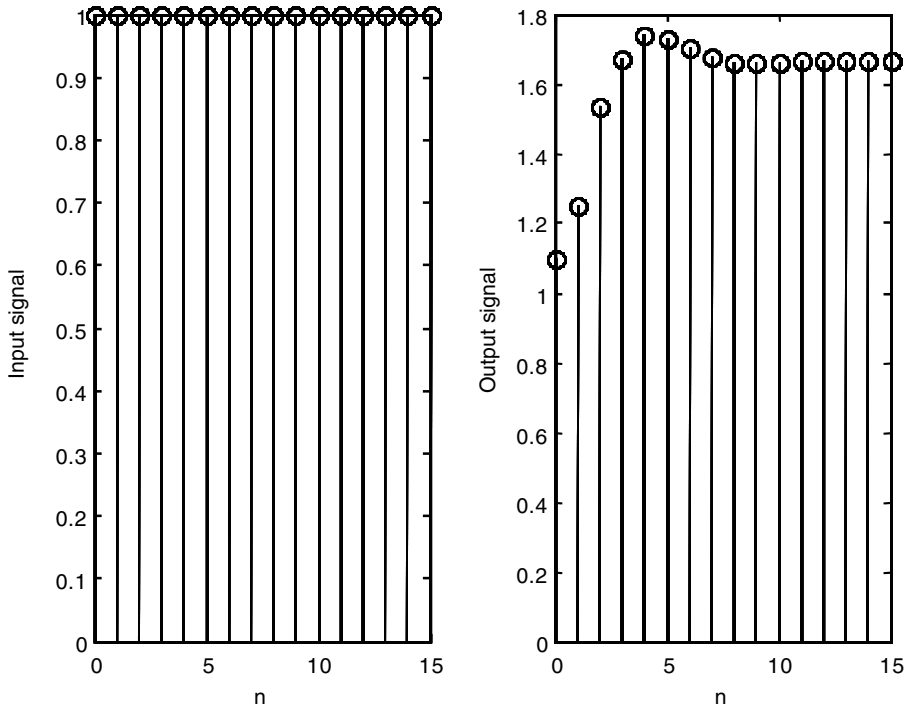

FIGURE 4.8 Signals for EOCE 4.14.

2. The system transfer function is obtained by taking the z-transform of the difference equation with zero initial conditions. The result is

$$Y(z) - .5z^{-1}zY(z) - .1z^{-2}Y(z) + .2z^{-3}Y(z) = X(z)$$

By grouping similar terms we get

$$Y(z)\left[1 - .5z^{-1} - .1z^{-2} + .2z^{-3}\right] = X(z)$$

and the transfer function is

$$\frac{Y(z)}{X(z)} = H(z) = \frac{1}{1 - .5z^{-1} - .1z^{-2} + .2z^{-3}}$$

3. We can use MATLAB to find the poles of $H(z)$. We type at the MATLAB prompt

```
r=roots([1 -.5 -.1 .2]);
abs(r)%the magnitude of the roots
```

to get 0.5000, 0.6325, and 0.6325. The roots are within the unit circle and thus the system is stable.

EOCE 4.15

Consider the system

$$y(n) - \frac{1}{2}y(n-1) = x(n) + x(n-1)$$

1. Find the impulse response $h(n)$.
2. What is the step response?
3. Use MATLAB to find $y(n)$ if $x(n) = (.5)^n \sin(n)\, u(n)$.

Solution

1. If we can find $H(z)$ for the given system, we will inverse transform $H(z)$ to get $h(n)$. Let us z-transform the given difference equation to get

$$Y(z) - \frac{1}{2}z^{-1}Y(z) = X(z) + z^{-1}X(z)$$

and the transfer function is then

$$\frac{Y(z)}{X(z)} = H(z) = \frac{1+z^{-1}}{1-\frac{1}{2}z^{-1}}$$

The ROC is $|z| > 0.5$. Using partial fraction expansion on $H(z)$, we have

$$\frac{H(z)}{z} = \frac{z+1}{z\left(z-\left(1/2\right)\right)} = \frac{A}{z} + \frac{B}{z-\left(1/2\right)}$$

with

$$A = \left.\frac{z+1}{z-\left(1/2\right)}\right|_{z=0} = -2$$

and

$$B = \left.\frac{z+1}{z}\right|_{z=1/2} = \frac{1/2+1}{\left(1/2\right)} = \frac{\left(3/2\right)}{\left(1/2\right)} = 3$$

Thus

$$H(z) = \frac{-2z}{z} + \frac{3z}{z-\left(1/2\right)}$$

and from Table 4.1 we get

$$h(n) = -2\delta(n) + 3\left(\frac{1}{2}\right)^n u(n)$$

2. The step response is obtained with $X(z) = \frac{z}{z-1}$. The output will be

$$Y(z) = X(z)H(z) = \frac{z+1}{z-\frac{1}{2}}\,\frac{z}{z-1}$$

We will do partial fraction expansion on

$$\frac{Y(z)}{z} = \frac{z+1}{\left(z-\frac{1}{2}\right)(z-1)} = \frac{A}{z-\frac{1}{2}} + \frac{B}{z-1}$$

to get

$$A = \left.\frac{z+1}{z-1}\right|_{z=\frac{1}{2}} = \frac{\frac{1}{2}+1}{\frac{1}{2}-1} = \frac{3/2}{-1/2} = -3$$

$$B = \left.\frac{z+1}{z-\frac{1}{2}}\right|_{z=1} = \frac{1+1}{1-\frac{1}{2}} = \frac{2}{1/2} = 4$$

The resulting output in the z-domain is

$$Y(z) = \frac{-3z}{z-\frac{1}{2}} + \frac{4z}{z-1}$$

and the inverse z-transform is

$$y(n) = -3\left(\frac{1}{2}\right)^n u(n) + 4u(n)$$

We can use MATLAB to verify this solution by writing the script

```
n = 0: 20;
num = [1 1];
den = [1 -1/2];
x = [ones(length(n))];
ymatlab = filter(num, den, x);
yanalyt = -3*(0.5.^n) + 4*(1.^n) ;
subplot (3, 1, 1), stem(n, x), ylabel('Input');
subplot (3, 1, 2), stem(n, ymatlab), ylabel('MATLAB
 output');
subplot (3, 1, 3), stem(n,yanalyt); ylabel('Analytical');
xlabel ('n');
```

The plots are shown in Figure 4.9.

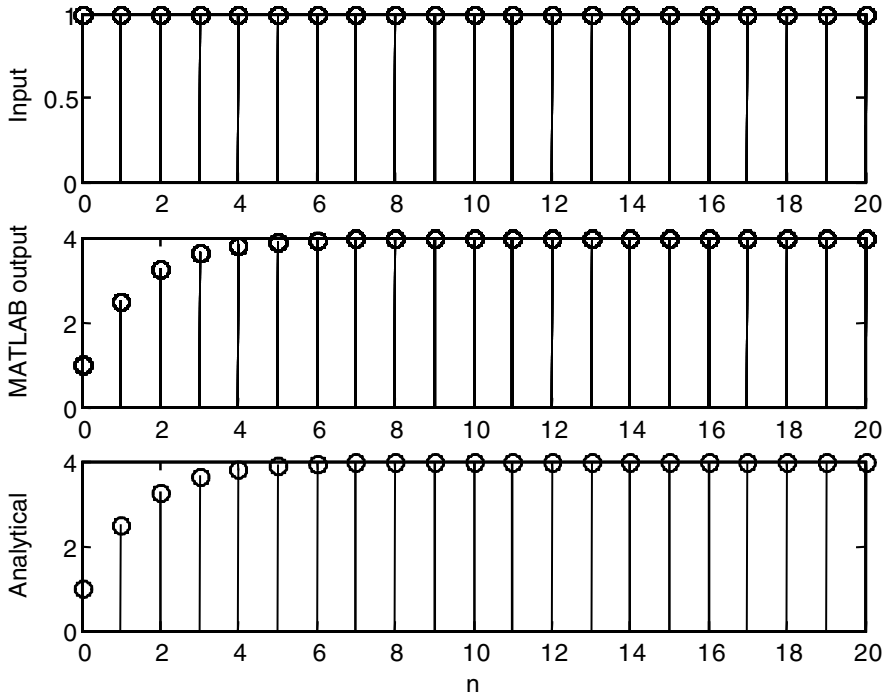

FIGURE 4.9 Signals for EOCE 4.15.

3. We will use MATLAB to do that.

```
n = 0: 20;
num = [1 1];
den = [1 -1/2];
x = ((.5).^n).*sin(n);
y = filter(num, den, x);
subplot (2, 1, 1), stem(n, x), ylabel('Input');
zeroline=zeros(1, length(n));
hold on; plot(n,zeroline);hold off;
subplot (2, 1, 2), stem(n, y), ylabel('Matlab output');
hold on; plot(n,zeroline);hold off;
xlabel ('n');
```

The plots are shown in Figure 4.10.

EOCE 4.16

Consider the following system

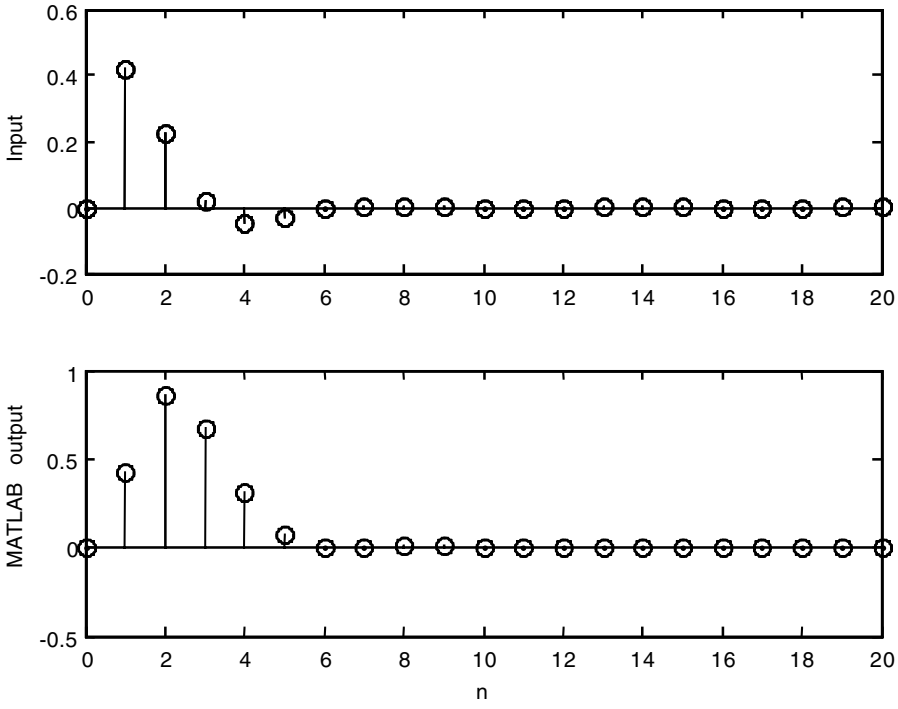

FIGURE 4.10 Signals for EOCE 4.15.

$$H(z) = \frac{z}{z^2 + 1}$$

What is $h(n)$, the impulse response?

Solution

We will use partial fraction expansion to find $h(n)$. As we did previously, we will put $H(z)/z$ in partial fraction form, then multiply by z to introduce a z into the numerator. By doing so we will easily inverse-transform back to real-time since the entries in Table 4.1 have z in the numerator.

$$\frac{H(z)}{z} = \frac{1}{z^2 + 1} = \frac{A}{z - j} + \frac{B}{z + j}$$

We next find the constants and write

$$\frac{H(z)}{z} = \frac{-1/2j}{z - j} + \frac{1/2j}{z + j}$$

Using Table 4.1 we get

$$h(n)=\frac{-1}{2}j(j)^n u(n)+\frac{1}{2}j(-j)^n u(n)=\frac{-1}{2}j\left(e^{j\frac{\pi}{2}n}\right)u(n)+\frac{1}{2}j\left(e^{-j\frac{\pi}{2}n}\right)u(n)$$

By taking common factors we get

$$h(n)=\frac{1}{2}j\left[e^{-j\frac{\pi}{2}n}-e^{j\frac{\pi}{2}n}\right]u(n)=-\frac{1}{2}j\left[e^{j\frac{\pi}{2}n}-e^{-j\frac{\pi}{2}n}\right]u(n)$$

and finally

$$h(n)=-\frac{1}{2}(2j)j\sin\left(\frac{\pi}{2}n\right)u(n)=\sin\left(\frac{\pi}{2}n\right)u(n)$$

EOCE 4.17

Consider the system

$$H(z)=\frac{z}{z^2-2z+2}$$

What is $h(n)$?

Solution

The impulse response $h(n)$ is the inverse z-transform of $H(z)$. We will put $H(z)/z$ in partial fraction form first.

$$\frac{H(z)}{z}=\frac{1}{z^2-2z+2}=\frac{A}{z-(1+j)}+\frac{B}{z-(1-j)}$$

and $H(z)$ will be

$$H(z)=\frac{zA}{z-(1+j)}+\frac{Bz}{z-(1-j)}.$$

From this last expression for $H(z)$ and using Table 4.1 we get

$$h(n)=A\left(\sqrt{2}\right)^n e^{j\frac{\pi}{4}n}u(n)+B\left(\sqrt{2}\right)^n e^{-j\frac{\pi}{4}n}u(n)$$

where $\sqrt{2}$ is the magnitude of $1- j$ and $1 + j$ and $-\pi/4$ is the phase of $1 - j$.

We still need to find A and B. We can use MATLAB to do that and to confirm some of the results that we arrived at (the magnitude and the phase of the poles). We will write the script

```
n = [ 0 1 0 ];
d = [1 -2 2 ];
[r, p, k] = residuez(n, d);
mag_r = abs(r)
phase_r=angle(r)
mag_p = abs(p)
phase_p=angle(p)
```

to get

mag_r = 0.5000 0.5000
phase_r = –1.5708 1.5708
mag_p = 1.4142 1.4142
phase_p = 0.7854 –0.7854

The impulse response is then

$$h(n) = 0.5e^{-1.57j}\left(\sqrt{2}\right)^n e^{j\frac{\pi}{4}n}u(n) + 0.5e^{1.57j}\left(\sqrt{2}\right)^n e^{-j\frac{\pi}{4}n}u(n)$$

In the above expression we have two complex conjugate terms. The sum of two conjugate terms is two times the real part of the term. Thus we write

$$h(n) = 2\text{Real}\left[0.5e^{-1.57j}\left(\sqrt{2}\right)^n e^{j\frac{\pi}{4}n}\right] = \left(\sqrt{2}\right)^n \text{Real}\left[e^{-1.57j}e^{j\frac{\pi}{4}n}\right]$$

$$h(n) = \left(\sqrt{2}\right)^n \text{Real}[e^{j(\frac{\pi}{4}n-1.57)}] = \left(\sqrt{2}\right)^n \cos\left(\frac{\pi}{4}n - 1.57\right)u(n)$$

EOCE 4.18

Consider the system

$$y(n) - .4y(n-1) = x(n)$$

with $x(n) = 2sin\ (2\pi/4\ n)u(n)$. What are the initial and final values for $y(n)$ with $x(n)$ as the input?

Solution

With

$$x(n) = 2\sin\left(\frac{2\pi}{4}n\right)u(n)$$

$X(z)$ is

$$X(z) = \frac{2z^{-1}\left(\sin\frac{\pi}{4}\right)}{1-\left(2\cos\left(\frac{\pi}{4}\right)\right)z^{-1}+z^{-2}} = \frac{2z\sin\left(\frac{\pi}{4}\right)}{z^2-\left(2\cos\left(\frac{\pi}{4}\right)\right)z+1} = \frac{\sqrt{2}z}{z^2-\sqrt{2}z+1}$$

We know that $Y(z) = X(z)H(z)$. But from the difference equation, $H(z)$ is

$$H(z) = \frac{1}{1-.4z^{-1}} = \frac{z}{z-.4}$$

Therefore,

$$Y(z) = \frac{z}{z-.4}\ \frac{\sqrt{2}z}{z^2-\sqrt{2}z+1}$$

The initial value of $y(n)$ is $y(0)$. Using the initial value theorem we can find $y(0)$ without solving for $y(n)$ from $Y(z)$.

$$y(0) = \lim_{z\to\infty} Y(z) = 0$$

The final value of $y(n)$ is

$$y(\infty) = \lim_{z\to 1}(z-1)\frac{z}{z-.4}\frac{\sqrt{2}z}{z^2-\sqrt{2}z+1}$$

provided that all poles of $(z-1)\,Y(z)$ lie inside the unit circle. This is not the case here because the input is a pure sinusoid and it has its pole on the unit circle. Without paying attention to this we may mistakenly use this theorem and write

$$y(\infty) = \lim_{z\to 1}(z-1)\frac{z}{z-.4}\frac{\frac{\sqrt{2}}{2}z}{z^2-\sqrt{2}z+1} = 0$$

But we know that the output $y(n)$ is also sinusoidal and does not settle on a single value as n approaches infinity. However, if $x(n) = u(n)$ then

$$Y(z) = \frac{z}{z-1}\frac{z}{z-.4}$$

and

$$y(0) = \lim_{z\to\infty} \frac{z^2}{(z-1)(z-.4)} = 1$$

The final value in this case will be

$$y(\infty) = \lim_{z\to 1}(z-1)\frac{z}{z-1}\frac{z}{z-0.4} = \frac{1}{1-.4} = \frac{1}{.6} = \frac{10}{6}$$

EOCE 4.19

Consider the system in Figure 4.11.

1. Find the transfer function representation.
2. Find the difference equation representation.
3. Find $y(n)$ if $x(n) = u(n)$.
4. Is the system stable?

Solution

1. The signal after the first summer is $X(z) + Y(z)$. This signal then passes through a delay and becomes $[X(z) + Y(z)]\, z^{-1}$. Then it passes through another delay and becomes $[[X(z) + Y(z)]z^{-1}]z^{-1}$. The output of the last summer is $Y(z)$ which is

$$Y(z) = 2X(z) + \left[\left[X(z) + Y(z)\right]z^{-1}\right]z^{-1} = 2X(z) + X(z)z^{-2} + Y(z)z^{-2}$$

By grouping like terms we arrive at

$$Y(z)\left[1 - z^{-2}\right] = X(z)\left[2 + z^{-2}\right]$$

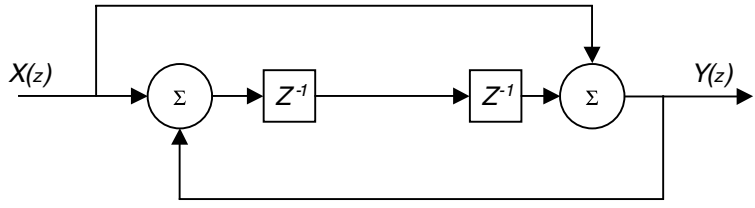

FIGURE 4.11 System for EOCE 4.19.

And the transfer function is

$$\frac{Y(z)}{X(z)} = H(z) = \frac{2+z^{-2}}{1-z^{-2}} = \frac{z^2+2}{z^2-1}$$

2. From

$$\frac{Y(z)}{X(z)} = \frac{2+z^{-2}}{1-z^{-2}}$$

we can write

$$Y(z)\left[1-z^{-2}\right] = X(z)\left[2+z^{-2}\right]$$

The inverse transform is

$$y(n) - y(n-2) = 2x(n) + x(n-2)$$

3. If $x(n) = u(n)$ then $X(z) = \dfrac{z}{z-1}$. The output then is

$$Y(z) = \frac{z^2+2}{z^2-1}\frac{z}{z-1}$$

$$\frac{Y(z)}{z} = \frac{z^2+2}{\left(z^2-1\right)(z-1)} = \frac{z^2+2}{(z-1)^2(z+1)} = \frac{A}{z+1} + \frac{B}{(z-1)^2} + \frac{C}{z-1}$$

The constants are found next.

$$A = \left.\frac{z^2+2}{(z-1)^2}\right|_{z=-1} = \frac{1+2}{4} = 3/4$$

$$B = \left.\frac{z^2+2}{z+1}\right|_{z=1} = \frac{3}{2}$$

$$C = \frac{d}{dz}\left(B(z)\right) = \left.\frac{2z(z+1)-\left(z^2+2\right)}{(z+1)^2}\right|_{z=1} = \frac{4-3}{4} = \frac{1}{4}$$

Therefore, the output in the z-domain is

$$Y(z) = \frac{\frac{3}{4}z}{z+1} + \frac{\frac{3}{2}z}{(z-1)^2} + \frac{\frac{1}{4}z}{z-1}$$

The output in the time domain is

$$y(n) = \frac{3}{4}(-1)^n u(n) + \frac{3}{2}(1)^n nu(n) + \frac{1}{4}(1)^n u(n)$$

This solution can be verified using MATLAB as in the script

```
n = 0: 20;
num = [2 0 1];
den = [1 0 -1];
x = [ones(length(n))];
ym = filter(num, den, x);
ya = 3/4*(-1.^n) + 3/2*n.*(1.^n)+1/4*(1.^n) ;
subplot (3, 1, 1), stem(n, x), ylabel('Input');
subplot (3, 1, 2), stem(n, ym), ylabel('MATLAB output');
subplot (3, 1, 3), stem(n,ya); ylabel('Analytical');
xlabel ('n');
axis([0 20 0 40]);
```

The plots are in Figure 4.12.

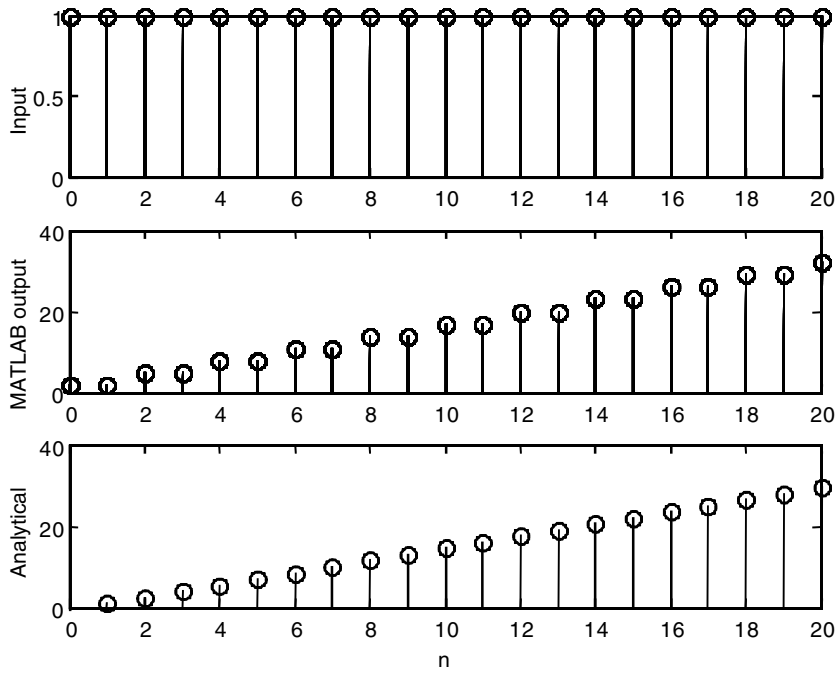

FIGURE 4.12 Signals for EOCE 4.19.

4. To check stability we look at the poles of

$$H(z) = \frac{z^2 + 2}{z^2 - 1} = \frac{z^2 + 2}{(z-1)(z+1)}$$

We can see that the poles are at $z = 1$ and $z = -1$ and they are on the unit circle. Hence the system is on the verge of stability.

EOCE 4.20

Consider the system in Figure 4.13, with

$$H_1(z) = \frac{z}{z - .5} = H_2(z)$$

1. Find the transfer function.
2. Is the system stable?
3. Find the general form of $h(n)$ if $h(n)$ is real and causal.
4. If $x(n) = u(n)$, what is $y(n)$?

Solution

1. From the Figure 4.13 we see that the output is

$$y(n) = \left[x(n) * h_1(n)\right] * h_2(n) + x(n) * h_2(n)$$

and from the properties of the z-transform we write $Y(z)$ as

$$Y(z) = X(z)H_1(z)H_2(z) + X(z)H_2(z) = X(z)\left[H_1(z)H_2(z) + H_2(z)\right]$$

and the transfer function is

$$\frac{Y(z)}{X(z)} = H(z) = H_1(z)H_2(z) + H_2(z)$$

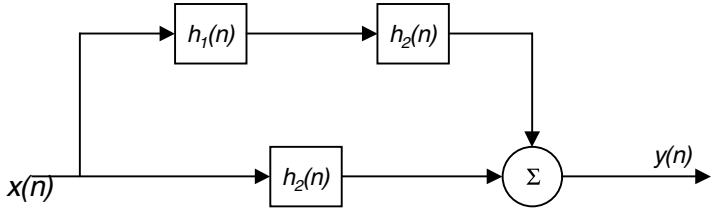

FIGURE 4.13 Signals for EOCE 4.20.

By substituting the individual transfer functions given to us we arrive at

$$H(z)=\frac{z}{z-.5}\frac{z}{z-.5}+\frac{z}{z-.5}=\frac{z^2+z(z-.5)}{(z-.5)^2}=\frac{2z^2-.5z}{(z-.5)^2}=\frac{z(2z-.5)}{(z-.5)^2}$$

2. The poles are at $z = .5$ and they are within the unit circle. This means that the system is stable.
3. Since we are assuming that $h(n)$ is real and causal then

$$\frac{H(z)}{z}=\frac{2z-.5}{(z-.5)^2}=\frac{c_1}{(z-.5)^2}+\frac{c_2}{z-.5}$$

and

$$H(z)=\frac{zc_1}{(z-.5)^2}+\frac{zc_2}{z-.5}$$

With the help of Table 4.1 we get

$$h(n)=c_1 n(.5)^n u(n)+c_2(.5)^n u(n)$$

This means that the ROC for $H(z)$ is $|z| > .5$. If the ROC for $H(z)$ were $|z| < .5$ then

$$h(n)=c_3 n(.5)^n u(-n-1)+c_4(.5)^n u(-n-1)$$

4. If the input $x(n) = u(n)$ then $X(z) = \frac{z}{z-1}$ and

$$Y(z)=\left[H_1(z)H_2(z)+H_2(z)\right]\left(\frac{z}{z-1}\right)$$

$$Y(z)=\frac{z}{(z-.5)^2}\frac{z}{z-1}+\frac{z}{z-.5}\frac{z}{z-1}$$

Let us write the output as the sum

$$Y(z)=Y_1(z)+Y_2(z)$$

and perform partial fraction expansion as

$$\frac{Y_1(z)}{z}=\frac{z}{(z-.5)^2(z-1)}=\frac{c_1}{(z-.5)^2}+\frac{c_2}{z-.5}+\frac{c_3}{z-1}$$

$$\frac{Y_2(z)}{z}=\frac{z}{(z-.5)(z-1)}=\frac{c_4}{z-.5}+\frac{c_5}{z-1}$$

Next we find the constants as

$$c_1 = \left.\frac{z}{z-1}\right|_{\frac{1}{2}} = \frac{1/2}{1/2-1} = -1$$

$$c_2 = \left.\frac{d}{dz}c_1(z)\right|_{z=\frac{1}{2}} = \left.\frac{z-1-z}{(z-1)^2}\right|_{\frac{1}{2}} = \frac{-1}{1/4} = -4$$

$$c_3 = \left.\frac{z}{(z-.5)^2}\right|_{z=1} = \frac{1}{1/4} = 4$$

and

$$c_4 = \left.\frac{z}{z-1}\right|_{z=\frac{1}{2}} = \frac{1/2}{-1/2} = -1$$

$$c_5 = \left.\frac{z}{z-.5}\right|_{z=1} = \frac{1}{1-.5} = 2$$

The output in the z-domain is then

$$Y(z) = \frac{-1z}{(z-.5)^2} - \frac{4z}{z-.5} + \frac{4z}{z-1} - \frac{1z}{z-0.5} + \frac{2z}{z-1}$$

The output in the time domain is

$$y(n) = -n(.5)^n u(n) - 4(.5)^n u(n) + 4u(n) - (.5)^n u(n) + 2u(n)$$

EOCE 4.21

If $x(n) = \delta(n) + \delta(n-3)$ and the output $y(n)$ is $\delta(n-4)$, what is the transfer function $H(z)$? Assume the system is linear and time invariant.

Solution

We know that $y(n) = x(n) * h(n)$ in the discrete time domain. Mathematically

$$y(n) = \sum_{m=-\infty}^{+\infty} x(m)h(n-m)$$

An easier way of finding $H(z)$ is to work directly with the z-domain by taking advantage of the convolution property of the z-transform

$$Y(z) = H(z)\ X(z)$$

After substitution we get

$$z^{-4} = H(z)\left[1 + z^{-3}\right]$$

and finally

$$H(z) = \frac{z^{-4}}{1 + z^{-3}} = \frac{1}{z^4 + z} = \frac{1}{z\left(z^3 + 1\right)}$$

4.15 End of Chapter Problems

EOCP 4.1

Find the *z*-transform for the following signals and indicate the ROC.

1. $x(n) = 3(.3)^n u\ (n)$
2. $x(n) = (.3)^n u\ (n) - (.3)^n u(-n - 1)$
3. $x(n) = u(n) - u(n - 1)$
4. $x(n) = \sin\left(n\frac{\pi}{3}\right)u(n) + (.3)^n u(-n - 1)$
5. $x(n) = u(n)*(.5)^n\ u(n)$
6. $x(n) = u(n)*(.5)^n\ u(n)*(.5)^n\ u(-n - 1)$
7. $x(n) = nu(n) - n\sin\left(\frac{2\pi}{3}n\right)u(n)$
8. $x(n) = (n - 1)\ u(n - 1) - 2\delta(n - 1)$
9. $x(n) = u(-n - 1)*\ u(n) + (n - 1)\sin((n - 1)\pi/4)\ u(n - 1)$
10. $x(n) = n\ (.5)^n \sin(n)\ u(n) + u(-n - 1)$

EOCP 4.2

Consider the following signals in the *z*-transform domain. Find $h(n)$ for each case.

1. $H(z) = 10\,\frac{z}{z - .5}$ ROC : $|z| < .5$
2. $H(z) = \frac{z}{(z - 1)(z - .5)}$ ROC : $|z| > .5$
3. $H(z) = \frac{1}{(z - .3)(z + 2)}$ ROC : $|z| < 2$

4. $H(z) = \dfrac{z^2 + z + 2}{(z-3)(z+2)(z-.1)}$ ROC : $.1 < |z| < 3$
5. $H(z) = \dfrac{z^2 + z + 2}{(z-3)(z+2)(z-.1)}$ ROC : $|z| > -2$
6. $H(z) = \dfrac{z+1}{(z-.5)(z-.5)}$ ROC : $|z| > .5$
7. $H(z) = \dfrac{z+1}{(z-.5)^2(z-.3)}$ ROC : $.3 < |z| < .5$
8. $H(z) = \dfrac{1}{(z^2 + z + 1)(z-.5)}$ ROC : $|z| < .5$
9. $H(z) = \dfrac{z^2 + z}{(z^2 + 2z + 2)(z-.1)(z-.3)}$ ROC : $|z| > .1$
10. $H(z) = \dfrac{(z-1)(z+1)}{z}$ ROC : $|z| > 0$

EOCP 4.3

The following signals will produce a causal $h(n)$. Find $h(n)$ using partial fraction, long division and MATLAB.

1. $H(z) = \dfrac{1}{z(z-1)}$
2. $H(z) = \dfrac{z}{z(z-1)}$
3. $H(z) = \dfrac{z^2 + z + 1}{z^2 + 5z + 6}$
4. $H(z) = \dfrac{z+1}{z^2 + 2z + 4}$
5. $H(z) = \dfrac{z^3 + z^2 + z + 1}{z}$
6. $H(z) = \dfrac{z^2 + 1}{z^3 + 2z^2 + 4z}$

EOCP 4.4

Draw the block diagrams for the following systems in the z-domain.

1. $\dfrac{Y(z)}{X(z)} = \dfrac{1}{z^2(z-1)}$
2. $\dfrac{Y(z)}{X(z)} = \dfrac{z^2 + z}{z^2 + 5z + 6}$
3. $\dfrac{Y(z)}{X(z)} = \dfrac{z^2 + z + 1}{z^2 + 2z + 2}$

4. $\dfrac{Y(z)}{X(z)} = \dfrac{z^2 + 2z + 1}{z}$
5. $\dfrac{Y(z)}{X(z)} = \dfrac{z}{z^2 + 2z + 1}$

EOCP 4.5

Find $\dfrac{Y(z)}{X(z)}$ for the block diagrams in Figures 4.14 through 4.18.

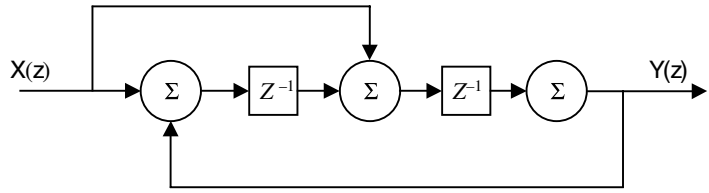

FIGURE 4.14 Block for EOCP 4.5.

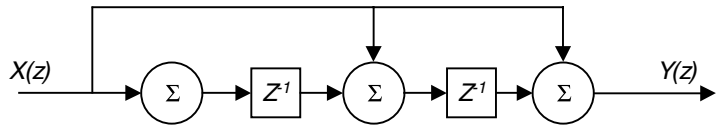

FIGURE 4.15 Block for EOCP 4.5.

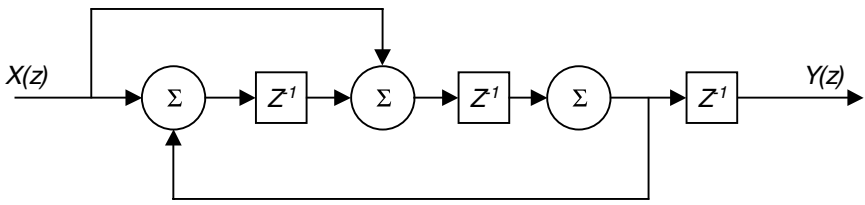

FIGURE 4.16 Block for EOCP 4.5.

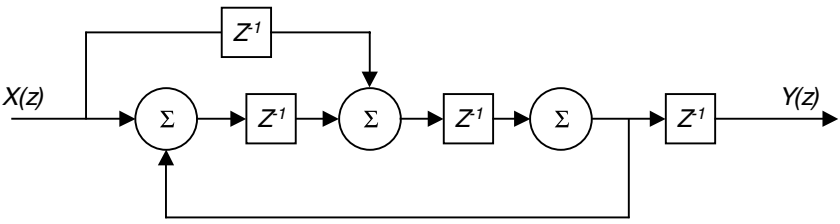

FIGURE 4.17 Block for EOCP 4.5.

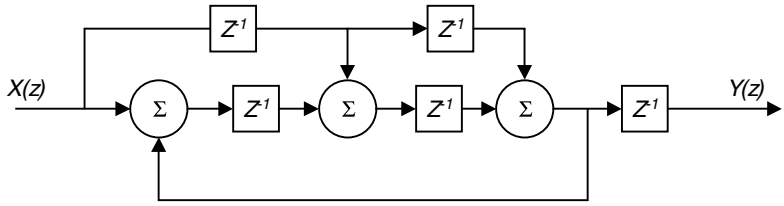

FIGURE 4.18 Block for EOCP 4.5.

EOCP 4.6

For each block diagram in Figures 4.19 through 4.23 find $y(n)$, the step response.

EOCP 4.7

Consider the following transfer functions. Find the difference equations representing these systems and indicate if any of them is stable. Use MATLAB to find $h(n)$.

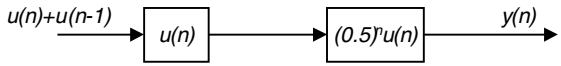

FIGURE 4.19 Block for EOCP 4.6.

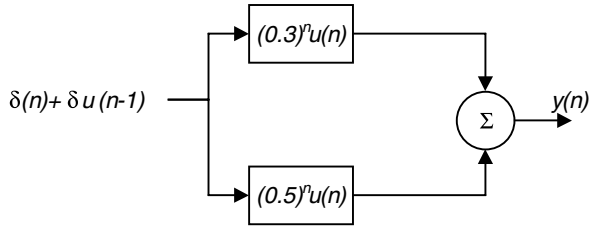

FIGURE 4.20 Block for EOCP 4.6.

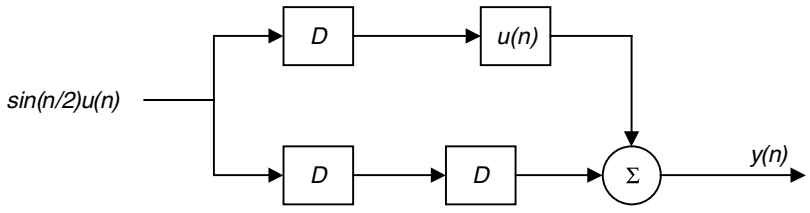

FIGURE 4.21 Block for EOCP 4.6.

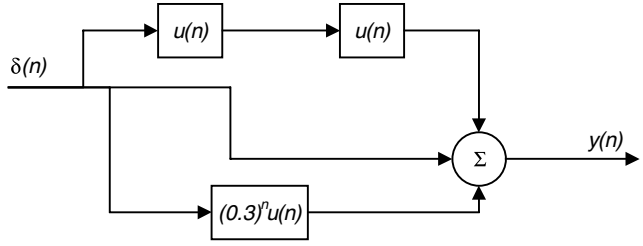

FIGURE 4.22 Block for EOCP 4.6.

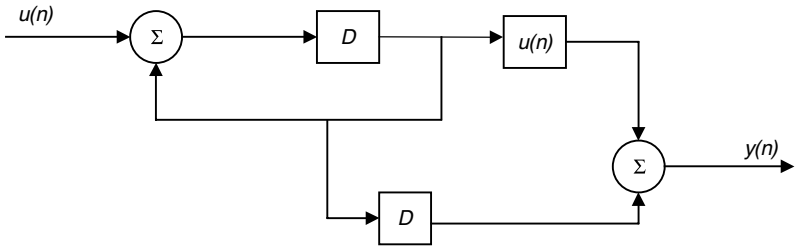

FIGURE 4.23 Block for EOCP 4.6.

1. $H(z) = \dfrac{z+2}{z^2+2z+5}$
2. $H(z) = \dfrac{z+2}{z^3+2z^2+5z}$
3. $H(z) = \dfrac{1}{(2z^3-1)(z-.1)}$
4. $H(z) = \dfrac{z^2+z+1}{(z^2+.2)z}$
5. $H(z) = \dfrac{1}{z^3+3z^2+2z+1}$

EOCP 4.8

Consider the following difference equations. Find the transfer functions and the outputs $y(n)$. Are the systems stable?

1. $y(n) - .5y(n-1) = x(n) + x(n-1),\ y(-1) = 0,\ x(n) = \delta(n)$
2. $y(n) - .5y(n-1) = x(n) + x(n-1),\ y(-1) = 1,\ x(n) = \delta(n)$
3. $y(n) - .5y(n-1) = x(n) + x(n-1),\ y(-1) = 0,\ x(n) = \sin(2\pi/1n)u(n)$
4. $y(n) - .3y(n-1) = x(n) + x(n-1),\ y(-1) = -1,\ x(n) = \sin(2\pi n/3n)$ $u(n) + \cos(2\pi n/3)u(n)$
5. $y(n) - .8y(n-1) + .2y(n-2) = x(n),\ y(-1) = 0\ ,\ y(-2) = 1,\ x(n) = \delta(n)$

6. $y(n) - .8y(n-1) + .2y(n-2) = x(n-1),\ y(-1) = 1,\ y(-2) = 1, x(n) = u(n)$
7. $y(n) + y(n-1) + y(n-2) = x(n),\ y(-1) = y(-2) = 0, x(n) = u(n)$
8. $y(n) + .1y(n-2) = x(n-2), y(-1) = y(-2) = 1, x(n) = u(n)$
9. $y(n) - .5y(n-3) = x(n), y(-1) = y(-2) = 0, y(-3) = 1, x(n) = \delta(n)$
10. $y(n) = x(n) + x(n-1) + x(n-2) + x(n-3), x(n) = \sin(n)u(n)$

EOCP 4.9

The following are outputs of linear discrete systems. Find $y(0)$ and $y(\infty)$ for each output.

1. $Y(z) = \dfrac{z}{z-10}$
2. $Y(z) = \dfrac{z^2+z}{z^2+2z+1}$
3. $Y(z) = \dfrac{z^2+2z+1}{(z-1)(z+2)}$
4. $Y(z) = \dfrac{z^2+3z+1}{(z-.1)(z-.2)(z-.3)}$
5. $Y(z) = \dfrac{z}{z-1} + \dfrac{z}{(z-1)^2}$

EOCP 4.10

Find the steady-state output for the systems

1. $H(z) = \dfrac{1}{z-.5} \quad x(n) = \sin(n)u(n) + \cos(n)\,u(n)$
2. $H(z) = \dfrac{1}{(z-.2)(z-.3)} \quad x(n) = \sin\left(\dfrac{2\pi}{3}n\right)u(n)$
3. $H(z) = \dfrac{z+1}{z} \quad x(n) = 10$
4. $H(z) = \dfrac{z+1}{z^2} \quad x(n) = \delta(n) + \mu(n)$
5. $H(z) = \dfrac{z^2+z+1}{z} \quad x(n) = 3$

EOCP 4.11

Consider the following difference equation

$$y(n) - .5y(n-1) - .3y(n-2) - .4y(n-3) = x(n)$$

a) With $y(-1) = y(-2) = y(-3) = 0$ and

1. $x(n) = \delta(n)$
2. $x(n) = u(n)$
3. $x(n) = \cos(10\pi n/3)u(n)$

Use MATLAB to find $y(n)$ assuming that $y(n)$ is causal; use long division to verify the MATLAB results.

b) With $y(-1) = y(-2) = y(-3) = -1$ and

1. $x(n) = \delta(n)$
2. $x(n) = u(n)$
3. $x(n) = \cos(10\pi n/3)u(n)$

Use MATLAB to find $y(n)$ assuming that $y(n)$ is causal; use long division to verify the MATLAB results.

c) Is the system stable?
d) Find $h(n)$ if ROC is $|z| > 2$.

EOCP 4.12

Consider the system in Figure 4.24.

1. Find the difference equation representing the system.
2. For what value(s) of a is the system stable?
3. Pick a value for a to make the system stable, and find $y(n)$ for $x(n) = \delta(n)$
4. Use MATLAB to find $y(n)$ for $a = .5$ if
   a) $x(n) = 10\ \delta(n)$
   b) $x(n) = 10\ \sin(3/4\ \pi)u(n)$
5. Find the initial and final values for $y(n)$ is 4.

EOCP 4.13

Consider the following system in Figure 4.25.

1. Find $\dfrac{Y(z)}{X(z)}$ as a function of $F(z)$, the feedback function.
2. If $F(z) = 1$, find $\dfrac{Y(z)}{X(z)}$.
3. Is the system stable in 2? If yes, for what k values?
4. If $F(z) = \dfrac{z}{z-.2}$, find $\dfrac{Y(z)}{X(z)}$.
5. Is the system stable in 4? If yes, for what k values?

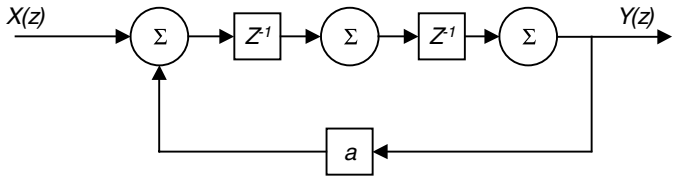

FIGURE 4.24 Block for EOCP 4.12.

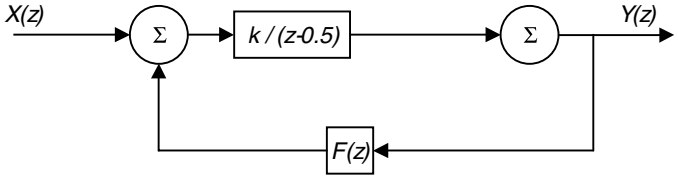

FIGURE 4.25 Block for EOCP 4.13.

6. Find the difference equation representing the system for both cases $F(z) = 1$ and $F(z) = \dfrac{z}{z-.2}$.
7. Find the impulse response of the system when $F(z) = 1$ and $F(z) = \dfrac{z}{z-.2}$ with a value of k that makes the system stable.
8. Find the step response for both $F(z) = 1$ and $F(z) = \dfrac{z}{z-.2}$ with a k value that stabilizes the system.
9. Let $E(z) = X(z) - F(z)\,Y(z)$. Find this error signal $E(z)$ as a function of k.
10. Find $e(0)$ and $e(\infty)$. $e(n)$ is called the error signal.
11. What value for k will make $e(\infty) = .1$? Pick a suitable input $x(n)$ that will give you this k value.

EOCP 4.14

The outputs and the inputs are given for an unknown linear time-invariant system. What is the transfer function for each?

1. $x(n) = \delta(n) + \delta(n-1) + \delta(n-2)$
   $y(n) = \delta(n-1) + \delta(n-2) + \delta(n-3)$
2. $x(n) = \sin\left(\dfrac{2\pi}{3}n\right)u(n)$
   $y(n) = \cos\left(\dfrac{2\pi}{3}n\right)u(n)$
3. $x(n) = u(n)$
   $y(n) = nu(n)$

4. $x(n) = \delta(n)$
   $y(n) = u(n)$
5. $x(n) = 2\sin\left(\frac{3\pi}{11}n\right)u(n) + u(n)$
   $y(n) = 10\cos\left(\frac{3\pi}{11}n\right)u(n)$

5

State-Space and Discrete Systems

5.1 Introduction

As the interest in many scientific fields increased, modeling systems using linear time-invariant equations and tools such as transfer functions were not adequate. The state-space approach is superior to other classical methods of modeling. This modern approach can handle systems with nonzero initial conditions (modeling using transfer functions requires that initial conditions be set to zero) as well as time-variant systems. It can also handle linear and nonlinear systems. We also have been considering systems with single-input single-output. The state-space approach can handle multiple-input multiple-output systems.

Systems can have many variables. An example is an electrical circuit where the variables are the inductor current, the capacitor voltage and the resistor voltage among others. With the state-space approach we can solve for a selected set of these variables. The other variables in the circuit system can be found using the solution for the selected variables.

Using the state-space approach we will follow the subsequent procedure. We will select specific variables in the system and call them state variables. No state variable selected can be written as a linear combination of the other state variables. Linear combination means that if

$$v_1(t) = 3v_3(t) + 2v_2(t) \tag{5.1}$$

where $v_1(t)$, $v_2(t)$ and $v_3(t)$ are state variables, we say that $v_1(t)$ is a linear combination of $v_2(t)$ and $v_3(t)$. If we have a first-order differential or difference equation, we will have only one state variable. If the differential or the difference equation is second order, we will have only two state variables. Similarly, if we have an n^{th} order differential or difference equation, we will have only n state variables. Once we select or decide on the state variables in the system under consideration, we will write a set of first-order simultaneous differential or difference equations where the right side of these equations is a function only of the state variables (no derivatives or shifts) and the inputs to the

system, and the number of these equations is determined by the number of state variables selected. We will call this set the state equation set. These state equations will be solved for the selected state variables. All other variables in the system under consideration can be solved using the solutions of these selected state variables and the inputs to the system. We can use any approach we desire to solve for these selected states. The equations we write to a set of selected outputs in the system are called output equations.

The above discussion relates closely to continuous systems and explains the evolution of state equations from real physical systems. In many cases, we derive discrete systems from continuous systems by many means such as sampling or transformation. The equivalence of the differential equations is the difference equations where the basic unit is the delay element. From these difference equations the states in discrete form are obtained. Some systems are inherently discrete and the difference equation is readily obtained. The concept of the state is best understood with dynamic physical systems and that is how we explain it here.

5.2 A Review on Matrix Algebra

What follows is a brief review of some of the concepts and definitions we need in this chapter. We will deal with second-order systems when we deal with hand solutions. For matrices of higher dimensions you can consult any linear algebra book.

5.2.1 Definition, General Terms and Notations

A matrix is a collection of elements arranged in a rectangular or square array. The size of the matrix is determined by the number of rows and the number of columns in the matrix. A matrix **A** of m rows and n column is represented as $\mathbf{A}_{mxn}$. If $m = 1$ then **A** is a row vector and is written as $\mathbf{A}_{1xn}$. If $n = 1$ then **A** is a column vector and is written as $\mathbf{A}_{mx1}$. If $n = m$ then **A** is a square matrix and we write it as $\mathbf{A}_{nxn}$ or $\mathbf{A}_{mxm}$. If all elements in the matrix are zeros we say **A** is a null matrix or a zero matrix.

5.2.2 The Identity Matrix

The identity matrix is the square matrix where elements along the main diagonal are ones and elements off the main diagonal are zeros. A two-by-two identity matrix is

$$\mathbf{I}_{2x2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{5.2}$$

5.2.3 Adding Two Matrices

If $\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} e & f \\ g & h \end{pmatrix}$ then

$$\mathbf{A} + \mathbf{B} = \begin{pmatrix} a+e & b+f \\ c+g & d+h \end{pmatrix} \tag{5.3}$$

To add two matrices they must be of the same size. If the matrices are of higher order, the procedure is the same; we add the corresponding entries.

5.2.4 Subtracting Two Matrices

If $\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} e & f \\ g & h \end{pmatrix}$ then

$$\mathbf{A} - \mathbf{B} = \begin{pmatrix} a-e & b-f \\ c-g & d-h \end{pmatrix} \tag{5.4}$$

To subtract two matrices they must be of the same size. If the matrices are of higher order, the procedure is the same; we subtract the corresponding entries.

5.2.5 Multiplying A Matrix by a Constant

If $\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and k is any given constant, then

$$k \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} ka & kb \\ kc & kd \end{pmatrix} \tag{5.5}$$

If the matrix **A** is of higher order, then k is multiplied by each entry in **A**.

5.2.6 Determinant of a Two-by-Two Matrix

Consider the $A_{2\times2}$ matrix

$$\mathbf{A}_{2\times2} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

The determinant of **A** is

$$\det(\mathbf{A}) = ad\text{-}bc \tag{5.6}$$

5.2.7 Transpose of A Matrix

If $\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ then the transpose of **A** is given by

$$\mathbf{A}^T = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \tag{5.7}$$

This works for higher order matrices as well, where the first column in **A** becomes the first row in $\mathbf{A}^T$ and so on.

5.2.8 Inverse of A Matrix

If $\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ then the inverse of **A** is

$$\mathbf{A}^{-1} = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \tag{5.8}$$

and since $\frac{1}{ad-bc}$ is a constant, we can write the inverse as

$$\mathbf{A}^{-1} = \begin{pmatrix} \frac{d}{ad-bc} & \frac{-b}{ad-bc} \\ \frac{-c}{ad-bc} & \frac{a}{ad-bc} \end{pmatrix}$$

The inverse of a square matrix exists if the determinant of the matrix is not zero. Also, to find an inverse of a certain matrix, that matrix has to be square.

The procedure above for finding the inverse is only for a 2×2 matrices. For higher order matrices the procedure is different and is found in any linear algebra book.

5.2.9 Matrix Multiplication

We can multiply two matrices **A** and **B** if the number of columns in **A** is equal to the number of rows in **B**. If $\mathbf{A}_{mxn}$ is to be multiplied by $\mathbf{B}_{rxp}$ then n must be equal to r and the resulting matrix should have m rows and n columns.

$$\text{If } \mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \text{ and } \mathbf{B} = \begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

then if we multiply **A** by **B** and let matrix **C** hold the resulting product, the size of **C** is 2×2. We could multiply **A** by **B** because the number of columns in **A**, which is two, is equal to the number of rows in **B**, which is also two. The multiplication of **A** by **B** is **C** and it is

$$\mathbf{C} = \mathbf{AB} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{pmatrix} \tag{5.9}$$

We multiply the first row of **A**, element by element, by all the columns of **B**. Similarly, we take the second row of **A** and multiply it by all the columns of **B**. Note that in general **AB** is not the same as **BA**. The rules for multiplication have to be observed.

5.2.10 Eigenvalues of a Matrix

The eigenvalues of a matrix **A** are the roots of the determinant of $(\lambda\mathbf{I} - \mathbf{A})$, where **I** is the identity matrix and λ is a variable.

5.2.11 Diagonal Form of a Matrix

A matrix **A** is in diagonal form if all elements in the matrix that are off the diagonal are zeros. If **A** is not diagonal, we can make it diagonal by finding the matrix **P** that contains the eigenvectors of **A**. So if **A** is not a diagonal matrix then $(\mathbf{P}^{-1}\mathbf{AP})$ will transform **A** into a diagonal matrix. If the eigenvalues of **A** are distinct (distinct means that no one eigenvalue is equal to the other) then $\mathbf{P}^{-1}\mathbf{AP}$ will look like a diagonal matrix where the elements on the main diagonal are the eigenvalues of **A**. If some of the eigenvalues of **A** are repeated, then the matrix $\mathbf{P}^{-1}\mathbf{AP}$ will be in a block diagonal form or what is known as Jordan form.

5.2.12 Eigenvectors of a Matrix

The eigenvectors of a matrix **A** are the nonzero roots of the homogeneous matrix equation

$$(\lambda_i \mathbf{I} - \mathbf{A})\mathbf{p}_i = \mathbf{0} \tag{5.10}$$

where $\mathbf{p}_i$ is a column vector that represents the eigenvector for a certain eigenvalue λ_i. All eigenvectors must be independent. If we have n distinct eigenvalues, then we will have n independent eigenvectors for each eigenvalue, and each eigenvector is obtained as the nonzero solution to $(\lambda_i \mathbf{I} - \mathbf{A})\mathbf{p}_i = \mathbf{0}$.

If $\mathbf{A}$ is nxn and we have $n - k$ distinct eigenvalues (again distinct means no one eigenvalue is equal to any of the remaining $n - 1$ eigenvalues) then we will have $n - k$ independent eigenvectors for each distinct eigenvalue and each eigenvector is obtained as the nonzero solution of $(\lambda_i \mathbf{I} - \mathbf{A})\mathbf{p}_i = \mathbf{0}$. Each remaining eigenvalue in the set k of eigenvalues is therefore not distinct. We may have groups of repeated eigenvalues in the k set of eignenvalues. We will assume here that we have only one set which is k and thus we have k repeated eigenvalues. The following procedure can be applied to other repeated sets in the set k.

Assume that $k = 3$. Let the repeated eigenvalues be denoted as λ_r and let us denote the eigenvectors as $\mathbf{p}_{1r}$, $\mathbf{p}_{2r}$ and $\mathbf{p}_{3r}$.

To find $\mathbf{p}_{1r}$ we will find the nonzero solution to $(\lambda_r \mathbf{I} - \mathbf{A})\mathbf{p}_{1r} = \mathbf{0}$

To find $\mathbf{p}_{2r}$ we will find the nonzero solution to $(\lambda_r \mathbf{I} - \mathbf{A})\mathbf{p}_{2r} = \mathbf{p}_{1r}$

To find $\mathbf{p}_{3r}$ we will find the nonzero solution to $(\lambda_r \mathbf{I} - \mathbf{A})\mathbf{p}_{3r} = \mathbf{p}_{2r}$

These are called generalized eigenvectors corresponding to the three repeated eigenvalues. In some situations, even if we have repeated eigenvalues, we still can get independent eigenvectors if the matrix $\mathbf{A}$ is symmetric. We do that by finding a nonzero solution to $(\lambda_i \mathbf{I} - \mathbf{A})\mathbf{p}_i = \mathbf{0}$ where λ_i is the repeated eigenvalue.

5.3 General Representation of Systems in State-Space

Thus far we have seen many representations of discrete linear time-invariant systems. We have seen the difference equation representation, the block diagram representation, the z-transform representation, and the impulse response representation. Given any representation we should be able to deduce the other. In this section we will study the state-space representation starting with the difference equation. Using state-space representation, a difference equation of order n can be reduced to n first-order difference equations.

5.3.1 Recursive Systems

Consider the fourth order equation

$$y(n) - a_1 y(n-1) - a_2 y(n-2) - a_3 y(n-3) - a_4 y(n-4) = b_o x(n)$$

We will have four states v_1 through v_4 since we have a fourth-order difference equation. Let

$$v_1(n) = y(n-4)$$

$$v_2(n) = y(n-3)$$

$$v_3(n) = y(n-2)$$

$$v_4(n) = y(n-1)$$

from which we write

$$v_1(n+1) = y(n-4+1) = y(n-3) = v_2(n)$$

$$v_2(n+1) = y(n-3+1) = y(n-2) = v_3(n)$$

$$v_3(n+1) = y(n-2+1) = y(n-1) = v_4(n)$$

and we can also write the output as

$$y(n) = a_1y(n-1) + a_2y(n-2) + a_3y(n-3) + a_4y(n-4) + b_0x(n)$$

$$= a_1v_4(n) + a_2v_3(n) + a_3v_2(n) + a_4v_1(n) + b_0x(n)$$

Therefore, we can finally write

$$v_1(n+1) = v_2(n)$$

$$v_2(n+1) = v_3(n)$$

$$v_3(n+1) = v_4(n)$$

$$v_4(n+1) = a_4v_1(n) + a_3v_2(n) + a_2v_3(n) + a_1v_4(n) + b_0x(n)$$

Notice that the right side of all of the above four state equations is a function of the states $v_1(n)$ through $v_4(n)$ and the input. This should be the case. No term such as $v_1(n-1)$ or $v_2(n-3)$ should appear on the right side of these equations.

We can now arrange the state equations in matrix form and write

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \\ v_4(n+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ a_4 & a_3 & a_2 & a_1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \\ v_4(n) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ b_0 \end{pmatrix} \mathbf{x}(n)$$

and for the output equation we have

$$\mathbf{y}(n) = \begin{pmatrix} a_4 & a_3 & a_2 & a_1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \\ v_4(n) \end{pmatrix} + \mathrm{b}_0\mathbf{x}(n)$$

with

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ a_4 & a_3 & a_2 & a_1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ b_0 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} a_4 & a_3 & a_2 & a_1 \end{pmatrix} \quad \mathbf{D} = (b_0)$$

We can write the state and output equations in matrix form as

$$\mathbf{v}(n + 1) = \mathbf{A}\,\mathbf{v}(n) + \mathbf{B}\,\mathbf{x}(n)$$

$$\mathbf{y}(n) = \mathbf{C}\,\mathbf{v}(n) + \mathbf{D}\,\mathbf{x}(n)$$

When $\mathbf{v}(n + 1)$ is a 4×1 matrix, $\mathbf{A}$ is 4×4 matrix, $\mathbf{B}$ is 4×1 matrix, $\mathbf{C}$ is a 1×4 matrix and $\mathbf{D}$ is a 1×1 matrix. Notice that these state equations are now in the form of a first-order matrix difference equation.

5.3.2 Nonrecursive Systems

Consider the difference equation

$$y(n) = a_0x(n) + a_1x(n - 1) + a_2x(n - 2) + a_3x(n - 3)$$

Let

$$v_1(n) = x\,(n - 3)$$

$$v_2(n) = x\,(n - 2)$$

$$v_3(n) = x\,(n - 1)$$

Then

$$v_1(n + 1) = x\,(n - 2) = v_2(n)$$

$$v_2(n + 1) = x\,(n - 1) = v_3(n)$$

$$v_3(n + 1) = x\,(n)$$

and the output equation is

$$y(n) = a_0x(n) + a_1x(n-1) + a_2x(n-2) + a_3x(n-3)$$

Substituting the states in the above output equation gives

$$y(n) = a_0x(n) + a_1v_3(n) + a_2v_2(n) + a_3v_1(n)$$

The state matrix equations are then

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \mathbf{x}$$

$$\mathbf{y}(n) = \begin{pmatrix} a_3 & a_2 & a_1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{pmatrix} + a_0\mathbf{x}(n)$$

where in this case

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} a_3 & a_2 & a_1 \end{pmatrix} \quad \mathbf{D} = \begin{pmatrix} a_0 \end{pmatrix}$$

5.3.3 From the Block Diagram to State-Space

Consider the block diagram shown in Figure 5.1. To obtain the states from the block diagram, we need first to determine the order of the system so that we know how many states to consider. Then we will let the output of every delay represent a state. Since the system is second order, we will have two states $v_1(n)$ and $v_2(n)$. Let the output of the first delay from the input side be denoted as $v_1(n)$. Then the input of this delay is $v_1(n+1)$. Therefore, from the graph we have

$$v_1(n+1) = x(n) + a_0y(n)$$

Let the output of the second delay be $v_2(n)$. Then

$$v_1(n+1) = x(n) + a_0\left[b_0x(n) + v_2(n)\right]$$

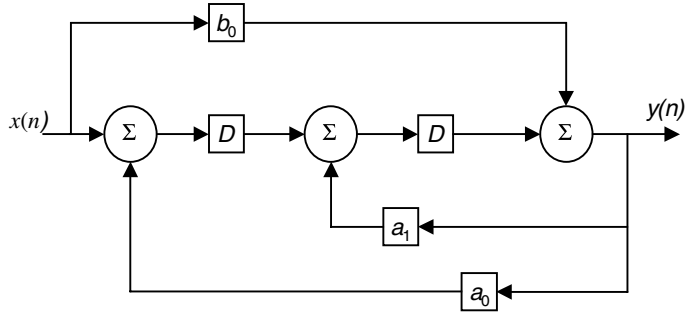

FIGURE 5.1 Block diagram representation 1.

or

$$v_1(n+1) = x(n)\left[1 + a_0 b_0\right] + a_0 v_2(n)$$

The input to the second delay is $v_2(n + 1)$ and is

$$v_2(n+1) = v_1(n) + a_1 y(n) = v_1(n) + a_1\left[b_0 x(n) + v_2(n)\right]$$
$$= v_1(n) + a_1 b_0 x(n) + a_1 v_2(n)$$

Also, the output $y(n)$ is

$$y(n) = b_0 x(n) + v_2(n)$$

Therefore, the state and the output matrix equations are

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \end{pmatrix} = \begin{pmatrix} 0 & a_0 \\ 1 & a_1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + b_0 \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + b_0 \mathbf{x}(n)$$

where

$$\mathbf{A} = \begin{pmatrix} 0 & a_0 \\ 1 & a_1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 + a_0 b_0 \\ a_1 b_0 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 0 & 1 \end{pmatrix} \quad \mathbf{D} = \begin{pmatrix} b_0 \end{pmatrix}$$

Consider another block diagram as shown in Figure 5.2. This system is third order and must have three states. Let the output of the first delay from the input side be $v_1(n)$, the output of the second delay be $v_2(n)$, and the output of the last delay be $v_3(n)$. The input of the first delay is

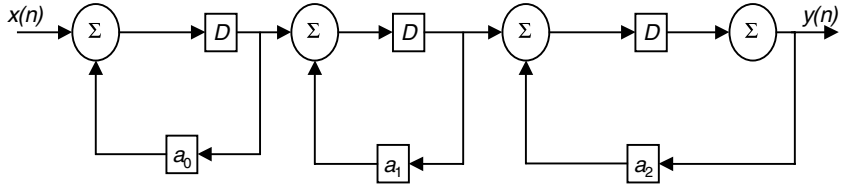

FIGURE 5.2 Block diagram representation 2.

$$v_1(n + 1) = x(n) + a_0\, v_1(n)$$

The input of the second delay is

$$v_2(n + 1) = v_1(n) + a_1\, v_2(n)$$

The input of the third delay is

$$v_3(n + 1) = v_2(n) + a_2\, v_3(n)$$

The output $y(n)$ is

$$y(n) = v_3(n)$$

We then have the state-space system in matrix form as

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \end{pmatrix} = \begin{pmatrix} a_0 & 0 & 0 \\ 1 & a_1 & 0 \\ 0 & 1 & a_2 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{pmatrix} + (0)\, \mathbf{x}(n)$$

where

$$\mathbf{A} = \begin{pmatrix} a_0 & 0 & 0 \\ 1 & a_1 & 0 \\ 0 & 1 & a_2 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \quad \mathbf{D} = (0)$$

5.3.4 From the Transfer Function $H(z)$ to State-Space

Given $H(z)$, the transfer function is the z-domain. We can obtain the state-space realization in many ways. We will illustrate that using an example.

Example 5.1

Consider the system

$$H(z) = \frac{z^2 - 4}{z^2 + 5z + 6}$$

What is the state-space representation?

Solution

1. Using direct realization, the block diagram for $H(z)$ is shown in Figure 5.3. Let the output of the first delay be $v_1(n)$. Then,

$$\begin{aligned} v_1(n+1) &= -4x(n) - 6y(n) = -4x(n) - 6\big[x(n) + v_2(n)\big] \\ &= -6\,v_2(n) - 10x(n) \end{aligned}$$

where $v_2(n)$ is the output of the second delay. Also,

$$\begin{aligned} v_2(n+1) &= v_1(n) - 5y(n) = v_1(n) - 5\big[x(n) + v_2(n)\big] \\ &= v_1(n) - 5v_2(n) - 5x(n) \end{aligned}$$

The output is

$$y(n) = x(n) + v_2(n)$$

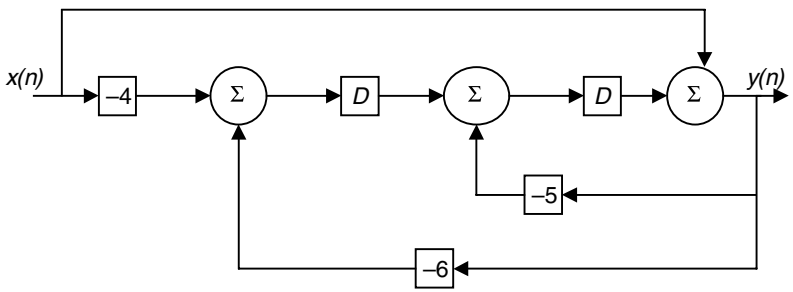

FIGURE 5.3 Block diagram for Example 5.1.

Then the state-space representation is

$$\begin{pmatrix} V_1(n+1) \\ V_2(n+1) \end{pmatrix} = \begin{pmatrix} 0 & -6 \\ 1 & -5 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + \begin{pmatrix} -10 \\ -5 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + 1\mathbf{x}(n)$$

with

$$\mathbf{A} = \begin{pmatrix} 0 & -6 \\ 1 & -5 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} -10 \\ -5 \end{pmatrix} \qquad \mathbf{C} = \begin{pmatrix} 0 & 1 \end{pmatrix} \qquad \mathbf{D} = (1)$$

2. The transfer function can also be written as

$$H(z) = \frac{z-1}{z^2+2} \frac{z+1}{z+3}$$

This is what we call the cascade system realization, and the block diagram can be drawn as in Figure 5.4. Again the system is still second order and if we let the output of the first delay be $v_1(n)$ and the output of the second delay be $v_2(n)$, then

$$\begin{aligned} v_1(n+1) &= -x(n) - 2x(n) - 2v_1(n) \\ v_2(n+1) &= x(n) + v_1(n) - 3y(n) \\ &= x(n) + v_1(n) - 3\left[x(n) + v_1(n) + v_2(n)\right] \end{aligned}$$

The output is

$$y(n) = x(n) + v_1(n) + v_2(n)$$

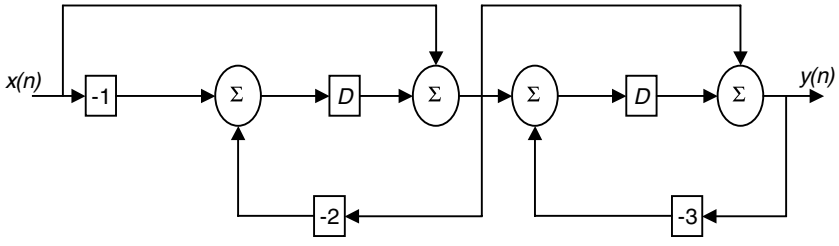

FIGURE 5.4 Block diagram for Example 5.1 in cascade form.

We can now clean the state equations above and write

$$v_1(n+1) = -2v_1(n) - 3x(n)$$

$$v_2(n+1) = -2v_1(n) - 3v_2(n) - 2x(n)$$

$$y(n) = v_1(n) + v_2(n) + x(n)$$

with

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \end{pmatrix} = \begin{pmatrix} -2 & 0 \\ -2 & -3 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + \begin{pmatrix} -3 \\ -2 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + (1)\,\mathbf{x}(n)$$

where

$$\mathbf{A} = \begin{pmatrix} -2 & 0 \\ -2 & -3 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} -3 \\ -2 \end{pmatrix} \quad \mathbf{C} = (1\ 1) \quad \mathbf{D} = (1)$$

3. We can also write $H(z)$ in a parallel realization form as

$$H(z) = \frac{3}{z+2} + \frac{-8}{z+3}$$

The block diagram is shown in Figure 5.5. Let $v_1(n)$ be the output of the upper delay and $v_2(n)$ be the output of the lower delay. Then

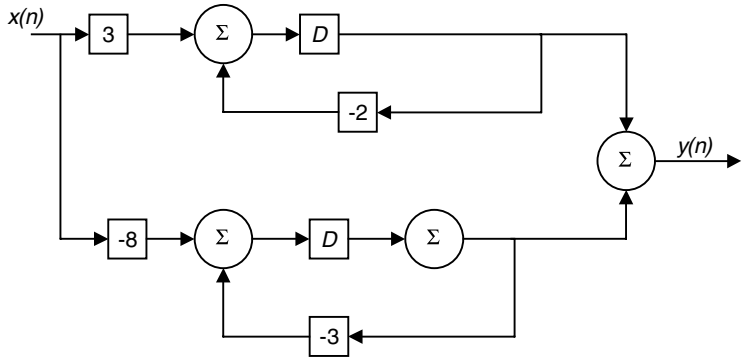

FIGURE 5.5 Block diagram for Example 5.1 in parallel form.

$$v_1(n + 1) = 3x(n) - 2v_1(n)$$

and

$$v_2(n + 1) = -8x(n) - 3v_2(n)$$

The output is

$$y(n) = v_1(n) + v_2(n)$$

The state-space system is then

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \end{pmatrix} = \begin{pmatrix} -2 & 0 \\ 0 & -3 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + \begin{pmatrix} 3 \\ -8 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y} = (1\ 1) \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + (0)\, \mathbf{x}(n)$$

with

$$\mathbf{A} = \begin{pmatrix} -2 & 0 \\ 0 & -3 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 3 \\ -8 \end{pmatrix} \quad \mathbf{C} = (1\ 1) \quad \mathbf{D} = (0)$$

4. Let us now draw a simulation diagram from

$$H(z) = \frac{z^2 - 1}{z^2 + 5z + 6}$$

that is different from the diagram we drew earlier in this example. From

$$H(z) = \frac{Y(z)}{X(z)}$$

we have

$$Y(z)\left[z^2 + 5z + 6\right] = X(z)\left[z^2 - 1\right]$$

By taking the inverse z-transform we get the difference equation

$$6y(n) + 5y(n + 1) + y(n + 2) = -x(n) + x(n + 2)$$

This is a second-order difference equation and its block diagram is shown in Figure 5.6. Let the output of the first delay on the top be $v_2(n)$ and the output of the lower delay be $v_1(n)$. Then we have the state and the output equations as

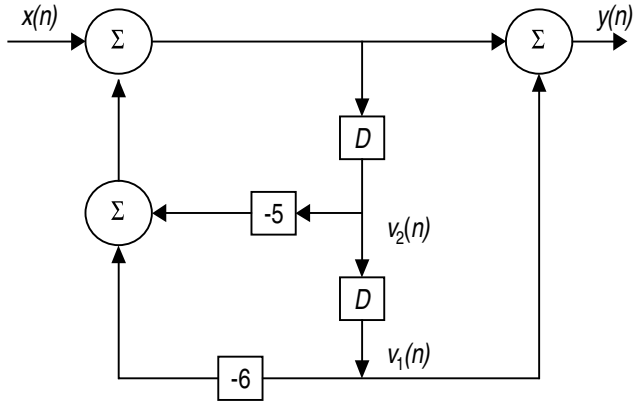

FIGURE 5.6 Block diagram for Example 5.1.

$$v_2(n+1) = x(n) - 5v_2(n) - 6v_1(n)$$

$$v_1(n+1) = v_2(n)$$

$$y(n) = v_1(n) + x(n) - 5\,v_2(n) - 6v_1(n)$$

Then the matrix state-space representation is

$$\begin{pmatrix} v_1(n+1) \\ v_1(n+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -6 & -5 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y} = (-5 \quad -5) = \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + (1)\mathbf{x}(n)$$

with

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -6 & -5 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} -5 & -5 \end{pmatrix} \quad \mathbf{D} = (1)$$

Notice that in all of these four state-space realizations we had different **A**, **B**, **C** and **D** matrices. To check that each system is a true realization, the eigenvalues in each case should be the same. In case 1

$$\mathbf{A} = \begin{pmatrix} 0 & -6 \\ 1 & -5 \end{pmatrix}$$

and the eigenvalues for the system are the roots of the determinant of ($\lambda\mathbf{I}$ **- A**).

$$\lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & -6 \\ 1 & -5 \end{pmatrix} = \begin{pmatrix} \lambda & 6 \\ -1 & \lambda + 5 \end{pmatrix}$$

$\det(\lambda\mathbf{I} - \mathbf{A}) = \lambda^2 + 5\lambda + 6$ and the eigenvalues are at –2 and –3.

In case 2

$$\mathbf{A} = \begin{pmatrix} -2 & 0 \\ -2 & -3 \end{pmatrix} \text{and } \det(\lambda\mathbf{I} - \mathbf{A}) = \det \begin{pmatrix} \lambda + 2 & 0 \\ 2 & \lambda + 3 \end{pmatrix} = \lambda^2 + 5\lambda + 6$$

The eigenvalues are at –2 and –3.

In case 3

$$\mathbf{A} = \begin{pmatrix} -2 & 0 \\ 0 & -3 \end{pmatrix} \text{and } \det(\lambda\mathbf{I} - \mathbf{A}) = \det \begin{pmatrix} \lambda + 2 & 0 \\ 0 & \lambda + 3 \end{pmatrix} = \lambda^2 + 5\lambda + 6$$

The eigenvalues are at –2 and –3.

In the last case

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -6 & -5 \end{pmatrix} \text{and } \det(\lambda\mathbf{I} - \mathbf{A}) = \lambda^2 + 5\lambda + 6$$

The eigenvalues are at –2 and –3.

Example 5.2

Consider the system

$$\frac{b_0 z^5 + b_1 z^4 + b_2 z^3 + b_3 z^2 + b_4 z + b_5}{z^5 + a_1 z^4 + a_2 z^3 + a_3 z^2 + a_4 z + a^5}$$

with the coefficient of z^5 in the denominator always unity. What is the state-space representation?

Solution

We start by the block diagram shown in Figure 5.7. From the graph we see that the state equations are

$$v_1(n+1) = v_2(n)$$

$$v_2(n+1) = v_3(n)$$

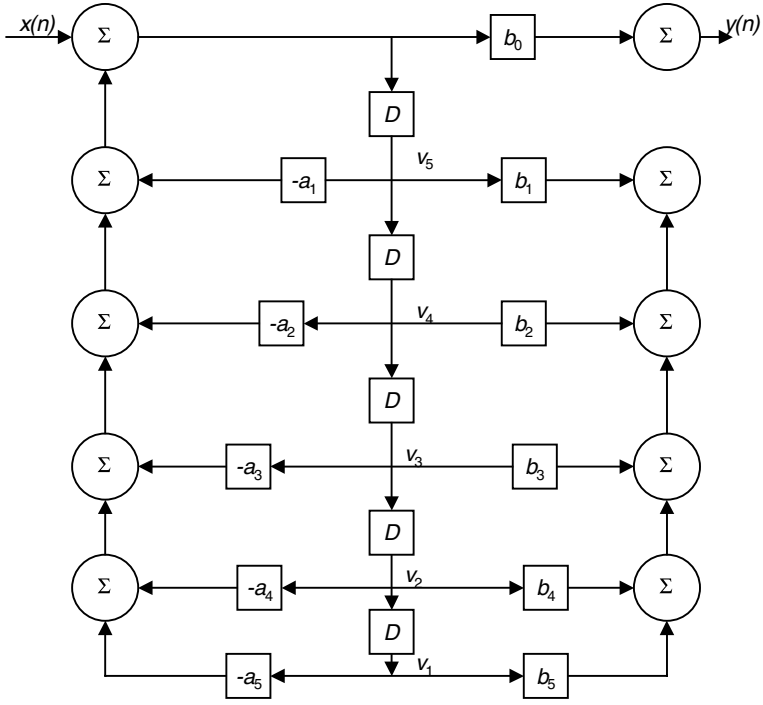

FIGURE 5.7 Block diagram for Example 5.2.

$$v_3(n+1) = v_4(n)$$

$$v_4(n+1) = v_5(n)$$

$$v_5(n+1) = x(n) - a_1 v_5(n) - a_2 v_4(n) - a_3 v_3(n) - a_4 v_2(n) - a_5 v_1(n)$$

and the output equation is

$$y(n) = (b_5 - a_5 b_0) v_1(n) + (b_4 - a_4 b_0) v_2(n) + (b_3 - a_3 b_0) v_3(n)$$
$$+ (b_2 - a_2 b_0) v_4(n) + (b_1 - a_1 b_0) v_5(n) + b_0 x(n)$$

The state equations and the output equation in state-space matrix form are

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+1) \\ v_3(n+1) \\ v_4(n+1) \\ v_5(n+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -a_5 & -a_4 & -a_3 & -a_2 & -a_1 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \\ v_4(n) \\ v_5(n) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{bmatrix} (b_5 - a_5 b_0) & (b_4 - a_4 b_0) & (b_3 - a_3 b_0) & (b_2 - a_2 b_0) & (b_1 - a_1 b_0) \end{bmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \\ v_4(n) \\ v_5(n) \end{pmatrix} + (b_0)\mathbf{x}(n)$$

Example 5.3

Given the system

$$H(z) = \frac{2z^3 + 3z^2 + 4z + 8}{z^3 + 5z^2 + 6z + 10}$$

What is the state space representation? Find **A**, **B**, **C** and **D**.

Solution

From Example 5.2 we can write these matrices by inspection and get

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -10 & -6 & -5 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} 8 - 10(2) & 4 - 6(2) & (3 - 5(2)) \end{pmatrix} = \begin{pmatrix} -12 & -8 & -7 \end{pmatrix} \quad \mathbf{D} = (2)$$

5.4 Solution of the State-Space Equations in the *z*-Domain

We are given the state-space system

$$\mathbf{v}(n+1) = \mathbf{A}\mathbf{v}(n) + \mathbf{B}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{v}(n) + \mathbf{D}\mathbf{x}(n)$$

By taking the *z*-transform on the above state equations we will get

$$z\mathbf{V}(z) - z\mathbf{v}(0) = \mathbf{A}\mathbf{V}(z) + \mathbf{B}\mathbf{X}(z)$$

Notice that z is a scalar and cannot be subtracted from **A**. Therefore we write the above equations as

$$(z\mathbf{I}-\mathbf{A})\mathbf{V}(z)=z\mathbf{v}(0)+\mathbf{B}\mathbf{X}(z)$$

where **I** is the identity matrix. Solving the above equation for the states we get

$$\mathbf{V}(z)=z(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{v}(0)+(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z) \tag{5.11}$$

$\mathbf{v}(n)$ is the inverse transform of the above equation. For the output $\mathbf{y}(n)$ we have

$$\mathbf{Y}(z)=\mathbf{C}\mathbf{V}(z)+\mathbf{D}\mathbf{X}(z)=\mathbf{C}\left[z(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{v}(0)+(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z)\right]+\mathbf{D}\mathbf{X}(z)$$

This is a good place to try to find $\mathbf{H}(z)$ from the state-space equations. With $\mathbf{v}(0) = 0$, we have

$$\mathbf{Y}(z)=\mathbf{C}\mathbf{V}(z)+\mathbf{D}\mathbf{X}(z)=\left[\mathbf{C}(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}+\mathbf{D}\right]\mathbf{X}(z)$$

or

$$\frac{\mathbf{Y}(z)}{\mathbf{X}(z)}=\left[\mathbf{C}(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}+\mathbf{D}\right] \tag{5.12}$$

5.5 General Solution of the State Equation in Real-Time

The state equation in matrix form is repeated here

$$\mathbf{v}(n+1)=\mathbf{A}\mathbf{v}(n)+\mathbf{B}\mathbf{x}(n)$$

Assume that the initial condition vector $\mathbf{v}(0)$ is known for $n = 0$. For $n = 0$, the state equation becomes

$$\mathbf{v}(1)=\mathbf{A}\mathbf{v}(0)+\mathbf{B}\mathbf{x}(0)$$

For $n = 1$ the state equation is

$$\mathbf{v}(2)=\mathbf{A}\mathbf{v}(1)+\mathbf{B}\mathbf{x}(1)$$

If we substitute $\mathbf{v}(1)$ in the equation for $\mathbf{v}(2)$ we get

$$\mathbf{A}\left[\mathbf{A}\mathbf{v}(0)+\mathbf{B}\mathbf{x}(0)\right]+\mathbf{B}\mathbf{x}(1)=\mathbf{A}^2\mathbf{v}(0)+\mathbf{A}\mathbf{B}\mathbf{x}(0)+\mathbf{B}\mathbf{x}(1)$$

For $n = 2$, the state equation is

$$\mathbf{v}(3) = \mathbf{A}\mathbf{v}(2) + \mathbf{B}\mathbf{x}(2) = \mathbf{A}\left[\mathbf{A}^2\mathbf{v}(0) + \mathbf{A}\mathbf{B}\mathbf{x}(0) + \mathbf{B}\mathbf{x}(1)\right] + \mathbf{B}\mathbf{x}(2)$$

$$= \mathbf{A}^3\mathbf{v}(0) + \mathbf{A}^2\mathbf{B}\mathbf{x}(0) + \mathbf{A}\mathbf{B}\mathbf{x}(1) + \mathbf{B}\mathbf{x}(2)$$

From the above we deduce that for $n \geq 0$ the states are given by

$$\mathbf{v}(n) = \mathbf{A}^n\mathbf{v}(0) + \sum_{k=0}^{n-1} \mathbf{A}^{(n-1-k)}\mathbf{B}\mathbf{x}(k) \tag{5.13}$$

Let $\mathbf{A}^n = \varphi(n)$, the state transition matrix. Then the solution for $\mathbf{v}(n)$ becomes

$$\mathbf{v}(n) = \boldsymbol{\varphi}(n)\mathbf{v}(0) + \sum_{k=0}^{n-1} \varphi(n-1-k)\mathbf{B}\mathbf{x}(k) \tag{5.14}$$

$$\mathbf{y}(n) = \mathbf{C}\boldsymbol{\varphi}(n)\mathbf{v}(0) + \sum_{k=0}^{n-1} \mathbf{C}\boldsymbol{\varphi}(n-1-k)\mathbf{B}\mathbf{x}(k) + \mathbf{D}\mathbf{x}(n) \tag{5.15}$$

In general we do not attempt to solve for $\mathbf{A}^n$ as a function of n. But we use recursion to solve for $\mathbf{v}(n)$.

Now let us compare the solutions for the state vector using the time-domain given by

$$\mathbf{v}(n) = \boldsymbol{\varphi}(n)\mathbf{v}(0) + \sum_{k=0}^{n-1} \boldsymbol{\varphi}(n-1-k)\mathbf{B}\mathbf{x}(k)$$

and the solution using the z-domain given as

$$\mathbf{V}(z) = z(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{v}(0) + (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z)$$

A close look at these solutions reveals that the transition matrix is found by comparing the coefficients of $\mathbf{v}(0)$ and it is the inverse transform of

$$z(z\mathbf{I} - \mathbf{A})^{-1} \tag{5.16}$$

5.6 Properties of $\mathbf{A}^n$ and Its Evaluation

In real-time we will use the computer to solve for the states $\mathbf{v}(n)$. We can also use the z-transform method to find a closed-form solution for the states if the system order is reasonably low. The transition matrix $\mathbf{A}''$has some interesting properties.

1. The response to a system with zero input is

$$\mathbf{v}(n) = \varphi(n)\mathbf{v}(0)$$

For $n = 0$, $\mathbf{v}(0) = \varphi(0)\mathbf{v}(0)$. This indicates that $\varphi(0) = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. This result is useful if we want to confirm the correctness of $\varphi(n)$.

2. With $\varphi(n) = \mathbf{A}^n$, let $n = n_1 + n_2$ to write

$$\boldsymbol{\varphi}(n_1 + n_2) = \mathbf{A}^{n_1+n_2} = \mathbf{A}^{n_1}\mathbf{A}^{n_2} = \boldsymbol{\varphi}(n_1)\boldsymbol{\varphi}(n$$

3. With $\varphi(n) = \mathbf{A}^n$, let $n = -m$ to get

$$\boldsymbol{\varphi}(-m) = \mathbf{A}^{-m} = (\mathbf{A}^m)^{-1} = \boldsymbol{\varphi}^{-1}(m$$

Thus we can write

$$\boldsymbol{\varphi}^{-1}(n) = \boldsymbol{\varphi}(-n)$$

Example 5.4

Consider the system

$$\mathbf{v}(n+1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 1 \end{pmatrix}\mathbf{v}(n)$$

Find the states and the output, $\mathbf{v}(n)$ and $\mathbf{y}(n)$.

Let $\mathbf{v}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ with $\mathbf{x}(n) = 0$

Solution

In real-time we can use recursion to solve $\mathbf{v}(n)$ and $\mathbf{y}(n)$. The state solution is

$$\mathbf{v}(n) = \mathbf{A}^n\mathbf{v}(0) + \sum_{k=0}^{n-1} \mathbf{A}^{(n-1-k)}\mathbf{B}\mathbf{x}(k)$$

But with $\mathbf{x}(n) = 0$,

$$\mathbf{v}(n) = \mathbf{A}^n\mathbf{v}(0)$$

Now we start the iteration and write

$$\mathbf{v}(1) = \mathbf{A}\mathbf{v}(0) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -2 \end{pmatrix}$$

$$\mathbf{v}(2) = \mathbf{A}\mathbf{v}(1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 0 \\ -2 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \end{pmatrix}$$

$$\mathbf{v}(3) = \mathbf{A}\mathbf{v}(2) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 0 \\ 4 \end{pmatrix} = \begin{pmatrix} 0 \\ -8 \end{pmatrix}$$

and so on. But since **A** is in a diagonal form we have

$$\mathbf{A}^n = \begin{pmatrix} (-1)^n & 0 \\ 0 & (-2)^n \end{pmatrix}$$

and therefore the state solution is

$$\mathbf{v}(n) = \mathbf{A}^n \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Thus

$$\mathbf{v}(0) = \mathbf{A}^0 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\mathbf{v}(1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -2 \end{pmatrix}$$

$$\mathbf{v}(2) = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \end{pmatrix}$$

$$\mathbf{v}(3) = \begin{pmatrix} -1 & 0 \\ 0 & -8 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -8 \end{pmatrix}$$

Finally, the closed form solution is

$$\mathbf{v}(n) = \begin{pmatrix} (-1)^n & 0 \\ 0 & (-2)^n \end{pmatrix}\mathbf{v}(0) \qquad n \geq 0$$

Using the z-transform, we have

$$\mathbf{V}(z) = z(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{v}(0) + (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z)$$

But again with x(n) = 0 we have

$$(z\mathbf{I} - \mathbf{A}) = \begin{pmatrix} z & 0 \\ 0 & z \end{pmatrix} - \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix} = \begin{pmatrix} z+1 & 0 \\ 0 & z+2 \end{pmatrix}$$

$$(z\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{z+2}{(z+1)(z+2)} & \dfrac{0}{(z+1)(z+2)} \\ \dfrac{0}{(z+1)(z+2)} & \dfrac{z+1}{(z+1)(z+2)} \end{pmatrix} = \begin{pmatrix} \dfrac{1}{z+1} & 0 \\ 0 & \dfrac{1}{z+2} \end{pmatrix}$$

Now we substitute this matrix inversion in the states solution to get

$$\mathbf{V}(z) = z\begin{pmatrix} \dfrac{1}{z+1} & 0 \\ 0 & \dfrac{1}{z+2} \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \dfrac{z}{z+2} \end{pmatrix}$$

and the inverse of $\mathbf{V}(z)$ is $\mathbf{v}(n)$ which is

$$\mathbf{v}(n) = \begin{pmatrix} 0 \\ (-2)^n u(n) \end{pmatrix}$$

For comparison we find the first few values next.

$$\mathbf{v}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \mathbf{v}(1) = \begin{pmatrix} 0 \\ -2 \end{pmatrix} \quad \mathbf{v}(2) = \begin{pmatrix} 0 \\ 4 \end{pmatrix} \quad \mathbf{v}(3) = \begin{pmatrix} 0 \\ -8 \end{pmatrix}$$

We can also see that the inverse z-transform of

$$z(z\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{z}{z+1} & 0 \\ 0 & \dfrac{z}{z+2} \end{pmatrix}$$

is

$$\boldsymbol{\varphi}(n) = \begin{pmatrix} (-1)^n u(n) & 0 \\ 0 & (-z)^n u(n) \end{pmatrix}$$

as obtained earlier in real-time. You can also see here that the transition matrix at $n = 0$ is

$$\boldsymbol{\varphi}(0) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

which is the identity matrix.

5.7 Transformations for State-Space Representations

Let us define another state vector called $\mathbf{w}(n)$ such that

$$\mathbf{w}(n) = \mathbf{P}\mathbf{v}(n) \tag{5.17}$$

where $\mathbf{P}$ is called the transformation matrix. With $\mathbf{w}(n) = \mathbf{P}\mathbf{v}(n)$ we have

$$\mathbf{v}(n) = \mathbf{P}^{-1}\mathbf{w}(n) \tag{5.18}$$

But we know that the original state equation is

$$\mathbf{v}(n+1) = \mathbf{A}\mathbf{v}(n) + \mathbf{B}\mathbf{x}(n)$$

and using the new transformation we have

$$\mathbf{v}(n+1) = \mathbf{P}^{-1}\mathbf{w}(n+1) \tag{5.19}$$

Then, by substituting in the original state equation we get

$$\mathbf{P}^{-1}\mathbf{w}(n+1) = \mathbf{A}\mathbf{P}^{-1}\mathbf{w}(n) + \mathbf{B}\mathbf{x}(n) \tag{5.20}$$

Multiply both sides in the equation above by $\mathbf{P}$ to get

$$\mathbf{w}(n+1) = \mathbf{P}\mathbf{A}\mathbf{P}^{-1}\mathbf{w}(n) + \mathbf{P}\mathbf{B}\mathbf{x}(n) \tag{5.21}$$

and for the output we have

$$\mathbf{y}(n) = \mathbf{C}\mathbf{P}^{-1}\mathbf{w}(n) + (\mathbf{D})\mathbf{x}(n) \tag{5.22}$$

These are the new state and output equations. The transfer function of the original system is

$$\mathbf{H}_{\text{old}}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})\ \mathbf{B} + \mathbf{D}$$

For the new system, the transfer function is

$$\mathbf{H}_{\text{new}}(z) = \mathbf{C}\mathbf{P}^{-1}(z\mathbf{I} - \mathbf{P}\mathbf{A}\mathbf{P}^{-1})^{-1}\mathbf{P}\mathbf{B} + \mathbf{D} \tag{5.23}$$

$$\mathbf{H}_{\text{new}}(z) = \mathbf{C}\mathbf{P}^{-1}\left(z\mathbf{P}\mathbf{P}^{-1} - \mathbf{P}\mathbf{A}\mathbf{P}^{-1}\right)^{-1}\mathbf{P}\mathbf{B} + \mathbf{D} = \mathbf{C}\mathbf{P}^{-1}\left[\mathbf{P}(z\mathbf{I} - \mathbf{A})\mathbf{P}^{-1}\right]^{-1}\mathbf{P}\mathbf{B} + \mathbf{D}$$

Knowing that

$$\mathbf{I} = \mathbf{P}\mathbf{P}^{-1} \text{ and } (\mathbf{LMN})^{-1} = \mathbf{N}^{-1}\ \mathbf{M}^{-1}\ \mathbf{L}^{-1}$$

we write the new transfer function as

$$\mathbf{H}_{\text{new}}(z) = \mathbf{C}\mathbf{P}^{-1}\mathbf{P}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}^{-1}\mathbf{P}\mathbf{B} + \mathbf{D} = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \tag{5.24}$$

You can see clearly now that

$$\mathbf{H}_{\text{old}}(z) = \mathbf{H}_{\text{new}}(z)$$

This also means that the eigenvalues are the same for the new transformed system.

Assume that the old system has the initial condition vector $\mathbf{v}(0)$. Then with $\mathbf{w}(n) = \mathbf{P}\mathbf{v}(n)$,

$$\mathbf{w}(0) = \mathbf{P}\mathbf{v}(0) \tag{5.25}$$

and

$$\mathbf{P}^{-1}\mathbf{w}(0) = \mathbf{v}(0) \tag{5.26}$$

The solution of the old system in z-domain for the states and the outputs are

$$\mathbf{V}(z) = z(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{v}(0) + (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z) \tag{5.27}$$

$$\mathbf{Y}(z) = \mathbf{C}\mathbf{V}(z) + \mathbf{D}\mathbf{X}(z) \tag{5.28}$$

and the solutions of the new system in the z-domain are

$$\mathbf{W}(z) = z\left(z\mathbf{I} - \mathbf{P}\mathbf{A}\mathbf{P}^{-1}\right)^{-1}\mathbf{w}(0) + \left(z\mathbf{I} - \mathbf{P}\mathbf{A}\mathbf{P}^{-1}\right)^{-1}\mathbf{P}\mathbf{B}\mathbf{X}(z) \tag{5.29}$$

$$\mathbf{Y}(z) = \mathbf{C}\mathbf{P}^{-1}\mathbf{W}(z) + \mathbf{D}\mathbf{X}(z) \tag{5.30}$$

If the matrix **P** is the matrix that contains the eigenvectors of **A**, then $\mathbf{PAP}^{-1}$ is a diagonal or a block diagonal matrix depending of the eigenvalues of **A** as we discussed at the beginning of the chapter. When the matrix **A** is transformed into a diagonal matrix where the eigenvalues are located on the main diagonal, the state equations are decoupled and can be solved one by one easily.

5.8 Some Insights: Poles and Stability

The objective of this chapter is to represent linear systems in state-space form and to look for ways of solving for the states. In that, the process was to represent an n^{th} order system (n^{th} order difference equation) as n first-order difference equations and arrange these equations in what we call state-space representation as

$$\mathbf{v}(n+1) = \mathbf{A}\mathbf{v}(n) + \mathbf{B}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{v}(n) + \mathbf{D}\mathbf{x}(n)$$

where **x** is the input vector (assuming multiple inputs) and **y** is the output vector (assuming multiple outputs). The **v** vector is the vector that contains the states of the system. The **A** matrix is the matrix that contains the parameters that control the dynamics of the system. As we saw in previous chapters, in every system representation there was a way to find the eigenvalues of the system. In state-space representation, the roots of the determinant of the matrix, ($s\mathbf{I} - \mathbf{A}$), where **I** is the identity matrix, are the eigenvalues of the system, the poles. And as we mentioned before, these poles determine the shape of the transients of the system under investigation.

Consider the case where the dynamics matrix **A** is

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -6 & -5 \end{pmatrix}$$

$$(\lambda\mathbf{I} - \mathbf{A}) = \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ -0 & -5 \end{pmatrix} = \begin{pmatrix} \lambda & -1 \\ 6 & \lambda + 5 \end{pmatrix}$$

The eigenvalues are the roots of the determinant of ($\lambda\mathbf{I} - \mathbf{A}$). They are the roots of $\lambda^2 + 5\lambda + 6 = 0$. The eigenvalues are at –3 and –2. Thus we expect a solution that will contain the terms $c_1(-3)^n + c_2(-2)^n$. The stability of this system depends entirely on the eigenvalues, not on the constants c_1 and c_2. These eigenvalues are the roots of ($\lambda\mathbf{I} - \mathbf{A}$). They are also the roots of the characteristic equation derived from the difference equation and also the roots of the denominator in the transfer function representing the system in the z-domain.

To summarize, if the system is given in state-space form, the stability of the system is determined by finding the roots of $\det(\lambda\mathbf{I} - \mathbf{A})$. If all the roots are within the unit circle, the system is stable. If one of the roots is not, the system is unstable. And again, the roots will determine the shape of the transients.

5.9 End of Chapter Examples

EOCE 5.1

Consider the following matrices

$$\mathbf{A} = \begin{pmatrix} -2 & 0 \\ 0 & -3 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 1 & 2 \end{pmatrix} \quad \mathbf{D} = (5)$$

Find the following:

1. $\mathbf{AB} - \mathbf{B}$ and $(\mathbf{AB})^T + \mathbf{C}$
2. $\mathbf{A}^{-1}$ and $\mathbf{AA}^{-1}$
3. The eigenvalues for $\mathbf{A}$ and $\mathbf{A}^2$
4. $\mathbf{CDB} + \mathbf{D}$
5. $\mathbf{BC} - \mathbf{A}$

Solution

We will use MATLAB to find the answers to the above questions. The MATLAB script is

```
A=[-2 0;0 -3]; B=[0;1]; C=[1 2]; D=[5];
ABminusB=A*B-B
ABtransplusC=(A*B)' + C
invA=inv(A)
AinvA=A*inv(A)
eigA=eig(A)
eigAA=eig(A*A)
CDBplusD=C*D*B + D
BCminusA=B*C-A
```

The result is

```
ABminusB =
     0
    -4
```

```
ABtransplusC =
    –1   –1
     2

invA =
    –0.5000       0
          0   –0.3333

AinvA =
     1    0
     0    1

eigA =
    –3
    –2

eigAA =
     4
     9

CDBplusD =
    15

BCminusA =
     2    0
     1    5
```

EOCE 5.2

Consider the following matrices

$$\mathbf{A} = \begin{pmatrix} 2 & 9 \\ 0 & 1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Find

1. Eigenvalues for **A** and **B**
2. $\mathbf{A}^{-1}$ and $\mathbf{B}^{-1}$
3. $(\mathbf{AB})^{-1}$ and $\mathbf{B}^{-1}\,\mathbf{A}^{-1}$
4. **BCBC** and $\mathbf{CA}^{-1}$
5. $(\mathbf{CA})^{-1}$ and $(\mathbf{BC})^{-1}\,\mathbf{A}$

Solution

MATLAB is used again here. The script is

```
A=[2 9;0 1]; B=[1 2;0 1]; C=[1 0;0 1];
Ainv=inv(A)
Binv=inv(B)
ABinv=inv(A*B)
BinvAinv=inv(B)*inv(A)
BCBC=B*C*B*C
CinvA=C*inv(A)
CAinv=inv(C*A)
BCinvA=inv(B*C)*A
```

The result is

```
Ainv =
    0.5000   –4.5000
         0    1.0000

Binv =
     1   –2
     0    1

ABinv =
    0.5000   –6.5000
         0    1.0000

BinvAinv =
    0.5000   –6.5000
         0    1.0000

BCBC =
     1    4
     0    1

CinvA =
    0.5000   –4.5000
         0    1.0000

CAinv =
    0.5000   –4.5000
         0    1.0000
```

```
BCinvA =
    2    7
    0    1
```

EOCE 5.3

Consider the matrices

$$\mathbf{A}=\begin{pmatrix}-1 & 0\\ 0 & -2\end{pmatrix}\ \mathbf{B}=\begin{pmatrix}-3 & 0\\ 0 & 4\end{pmatrix}\ \mathbf{C}=\begin{pmatrix}1 & 1\\ 0 & 2\end{pmatrix}$$

1. Find eigenvalues and eigenvectors for **A** and **B**.
2. Find eigenvalues and eigenvectors for $\mathbf{A}^2$ and $\mathbf{B}^2$.
3. Put **C** in the diagonal form by first calculating **P**, the matrix of the eigenvector for **C**.
4. What are the eigenvectors and eigenvalues of $\mathbf{P}^{-1}\mathbf{CP}$?

Solution

We will use MATLAB again. The MATLAB command

```
[V,D] = eig(A,B)
```

produces a diagonal matrix `D` of eigenvalues usually on the main diagonal and a matrix `V` with eigenvector columns. The script is

```
A=[-1 0;0 -2]; B=[-3 0;0 4]; C=[1 1;0 2];
[Aeigvectors Aeigenvalues]=eig(A)
[Beigvectors Beigenvalues]=eig(B)
[AAeigvectors AAeigenvalues]=eig(A*A)
[BBeigvectors BBeigenvalues]=eig(B*B)
[Pofeigenvectors Cindiagonalform]=eig(C)
[PinvCPeigvectors PinvCPeigenvalues]=eig(inv
 (Pofeigenvectors)*C*P)
```

The result is

```
Aeigvectors =
     0    1
     1    0

Aeigenvalues =
    -2    0
     0   -1
```

```
Beigvectors =
     1     0
     0     1

Beigenvalues =
    –3     0
     0     4

AAeigvectors =
     1     0
     0     1

AAeigenvalues =
     1     0
     0     4

BBeigvectors =
     1     0
     0     1

BBeigenvalues =
     9     0
     0    16

Pofeigenvectors =
    1.0000    0.7071
         0    0.7071

Cindiagonalform =
     1     0
     0     2

PinvCPeigvectors =
     1     0
     0     1

PinvCPeigenvalues =
     1     0
     0     2
```

EOCE 5.4

Put the following system in state-space form.

$$y(n) - 2y(n-1) + 3y(n-3) = 4x(n)$$

Solution

Let

$$v_1(n) = y(n-3)$$

$$v_2(n) = y(n-2)$$

$$v_3(n) = y(n-1)$$

We have three states since our system is third order. Thus we have

$$v_1(n+1) = y(n-2) = v_2(n)$$

$$v_2(n+1) = y(n-1) = v_3(n)$$

$$v_3(n+1) = 2y(n-1) - 3y(n-3) + 4x(n) = 2v_3(n) - 3v_1(n) + 4x(n)$$

The output $y(n)$ is

$$y(n) = 2v_3(n) - 3v_1(n) + 4x(n)$$

and the state and output equations are

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -3 & 0 & 2 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 0 \\ 0 \\ 4 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} -3 & 0 & 2 \end{pmatrix} \mathbf{v}(n) + (4)\mathbf{x}(n)$$

EOCE 5.5

Consider the system

$$H(z) = \frac{2z+1}{z^3 + 3z^2 + 2z + 8}$$

Write the state and output equations for this system.

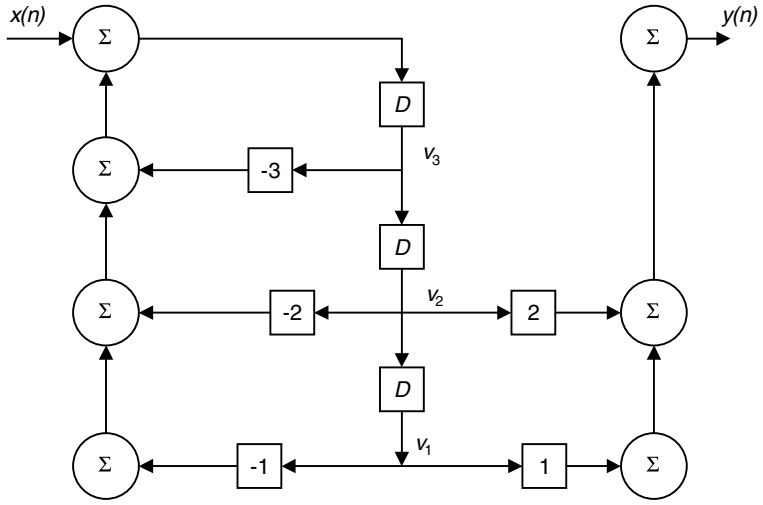

FIGURE 5.8 Block diagram for EOCE 5.5.

Solution

We can deduce the difference equation from $H(z)$ first, then find the state equations.

From the transfer function we can write

$$Y(z)\left[z^3 + 3z^2 + 2z + 1\right] = [2z + 1]X(z)$$

By taking the inverse transform we get

$$y(n) + 2y(n+1) + 3y(n+2) + y(n+3) = x(n) + 2x(n+1)$$

Let us draw the block diagram first. The block diagram is shown in Figure 5.8. From the figure we see that the states are

$$v_3(n+1) = x(n) - 3v_3(n) - 2v_2(n) - v_1(n)$$
$$v_2(n+1) = v_3(n)$$
$$v_1(n+1) = v_2(n)$$

and the output is

$$y(n) = v_1(n) + 2v_2(n)$$

From the state and the output equations we can form the state-space matrix equations as

$$\mathbf{v}(n+1)=\begin{pmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -1 & -2 & -3\end{pmatrix}\mathbf{v}(n)+\begin{pmatrix}0\\ 0\\ 1\end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n)=\begin{pmatrix}1 & 2 & 0\end{pmatrix}\mathbf{v}(n)+(2)\mathbf{x}(n)$$

We can also write different state equations using Example 5.2. Referring to Example 5.2 we have

$$H(z)=\frac{b_0z^3+b_1z^2+b_2z+b_3}{z^3+a_1z^2+a_2z+a_3}=\frac{0z^3+0z^2+2z+1}{z^3+3z^2+2z+1}$$

in our present case. Then by inspection we have

$$\mathbf{v}(n+1)=\begin{pmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -1 & -2 & -3\end{pmatrix}\mathbf{v}(n)+\begin{pmatrix}0\\ 0\\ 1\end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n)=\begin{pmatrix}b_3-a_3(b_0) & b_2-a_2(b_0) & b_1-a_1(b_0)\end{pmatrix}\mathbf{v}(n)+b_0\mathbf{x}(n)$$

After substitution we arrive at

$$\mathbf{y}(n)=\begin{pmatrix}1 & 2 & 0\end{pmatrix}\mathbf{v}(n)+(0)\mathbf{x}(n)$$

EOCE 5.6

Consider the following system in Figure 5.9. Write the state equations in matrix form.

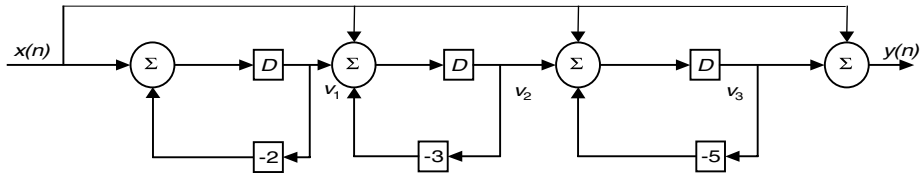

FIGURE 5.9 Block for EOCE 5.6.

Solution

Let the output of the first delay be $v_1(n)$, $v_2(n)$ for the second delay and $v_3(n)$ for the third delay. Then the state equations are

$$v_1(n+1) = x(n) - 2v_1(n)$$

$$v_2(n+1) = x(n) - 3v_2(n)$$

$$v_3(n+1) = x(n) - 5v_3(n)$$

The output equation is

$$y(n) = x(n) + v_3(n)$$

The state and output equations in state-space matrix form then are

$$\mathbf{v}(n+1) = \begin{pmatrix} -2 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -5 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \mathbf{v}(n) + \mathbf{x}(n)$$

EOCE 5.7

Consider the system

$$\mathbf{v}(n+1) = \begin{pmatrix} 1 & 0 \\ 1 & k \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 1 \end{pmatrix} \mathbf{v}(n)$$

1. For what values of k is the system stable?
2. For a value of k that stabilizes the system, find $H(z) = \dfrac{Y(z)}{X(z)}$.
3. Use recursion to solve for $\mathbf{v}(0)$, $\mathbf{v}(1)$, $\mathbf{v}(2)$, $\mathbf{v}(3)$ and $\mathbf{v}(4)$ with $\mathbf{x}(n) = u(n)$ and $\mathbf{v}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.
4. Use the z-transform to solve for $\mathbf{v}(n)$ for $n \geq 0$.
5. What is the transition matrix?

Solution

1. The eigenvalues of this system are the roots of the characteristic equation, which is the determinant of ($\lambda\mathbf{I} - \mathbf{A}$) set equal to zero.

$$(\lambda\mathbf{I} - \mathbf{A}) = \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 1 & k \end{pmatrix} = \begin{pmatrix} \lambda - 1 & 0 \\ -1 & \lambda - k \end{pmatrix}$$

$$\det(\lambda\mathbf{I} - \mathbf{A}) = (\lambda - 1)(\lambda - k)$$

   This system is stable if the eigenvalues of the system matrix **A** are all within the unit circle. The eigenvalues are at 1 and k. Therefore, for stability, the value of k has to be within the unit circle.

2. For $k = -\frac{1}{2}$, the state matrices are

$$\mathbf{A} = \begin{pmatrix} 1 & 0 \\ 1 & -\frac{1}{2} \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 0 & 1 \end{pmatrix} \quad \mathbf{D} = (0)$$

   The transfer function in this case is

$$\mathbf{H}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$$

$$\mathbf{H}(z) = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} z - 1 & 0 \\ -1 & z + \frac{1}{2} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\mathbf{H}(z) = \begin{pmatrix} 0 & 1 \end{pmatrix} \frac{1}{(z + \frac{1}{2})(z - 1)} \begin{pmatrix} z + \frac{1}{2} & 0 \\ 1 & z - 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\mathbf{H}(z) = \begin{pmatrix} \dfrac{1}{\left(z + \frac{1}{2}\right)(z - 1)} & \dfrac{z - 1}{\left(z + \frac{1}{2}\right)(z - 1)} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{z - 1}{\left(z + \frac{1}{2}\right)(z - 1)}$$

   We use MATLAB to find the transfer function $\mathbf{H}(z)$ from the matrices **A**, **B**, **C** and **D**. We will use the MATLAB function `ss2tf` (state-space to transfer function) to do that in the following script.

```
A=[1 0; 1 -1/2]; B=[0;1]; C=[0 1]; D=[0];
[num den]= ss2tf(A,B,C,D)
```

to get

```
num =
    0   1   –1

den =
    1.0000   –0.5000   –0.5000
```

and the transfer function is

$$\mathbf{H}(z) = \frac{z-1}{\left(z^2 - (1/2)z - 1/2\right)} = \frac{z-1}{\left(z+\frac{1}{2}\right)(z-1)}$$

3. To use recursion to solve for $\mathbf{v}(n)$ we will use the equation

$$\mathbf{v}(n+1) = \mathbf{A}\mathbf{v}(n) + \mathbf{B}\mathbf{x}(n)$$

and

$$\mathbf{y}(n) = \mathbf{C}\mathbf{v}(n) + \mathbf{D}\mathbf{x}(n)$$

We will have the initial sate as

$$\mathbf{v}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\mathbf{y}(0) = \begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = 1$$

The other few terms of the state vector and the output are given below.

$$\mathbf{v}(1) = \begin{pmatrix} 1 & 0 \\ 1 & -\frac{1}{2} \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}(1) = \begin{pmatrix} 0 \\ -\frac{1}{2} \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix}$$

$$\mathbf{y}(1) = \begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix} = \frac{1}{2}$$

$$\mathbf{v}(2) = \begin{pmatrix} 0 & 0 \\ 1 & -\frac{1}{2} \end{pmatrix}\begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}(1) = \begin{pmatrix} 0 \\ -\frac{1}{4} \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{3}{4} \end{pmatrix}$$

$$\mathbf{y}(2) = \begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ \frac{3}{4} \end{pmatrix} = \frac{3}{4}$$

and so on. We can use MATLAB to do this recursion. We will do that in the following script.

```
A=[1 0; 0 -1/2]; B=[0;1];C=[0 1];D=[0];
isv=[0;1];%initial state vector
for n=0:3 %only the first four values
  yn=C*isv
  isvplus1=A*isv + B*1; % 1 comes from unit step input
  isv=isvplus1
end
```

to get the same results as we had before.

4. We can also use the z-transform method to find the output and the state values.

$$\mathbf{V}(z) = z(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{v}(0) + (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z)$$

$$\mathbf{V}(z) = \frac{z}{\left(z+\frac{1}{2}\right)(z-1)}\begin{pmatrix} z+\frac{1}{2} & 0 \\ 1 & z-1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} + 5\frac{1}{\left(z+\frac{1}{2}\right)(z-1)}\begin{pmatrix} z+\frac{1}{2} & 0 \\ 1 & z-1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix}\frac{z}{z-1}$$

$$\mathbf{V}(z) = \begin{pmatrix} 0 \\ \frac{z(z-1)}{\left(z+\frac{1}{2}\right)(z-1)} \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{z-1}{\left(z+\frac{1}{2}\right)(z-1)} \end{pmatrix}\frac{z}{z-1}$$

After some simplifications we arrive at

$$\mathbf{V}(z) = \begin{pmatrix} 0 \\ \frac{z(z-1)+z}{\left(z+\frac{1}{2}\right)(z-1)} \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{z^2}{\left(z+\frac{1}{2}\right)(z-1)} \end{pmatrix}$$

and the output in the z-domain is given by

$$\mathbf{Y}(z) = \mathbf{C}\mathbf{V}(z) = (0\ \ 1)\mathbf{V}z = \frac{z^2}{\left(z+\frac{1}{2}\right)(z-1)}$$

The output is one-dimensional or scalar. We can do partial fraction expansion using MATLAB and write

```
num = [1 0 0];
den = [1 -1/2 -1/2];
[r p k]=residuez(num, den)
```

to get

```
r = 0.6667 0.3333
p = 1.0000 -0.5000
```

Therefore, with the help of MATLAB we arrive at

$$Y(z) = \frac{0.6667z}{z-1} + \frac{z(1/3)}{z+\frac{1}{2}}$$

and by taking the inverse transform we get

$$y(n) = v_2(n) = \left[0.6667(1)^n + 1/3\left(-\frac{1}{2}\right)^n\right]u(n)$$

If we substitute values for n we can verify the results we arrived at earlier. Try that.

5. The transition matrix $\boldsymbol{A}^n$ is the inverse transform of

$$z(z\mathbf{I}-\mathbf{A})^{-1} = \begin{pmatrix} \frac{z}{(z-1)} & 0 \\ \frac{z}{(z+1/2)(z-1)} & \frac{z}{(z+1/2)} \end{pmatrix}$$

$$z(z\mathbf{I}-\mathbf{A})^{-1} = \begin{pmatrix} \frac{z}{(z-1)} & 0 \\ \frac{0.6667}{(z-1)} + \frac{1/3}{z+1/2} & \frac{z}{(z+1/2)} \end{pmatrix}$$

By taking the inverse z-transform we find the transition matrix as

$$\mathbf{A}^n = \begin{pmatrix} (1)^n & 0 \\ 0.6667(1)^n + (1/3)(-1/2)^n & (-1/2)^n \end{pmatrix} u(n)$$

You can see that $\mathbf{A}^0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ as an indication that the transition matrix is correct.

We can also use MATLAB to verify the entries in the transition matrix $\mathbf{A}^n$. For the first entry we have $\boldsymbol{\varphi}_{11}(z) = \frac{z}{z-1}$. We will write the following MATLAB script to find $\boldsymbol{\varphi}_{11}(n)$.

```
syms Phill z %symbolic definitions
Phill = z/(z-1);
iztrans(Phill)
```

The result is $1u(n)$ as expected.

EOCE 5.8

Consider the system in Figure 5.10.

1. Write down the state and output equations.
2. What is the transition matrix?
3. With $\mathbf{x}(n) = \mathbf{0}$ and an initial state vector of $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$, what is the state vector and the output?
4. Is the system stable?
5. Use MATLAB to find $\mathbf{y}(n)$, for $\mathbf{x}(n) = u(n)$ and the initial state of $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$.
6. Find the transfer function $\mathbf{H}(z)$.

Solution

1. From Figure 5.10 and by taking $v_1(n)$ as the output of the upper delay and $v_2(n)$ as the output of the lower delay, we have the state equations

$$v_1(n+1) = x(n) - v_1(n)$$

$$v_2(n+1) = x(n) - 2v_2(n)$$

The output equation is

$$y(n) = v_1(n) + v_2(n)$$

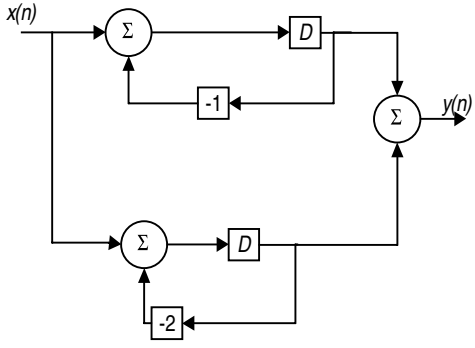

FIGURE 5.10 Block diagram for EOCE 5.8.

Then the state and output equations are grouped as

$$\mathbf{v}(n+1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\mathbf{v}(n) + \begin{pmatrix} 1 \\ 1 \end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 1 & 1 \end{pmatrix}\mathbf{v}(n) + (0)\mathbf{x}(n)$$

2. With $\mathbf{A} = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}$, the state transition matrix is the inverse transform of $z(z\mathbf{I} - \mathbf{A})^{-1}$. We have

$$(z\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} z+1 & 0 \\ 0 & z+2 \end{pmatrix}^{-1} = \begin{pmatrix} z+2 & 0 \\ 0 & z+1 \end{pmatrix}\frac{1}{(z+1)(z+2)}$$

and

$$z(z\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{z}{z+1} & 0 \\ 0 & \dfrac{z}{z+2} \end{pmatrix}$$

The transition matrix written is the inverse transform of $z(z\mathbf{I} - \mathbf{A})^{-1}$ and is

$$\mathbf{A}^n = \begin{pmatrix} (-1)^n u(n) & 0 \\ 0 & (-2)^n u(n) \end{pmatrix}$$

But since $\mathbf{A}$ is in diagonal form, we could have found $\mathbf{A}^n$ by inspection. With

$$\mathbf{A} = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}, \quad \mathbf{A}^n = \begin{pmatrix} (-1)^n u(n) & 0 \\ 0 & (-2)^n u(n) \end{pmatrix}$$

3. With $\mathbf{x}(n) = 0$, the state vector becomes

$$\mathbf{v}(n+1) = \mathbf{A}\mathbf{v}(n)$$

For $n = 0$ we have

$$\mathbf{v}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\mathbf{y}(0) = v_1(0) + v_2(0) = 1$$

For $n = 1$ we get

$$\mathbf{v}(1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$

$$\mathbf{y}(1) = v_1(1) + v_2(1) = -1 + 0 = -1$$

For $n = 2$ we have

$$\mathbf{v}(2) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\mathbf{y}(2) = v_1(2) + v_2(2) = 1 + 0 = 1$$

For $n = 3$ we get

$$\mathbf{v}(3) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$

$$\mathbf{y}(3) = v_1(3) + v_2(3) = -1 + 0 = -1$$

For $n = 4$ we get

$$\mathbf{v}(4) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} -1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\mathbf{y}(4) = v_1(4) + v_2(4) = 1 + 0 = 1$$

Then by induction we can see that

$$\mathbf{v}(n) = \begin{pmatrix} (-1)^n u(n) \\ 0 \end{pmatrix}$$

and

$$\mathbf{y}(n) = (-1)^n u(n)$$

Using MATLAB we can use recursion and write the following script to solve for the output and the state vector:

```
A=[-1 0; 0 -2]; B=[1;1];C=[1 1];D=[0];
isv=[1;0];%initial state vector
for n=0:4 %only the first five values
  yn=C*isv
  isvplus1=A*isv + B*0; % 0 comes from zero input
  isv=isvplus1
end
```

and the result will be identical to what we just found. We can also plot $y(n)$ vs. n by using the MATLAB function `dlsim` as in the following script:

```
A=[-1 0; 0 -2]; B=[1;1];C=[1 1];D=[0];
isv=[1;0];%initial state vector
n=0:10;
x=zeros(1,length(n));
[y,v]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear syatem
 simulation
stem(n,y); xlabel('n'); ylabel('Initial condition
 response');
```

The plots are shown in Figure 5.11.

4. The stability of the system depends on the location of the poles. One of the poles is not within the unit circle. Thus the system is not stable.
5. With x(n) = $u(n)$ and the initial conditions, $v_1(0) = 1$ and $v_2(0) = 0$, we can use MATLAB to find y(n) as in the following script:

```
A=[-1 0; 0 -2]; B=[1;1];C=[1 1];D=[0];
isv=[1;0];%initial state vector
n=0:10;
x=(1.^n);
[y,v]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear system
 simulation
stem(n,y); xlabel('n'); ylabel('The step response');
```

The plot is in Figure 5.12.

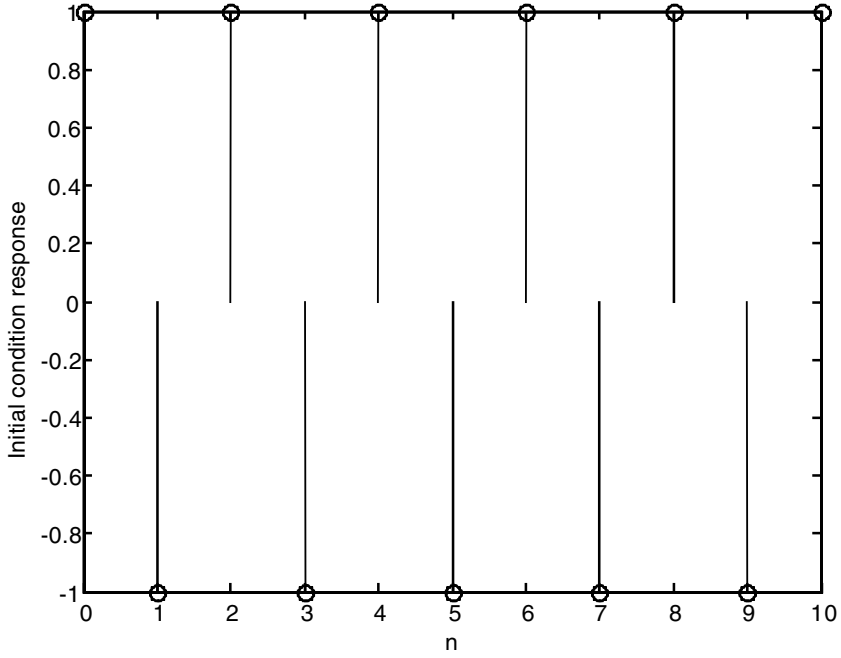

FIGURE 5.11 Plot for EOCE 5.8.

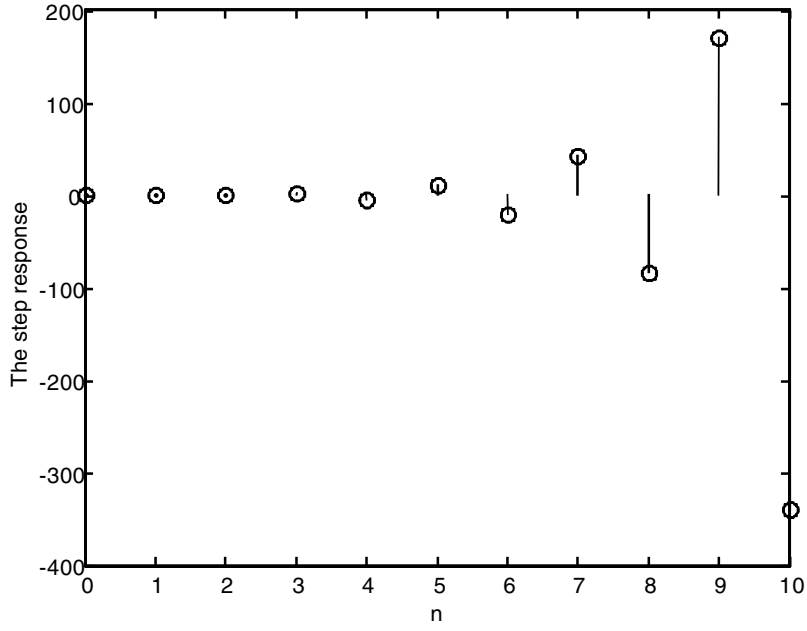

FIGURE 5.12 Plot for EOCE 5.8.

6. The transfer function $\mathbf{H}(z)$ is calculated using the equation

$$\mathbf{H}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$$

$$\mathbf{H}(z) = (1 \quad 1)\begin{pmatrix} \frac{1}{z+1} & 0 \\ 0 & \frac{1}{z+2} \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} + (0) = \begin{pmatrix} \frac{1}{z+1} & \frac{1}{z+2} \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Finally

$$\mathbf{H}(z) = \frac{1}{z+1} + \frac{1}{z+2} = \frac{z+z+2+1}{(z+1)(z+2)} = \frac{2z+3}{z^2+3z+2}$$

We can use MATLAB to verify this result as in the following script.

```
A=[-1 0; 0 -2]; B=[1;1];C=[1 1];D=[0];
[num den]=ss2tf(A,B,C,D)
```

to get

num = 0 2 3

den = 1 3 2

which verifies the result.

EOCE 5.9

We have seen in this chapter that the state-space representation is not unique. To illustrate that consider the following system

$$\mathbf{v}(n+1) = \mathbf{A}\mathbf{v}(n) + \mathbf{B}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{v}(n) + (\mathbf{D})\mathbf{x}(n)$$

with

$$\mathbf{A} = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \mathbf{C} = (0 \quad 1) \quad \mathbf{D} = (0)$$

What are the other state-space representations?

Solution

Let us define another state vector called $\mathbf{w}(n)$ such that $\mathbf{w}(n) = \mathbf{P}\mathbf{v}(n)$ where $\mathbf{P}$ is called the transformation matrix. Let

$$\mathbf{P} = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$$

Then the new system is

$$\mathbf{w}(n+1) = \mathbf{PAP}^{-1}\mathbf{w}(n) + \mathbf{PBx}(n)$$

and for the output we have

$$\mathbf{y}(n) = \mathbf{CP}^{-1}\mathbf{w}(n) + (\mathbf{D})\mathbf{x}(n)$$

Assume that the old system has the initial condition Vector $\mathbf{v}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Then with $\mathbf{w}(n) = \mathbf{Pv}(n)$, $\mathbf{w}(0) = \mathbf{Pv}(0)$ and $\mathbf{P}^{-1}\mathbf{w}(0) = \mathbf{v}(0)$. The solutions of the new system in the z-domain are

$$\mathbf{W}(z) = z\left(z\mathbf{I} - \mathbf{PAP}^{-1}\right)^{-1}\mathbf{w}(0) + \left(z\mathbf{I} - \mathbf{PAP}^{-1}\right)^{-1}\mathbf{PBX}(z)$$

and

$$\mathbf{Y}(z) = \mathbf{CP}^{-1}\mathbf{W}(z) + \mathbf{DX}(z)$$

Let us now use MATLAB to find the step response with zero initial conditions to the old system; we will then consider the new system and see that the two outputs are the same. The MATLAB script is next.

```
A=[0.5 0; 0 0.4]; B=[0;1];C=[0 1];D=[0];
n=0:10;
x=(1.^n);
isv=[0;0];% initial state vector
[yold v]=dlsim(A,B,C,D,x,isv);subplot(2,1,1); stem(n,yold);
title('The step response using the old system');
P=[ 1 0; 0 2];
Anew=P*A*inv(P);
Bnew=P*B;
Cnew=C*inv(P);
Dnew=D;
isvw=P*isv;
[ynew w]=dlsim(Anew, Bnew, Cnew, Dnew, x,isvw);
subplot(2,1,2); stem (n, ynew); xlabel('n');
title('The step response using the new system');
```

The plot s are shown in Figure 5.13.

We can also use MATLAB to check the stability and the eigenvalues for both systems as in the following script.

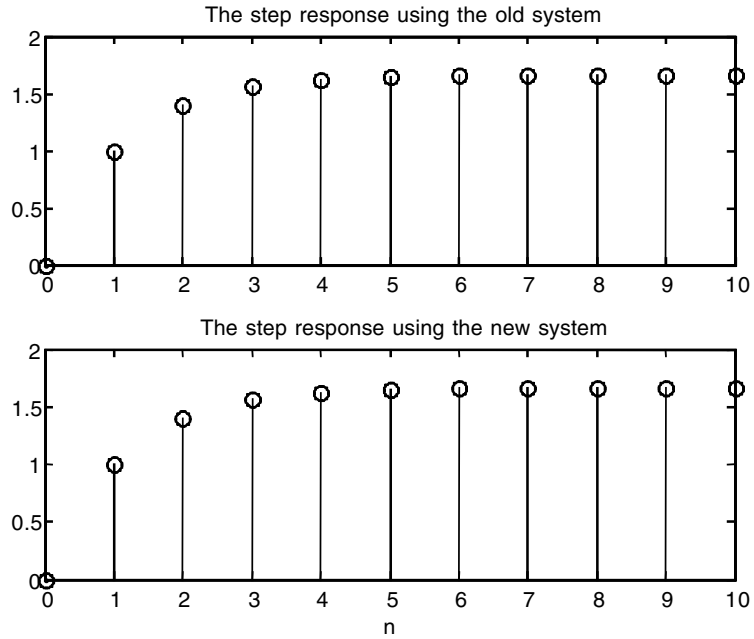

FIGURE 5.13 Plot for EOCE 5.9.

```
A=[0.5 0; 0 0.4]; B=[0;1];C=[0 1];D=[0];
P=[ 1 0; 0 2];
Anew=P*A*inv(P);
Aeigenvalues=eig(A)
Aneweigenvalues=(Anew)
```

to get

```
Aeigenvalues =
    0.4000
    0.5000

Aneweigenvalues =
    0.4000
    0.5000
```

EOCE 5.10

Consider the following system

$$y(n) + y(n-1) + y(n-2) = x_1(n) + x_2(n)$$

1. Is the system stable?
2. Find $y(n)$ if $x_1(n) = (.1)^n\, u(n)$ and $x_2(n) = (.1)^n\, u(n)$.
3. Find the transfer function $H(z)$.

Solution

1. Let us write the state and output equations first. Let

$$v_1(n) = y(n-2)$$

$$v_2(n) = y(n-1)$$

Then,

$$v_1(n+1) = y(n-1) = v_2(n)$$

and

$$\begin{aligned} v_2(n+1) = y(n) &= x_1(n) + x_2(n) - y(n-1) - y(n-2) \\ &= x_1(n) + x_2(n) - v_2(n) - v_1(n) \end{aligned}$$

The state and output equations are then

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 \\ -1 & -1 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_1(n) \\ x_2(n) \end{pmatrix}$$

$$\mathbf{y}(n) = \begin{pmatrix} -1 & -1 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} x_1(n) \\ x_2(n) \end{pmatrix}$$

with

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -1 & -1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} -1 & -1 \end{pmatrix} \quad \mathbf{D} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$

The eigenvalues for $\mathbf{A}$ are the roots of the determinant of $(z\mathbf{I} - \mathbf{A})$.

$$\det(z\mathbf{I} - \mathbf{A}) = \det\left(\begin{pmatrix} z & -1 \\ 1 & z+1 \end{pmatrix} \right)$$

$$\det(z\mathbf{I} - \mathbf{A}) = z^2 + z + 1$$

The roots are

$$z_{1,2} = \frac{-1 \pm \sqrt{1-4}}{2} = \frac{-1}{2} \pm j\sqrt{\frac{3}{2}}$$

The magnitude of the roots is unity. This means that the system is on the verge of instability or we can call it unstable.

2. With zero initial conditions,

$$\mathbf{V}(z) = (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z)$$

$$\mathbf{Y}(z) = \mathbf{C}\mathbf{V}(z) + \mathbf{D}\mathbf{X}(z)$$

We can substitute in the above state equation and write

$$\mathbf{V}(z) = \begin{pmatrix} \frac{z+1}{z^2+z+1} & \frac{1}{z^2+z+1} \\ \frac{-1}{z^2+z+1} & \frac{z}{z^2+z+1} \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \frac{z}{z-.1} \\ \frac{z}{z+.1} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{1}{z^2+z+1} & \frac{1}{z^2+z+1} \\ \frac{z}{z^2+z+1} & \frac{z}{z^2+z+1} \end{pmatrix} \begin{pmatrix} \frac{z}{z-.1} \\ \frac{z}{z+.1} \end{pmatrix}$$

$$\mathbf{V}(z) = \begin{pmatrix} \frac{z}{(z-.1)(z^2+z+1)} & + & \frac{z}{(z+.1)(z^2+z+1)} \\ \frac{z^2}{(z-.1)(z^2+z+1)} & + & \frac{z}{(z+.1)(z^2+z+1)} \end{pmatrix}$$

The output equation in the z-domain is

$$\mathbf{Y}(z) = \begin{pmatrix} -1 & -1 \end{pmatrix} \mathbf{V}(z) + \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} \frac{z}{z-.1} \\ \frac{z}{z+.1} \end{pmatrix}$$

or

$$\mathbf{Y}(z) = \frac{-z}{(z-.1)\left(z^2+z+1\right)} + \frac{-z}{(z+.1)\left(z^2+z+1\right)} + \frac{-z^2}{(z-.1)\left(z^2+z+1\right)}$$

$$+ \frac{-z}{(z+.1)\left(z^2+z+1\right)} + \frac{z}{z-.1} + \frac{z}{z+.1}$$

We can use MATLAB to find the inverse transform of $\mathbf{Y}(z)$ and get $\mathbf{y}(n)$. We will leave that as an exercise.

We can also use MATLAB to find $\mathbf{y}(n)$ as in the following script. We will use superposition first. This means that we will kill the inputs one at a time. In this case only the **B** and the **C** matrices will change.

$$\mathbf{A}=\begin{pmatrix}0 & 1\\ -1 & -1\end{pmatrix}\quad \mathbf{B}=\begin{pmatrix}0\\ 1\end{pmatrix}\quad \mathbf{C}=\begin{pmatrix}-1 & -1\end{pmatrix}\quad \mathbf{D}=(1)$$

We will then apply both inputs at once. The matrices in this case are

$$\mathbf{A}=\begin{pmatrix}0 & 1\\ -1 & -1\end{pmatrix}\quad \mathbf{B}=\begin{pmatrix}0 & 0\\ 1 & 1\end{pmatrix}\quad \mathbf{C}=\begin{pmatrix}-1 & -1\end{pmatrix}\quad \mathbf{D}=\begin{pmatrix}1 & 1\end{pmatrix}$$

The MATLAB script is next.

```
%We start with the solution using superposition
A=[0 1 ; -1 1]; B=[0;1];C=[-1 -1];D=[1];
isv=[0;0];%initial state vector
n=0:10;
x1=(.1).^n; x2=(-.1).^n;
[y1,v1]=dlsim(A,B,C,D,x1,isv);% dlsm: discrete linear
 syatem simulation
[y2,v2]=dlsim(A,B,C,D,x2,isv);
y=y1 + y2;
subplot(2,1,1);stem(n,y); xlabel('n'); ylabel('The
 response due to both inputs using superposition');
% Both inputs are applied together next
A=[0 1 ; -1 1]; B=[0 0;1 1];C=[-1 -1];D=[1 1];
isv=[0;0];%initial state vector
n=0:10;
x1=(.1).^n; x2=(-.1).^n;
x=[x1;x2];
[y,v]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear system
 simulation
subplot(2,1,2);stem(n,y); xlabel('n');
ylabel('The response due to both inputs applied together');
```

The plots are shown in Figure 5.14.

3. The transfer function $\mathbf{H}(z)$ is

$$\mathbf{H}(z)=\mathbf{C}(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}+\mathbf{D}=\begin{pmatrix}-1 & -1\end{pmatrix}\begin{pmatrix}\dfrac{z+1}{z^2+z+1} & \dfrac{1}{z^2+z+1}\\ \dfrac{-1}{z^2+z+1} & \dfrac{z}{z^2+z+1}\end{pmatrix}\begin{pmatrix}0 & 0\\ 1 & 1\end{pmatrix}+\begin{pmatrix}1 & 1\end{pmatrix}$$

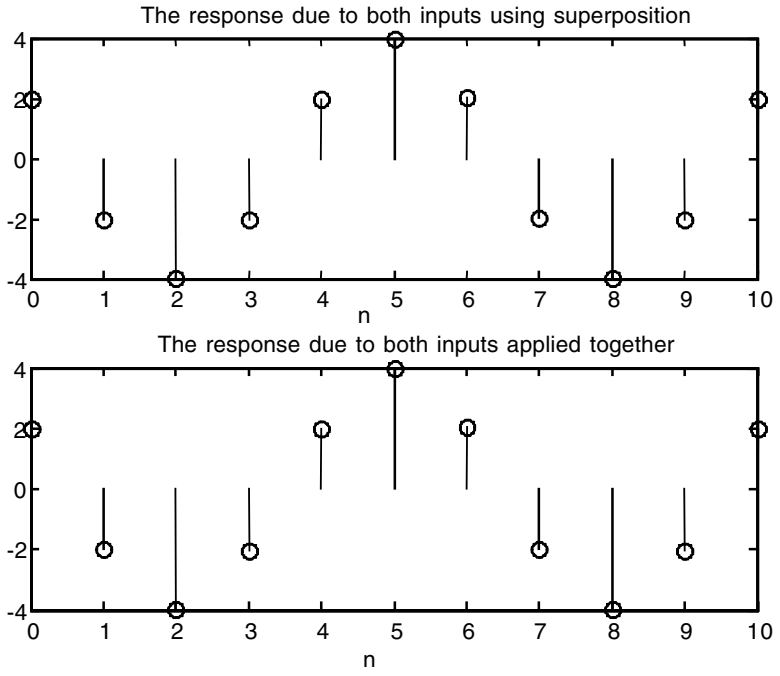

FIGURE 5.14 Plots for EOCE 5.10.

$$\mathbf{H}(z)=\begin{pmatrix}\dfrac{-(z+1)}{z^2+z+1}+\dfrac{1}{z^2+z+1} & \dfrac{-1}{z^2+z+1}+\dfrac{-z}{z^2+z+1}\end{pmatrix}\begin{pmatrix}0 & 0\\ 1 & 1\end{pmatrix}+\begin{pmatrix}1 & 1\end{pmatrix}$$

$$\mathbf{H}(z)=\begin{pmatrix}\dfrac{-1}{z^2+z+1}+\dfrac{-z}{z^2+z+1} & \dfrac{-1}{z^2+z+1}+\dfrac{-z}{z^2+z+1}\end{pmatrix}+\begin{pmatrix}1 & 1\end{pmatrix}$$

Since we had two inputs and single output, we should have one row in $\mathbf{H}(z)$ with two entries. They are

$$H_{11}(z)=\frac{-1}{z^2+z+1}+\frac{(-z)}{z^2+z+1}+1=\frac{z^2}{z^2+z+1}=\frac{Y(z)}{X_1(z)}$$

$$H_{12}(z)=\frac{-1}{z^2+z+1}+\frac{(-z)}{z^2+z+1}+1=\frac{z^2}{z^2+z+1}=\frac{Y(z)}{X_2(z)}$$

We can also use MATLAB to find $\mathbf{H}(z)$ as in the following script.

```
A=[0 1 ; -1 -1]; B=[0 0;1 1];C=[-1 -1];D=[1 1];
[num1 den]=ss2tf(A,B,C,D,1)%for the numerator of H11(z)
[num2 den]=ss2tf(A,B,C,D,2)%for the numerator of H12(z)
```

The result is

num1 = 1 0 0

den = 1.0000 1.0000 1.0000

num2 = 1 0 0

den = 1.0000 1.0000 1.0000

which agrees with the analytical results obtained previously.

EOCE 5.11

Consider the system

$$H(z) = \frac{z^3 + 3z^2 + 2z + 0}{z^3 - 6z^2 + 11z - 6}$$

Find the output $y(n)$ if the input is $x(n) = u(n)$. Use as many different state-space representations as you wish.

Solution

1. Using Example 5.2 we can write the first state-space representation by inspection. We have the state equation as

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 6 & -11 & 6 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

and the output equation as

$$\mathbf{y}(n) = \begin{bmatrix} 0-(-6)(1) & 2-11(1) & 3-(-6)(1) \end{bmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

$$= \begin{pmatrix} 6 & -9 & -9 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

2. Series connection. The transfer function can be written as

$$H(z) = \frac{z\left(z^2 + 3z + z\right)}{(z-2)(z-3)(z-1)} = \frac{z(z+2)(z+1)}{(z-2)(z-3)(z-1)}$$

$$H(z) = \frac{z}{z-2} \frac{z+2}{z-3} \frac{z+1}{z-1}$$

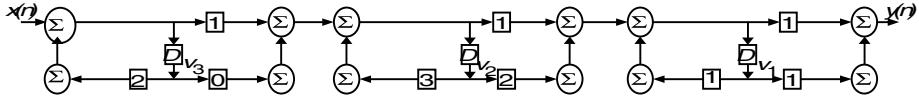

FIGURE 5.15 Block diagram for EOCE 5.11.

The block diagram is shown in Figure 5.15. From Figure 5.15 we have the state equations as

$$v_3(n+1) = x(n) + 2v_3(n)$$
$$v_2(n+1) = x(n) + 2v_3(n) + 3v_2(n)$$
$$v_1(n+1) = 2v_2(n) + x(n) + 2v_3(n) + 3v_2(n) + v_1(n)$$

and the output equation is

$$y(n) = 2v_2(n) + x(n) + 2v_3(n) + 3v_2(n) + 2v_1(n)$$

Thus the state-space system is

$$\mathbf{v}(n+1) = \begin{pmatrix} 1 & 5 & 2 \\ 0 & 3 & 2 \\ 0 & 0 & 2 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 2 & 5 & 3 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

3. Using partial fraction expansion we can write the transfer function as

$$H(z) = \frac{8z}{z-2} + \frac{10z}{z-3} + \frac{3z}{z-1}$$

The block diagram is shown in Figure 5.16. From Figure 5.16 we can see that the states are

$$v_1(n+1) = x(n) + 2v_1(n)$$
$$v_2(n+1) = x(n) + 3v_2(n)$$
$$v_3(n+1) = x(n) + v_3(n)$$

and the output is

$$y(n) = 8x(n) + 16v_1(n) + 10x(n) + 30v_2(n) + 3x(n) + 3v_3(n)$$

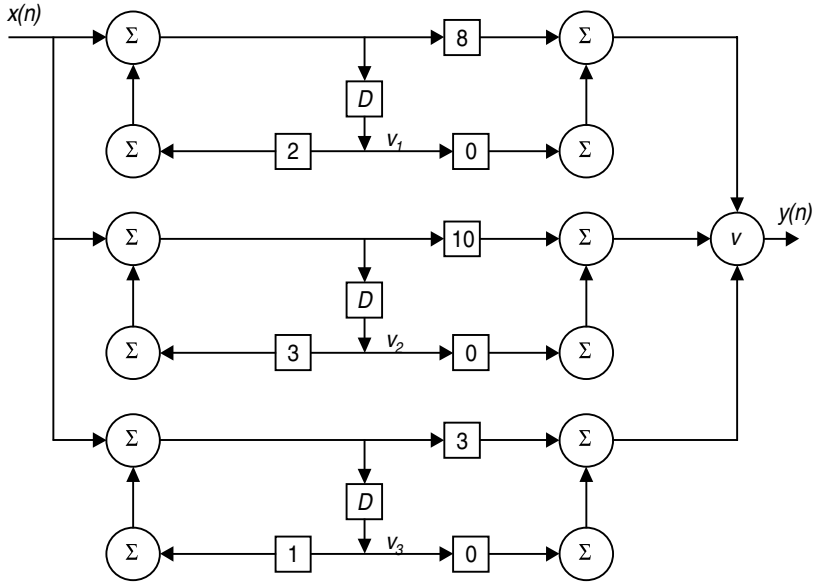

FIGURE 5.16 Block diagram for EOCE 5.11.

The states and output equations in matrix form are

$$\mathbf{v}(n+1) = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 16 & 30 & 3 \end{pmatrix} \mathbf{v}(n) + (21)\mathbf{x}(n)$$

4. We can represent the system in the block diagram as shown in Figure 5.17. Let the output of the first delay be $v_1(n)$. Then

$$v_1(n+1) = 6y(n) = 6\left[x(n) + v_3(n)\right] = 6v_3(n) + 6x(n)$$

Let the output of the second delay be $v_2(n)$ and the output of the third delay be $v_3(n)$. Thus

$$v_2(n+1) = 2x(n) + v_1(n) - 11v_3(n) - 11x(n)$$

$$v_3(n+1) = 3x(n) + v_2(n) + 6v_3(n) + 6x(n)$$

The output from the figure is given by

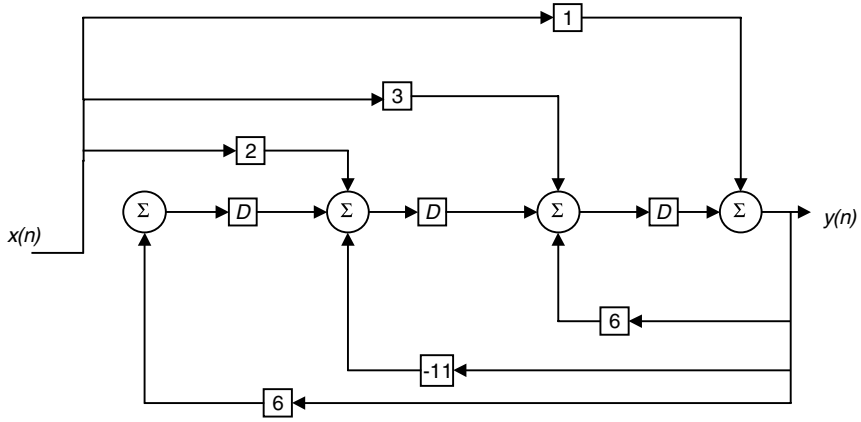

FIGURE 5.17 Block diagram for EOCE 5.11.

$$y(n) = x(n) + v_3(n)$$

The state and output equations are then

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 0 & 6 \\ 1 & 0 & -11 \\ 0 & 1 & 6 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 6 \\ -9 \\ 9 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

We will use MATLAB next to find the step response, $y(n)$, for all four representations and show that $y(n)$ for all four representations is the same. Let us also use zero initial conditions.

```
%For the first system
A=[0 1 0; 0 0 1; 6 -11 6]; B=[0;0;1];C=[6 -9 9];D=[1];
isv=[0;0;0];%initial state vector
n=0:10;
x=(1.^n);
[y,v]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear syatem
 simulation
subplot(2,2,1);
stem(n,y); ylabel('First representation');
%For the second system
A=[0 5 2; 0 3 2; 0 0 2]; B=[1;1;1];C=[2 5 3];D=[1];
```

```
[y,v]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear syatem
 simulation
subplot(2,2,2);
stem(n,y); ylabel('Second representation');
%For the third system
A=[2 0 0; 0 3 0; 0 0 1]; B=[1;1;1];C=[16 30 3];D=[21];
[y,v]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear syatem
 simulation
subplot(2,2,3);
stem(n,y); ylabel('Third representation');xlabel('n')
%For the fourth system
A=[0 0 6; 1 0 -11; 0 1 6]; B=[6;-9;9];C=[0 0 1];D=[1];
[y1,v1]=dlsim(A,B,C,D,x,isv);% dlsm: discrete linear syatem
 simulation
subplot(2,2,4);
stem(n,y); ylabel('Fourth representation');xlabel('n')
```

The plots are shown in Figure 5.18. Note in this example that the **A** matrix in the third case is diagonal. So for analytical solutions it is desirable that you do partial fraction expansion and then draw the block diagram from which you will obtain the **A** matrix in its diagonal form. Note also that if

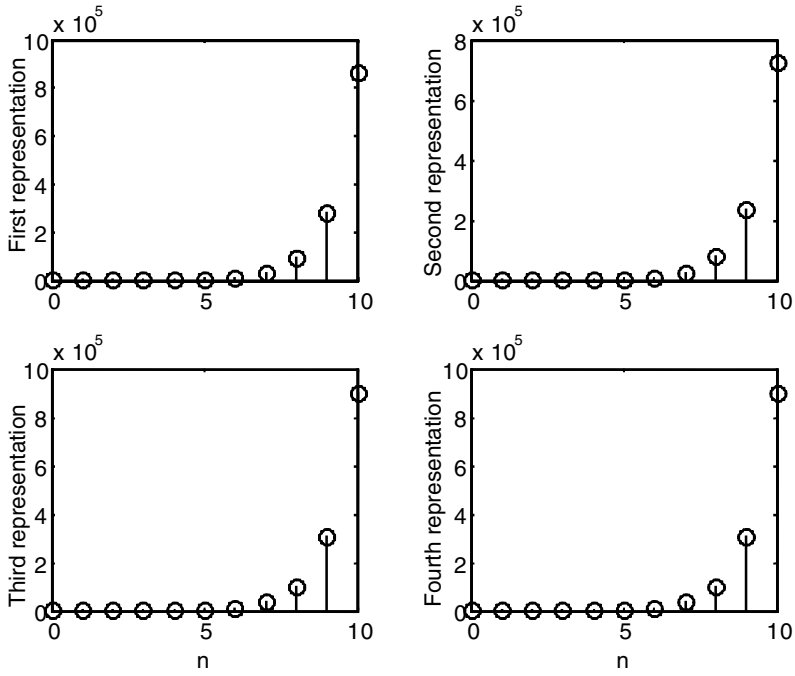

FIGURE 5.18 Plots for EOCE 5.11.

$$\mathbf{A} = \begin{pmatrix} a_1 & 0 & 0 \\ 0 & a_2 & 0 \\ 0 & 0 & a_3 \end{pmatrix}$$

then

$$\mathbf{A}^n = \begin{pmatrix} (a_1)^n & 0 & 0 \\ 0 & (a_2)^n & 0 \\ 0 & 0 & (a_3)^n \end{pmatrix}$$

5.10 End of Chapter Problems

EOCP 5.1

$$\text{Let } \mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -3 & -2 & -6 \end{pmatrix} \text{ and } \mathbf{B} = \begin{pmatrix} 0 & 0 & -1 \\ 1 & 0 & -2 \\ 0 & 1 & -3 \end{pmatrix}$$

1. Find $\mathbf{A}^2$, $\mathbf{A}^3$, $\mathbf{A}^{-1}$ and the eigenvalues and eigenvectors for $\mathbf{A}$.
2. Find $\mathbf{B}^2$, $\mathbf{B}^3$, $\mathbf{B}^{-1}$, eigenvalues and eigenvectors for $\mathbf{B}$.

EOCP 5.2

$$\text{For } \mathbf{B} = \begin{pmatrix} -2 & -3 & -4 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

1. Find eigenvalues and eigenvectors for $\mathbf{B}$.
2. Form the matrix $\mathbf{P}$ which has the eigenvectors as its columns.
3. Find $\mathbf{P}^{-1}\mathbf{B}\mathbf{P}$. Is it diagonal?
4. Find the eigenvalues for $\mathbf{P}^{-1}\mathbf{B}\mathbf{P}$.

EOCP 5.3

$$\text{With } \mathbf{A} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 4 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -1 & -3 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & -2 \\ 0 & 0 & -3 \end{pmatrix}$$

Find $\mathbf{A}^n$, $\mathbf{B}^n$ and $\mathbf{C}^n$.

EOCP 5.4

Consider the following difference equations

1. $y(n)+a_1y(n-1)+a_2y(n-2)=b_0x(n)$
2. $y(n)+a_1y(n-1)+a_2y(n-2)=b_0x(n)+b_1x(n-1)$
3. $y(n)+a_3y(n-3)=b_0x(n)$
4. $y(n)+a_4y(n-4)=b_0x(n)+b_1x(n-1)$
5. $y(n)+a_3y(n-3)+a_4y(n-4)=b_1x(n-1)$
6. $y(n)+a_1y(n-1)+a_2y(n-2)=b_0x_1(n)+b_1x_2(n)$
7. $y(n)-a_4y(n-4)=b_0x_1(n-1)+b_0x_2(n)$
8. $y(n)-y(n-3)=x(n)+1$
9. $y(n)+y(n-2)=x_1(n)+x_1(n)-10$

Find the state-space representation for each system above. Find **A**, **B**, **C** and **D**.

EOCP 5.5

Given the following impulse responses

1. $h(n)=(.1)^n u(n)+(-.1)^n u(n)$
2. $h(n)=(.1)^n \sin(n)u(n)$
3. $h(n)=n(.1)^n \cos\left(\frac{3\pi}{2}n\right)u(n)$
4. $h(n)=(n-1)(.1)^{n-1}u(n-1)$
5. $h(n)=(.1)^n u(n)*(.5)^n u(n)$

Find the state-space representation for each system. Find **A**, **B**, **C** and **D**.

EOCP 5.6

Consider the following blocks in Figures 5.19 through 5.23. Find the state-space representation for all blocks. Find **A**, **B**, **C** and **D**.

EOCP 5.7

Consider the following transfer functions

1. $H(z)=\dfrac{(z-1)(z-.5)}{(z+1)(z+.5)}$
2. $H(z)=\dfrac{z}{(z+1)(z+.5)(z-1)}$
3. $H(z)=\dfrac{(z-1)(z-.5)}{(z+1)}$

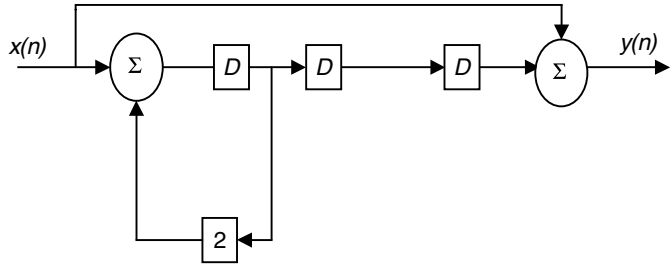

FIGURE 5.19 Block for EOCP 5.6.

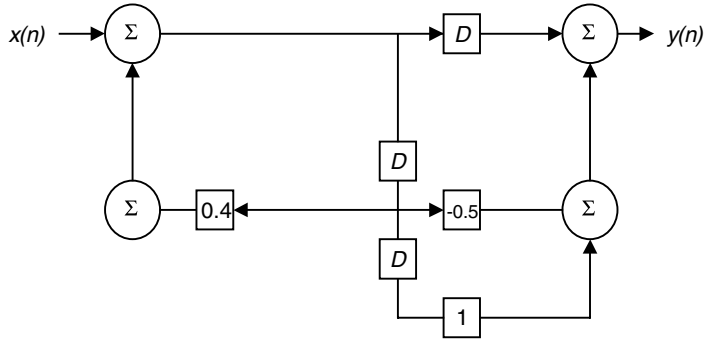

FIGURE 5.20 Block for EOCP 5.6.

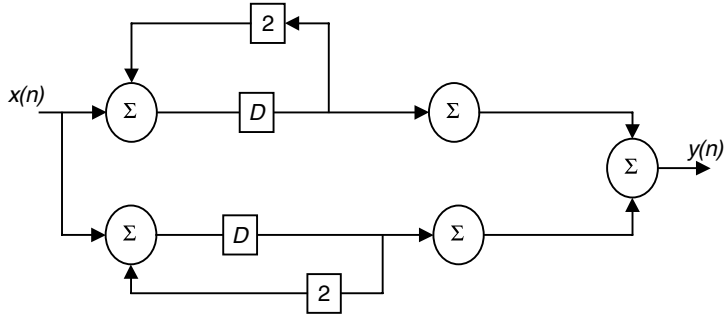

FIGURE 5.21 Block for EOCP 5.6.

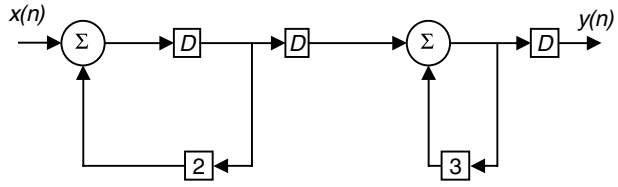

FIGURE 5.22 Block for EOCP 5.6.

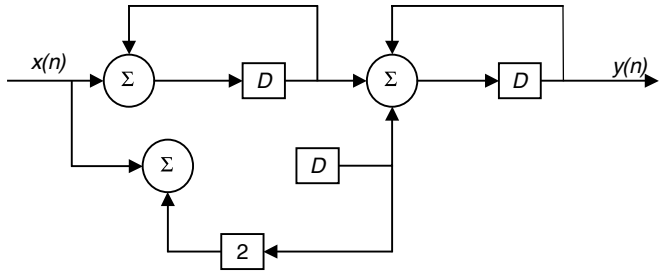

FIGURE 5.23 Block for EOCP 5.6.

4. $H(z) = \dfrac{z^2 - 1}{z^3}$
5. 5. $H(z) = \dfrac{z^3 + 3z^2 + 2z + 1}{z^3 + 4z^2 + 2z + 3}$

Find four different state-space representations for each transfer function.

EOCP 5.8

Consider the system

$$y(n) + ky(n-1) + 3y(n-2) = x(n)$$

Use the state-space in all parts.

1. For what value k is the system stable?
2. Take a value for k that makes the system stable and find the eigenvalues of the system.
3. For the value of k in part 2 find the output $y(n)$ for $x(n) = u(n)$ using MATLAB.
4. Repeat part 3 using the z-transform method.
5. Find the transition matrix $\mathbf{A}^n$.
6. Find $H(z)$ from the state equations analytically.
7. Find $H(z)$ using MATLAB.
8. Use MATLAB to find $h(n)$ from $H(z)$. Find the residues using MATLAB.

EOCP 5.9

Consider the system

$$y(n) + 0.1y(n-1) + y(n-2) = x_1(n) + x_2(n)$$

with $y(-1) = 0$ and $y(-2) = 1$.

1. Is the system stable?
2. Put the system in state-space.
3. Use MATLAB to find $\mathbf{h}(n)$ using the state equations.
4. Use MATLAB to find $y(n)$, the step response.
5. Use MATLAB to find the derived initial conditions with $x_1(0) = x_2(0) = 0$.
6. Find $\mathbf{H}(z)$ using MATLAB and identify each entry with the $\mathbf{H}(z)$ matrix.
7. Find $\mathbf{A}^n$.

EOCP 5.10

Consider the systems

$$H(z) = \frac{z+2}{z^2+z+1}$$

$$H(z) = \frac{1}{z^2-.6z+.05}$$

For both systems and using state-space method

1. Find two state-space representations.
2. Find $h(n)$.
3. Find $\mathbf{A}^n$.
4. Find the step response.
5. Check stability for both systems.

EOCP 5.11

Consider the system

$$H(z) = \frac{z^2+z+1}{z^2+kz+.05}$$

1. Write the state-space representation.
2. For what value(s) of k is the system stable?
3. Find the difference equation representing $H(z)$.
4. Pick a k that makes the system stable and find the output $y(n)$ for $x(n) = n\sin(n)u(n)$. Use the MATLAB function `dlsim` to do that.
5. Find $\mathbf{A}^n$ for a given k.

EOCP 5.12

Consider the system

$$y_1(n) - y_2(n-1) = x_1(n)$$

$$y_2(n) - y_1(n-1) = x_2(n)$$

where $y_1(n)$ and $y_2(n)$ are the outputs and $x_1(n)$ and $x_2(n)$ are the inputs.

1. Write the state-space equations describing the system.
2. Is the system stable? Find the eigenvalues for **A**.
3. With $x_1(n) = x_2(n) = u(n)$, find $y_1(n)$ and $y_2(n)$.
4. Find the transfer function matrix $H(z)$ from **A**, **B**, **C** and **D**.
5. Find the state transition matrix.
6. Find the impulse response of the system.
7. Draw the block diagram for the system and obtain a different state-space representation.
8. Find the step and impulse responses for the representation in part 7.
9. What are the eigenvalues for the new representation in part 7.

EOCP 5.13

Consider the system in Figure 5.24.

1. Write the state-equations for this system.
2. Is the system stable?
3. What is the transition matrix?
4. What is the transfer matrix $H(z)$?
5. Find $y_1(n)$ and $y_1(n)$ for $x(n) = 10\cos\left(\frac{2\pi n}{3} + .1\right)u(n)$.
6. Find the impulse response for the system.
7. Write the coupled two difference equations describing the system.

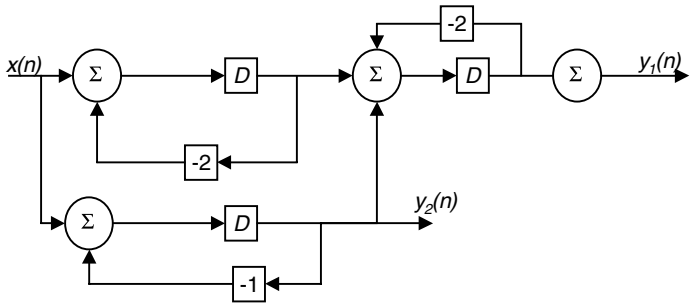

FIGURE 5.24 System for EOCP 5.13.

EOCP 5.14

Consider the systems

$$a)\quad H(z)=\begin{pmatrix} \dfrac{z}{z^2+z+1} & \dfrac{z^2+1}{z^2+z+1} \\ \dfrac{z+1}{z^2+z+1} & \dfrac{z^2+z}{z^2+z+1} \end{pmatrix} \qquad b)\quad H(z)=\begin{pmatrix} \dfrac{z}{z^2+1} & \dfrac{z+1}{z^2+1} \end{pmatrix}$$

1. Draw the block diagram for these systems.
2. What are the state-space representations for both systems?
3. What is the transition matrix for both systems?
4. Find the step response for both systems.
5. Find the impulse response for both systems.
6. Are the systems stable?

6

Modeling and Representation of Discrete Linear Systems

6.1 Introduction

The transfer function, $H(z)$, for linear time-invariant systems can be inverse transformed to produce $h(n)$, the impulse response of the system. Thus $h(n)$, the signal, can be modeled using the z-transform approach to represent the transfer function of the system. In communications, we broadcast signals of various types, and in most situations we sample the signal to be transmitted and then transmit it via a communication channel using a huge number of data points. Due to many factors, such as interference, these data values get distorted. However, we can represent the signal to be transmitted using the z-transform approach and transmit only the coefficients of the numerator and the denominator of the transfer function $H(z)$.

When we studied the general form of the sinusoidal signal in previous chapters, we modeled the signal as

$$x(n) = A\cos(\theta n+\varphi) \tag{6.1}$$

To avoid loosing data points during transmission, we can send only the amplitude of the signal, its frequency and its phase. This is easier and more efficient.

The transfer function, $H(z)$, and the signal $x(n) = A\cos(\theta n +\varphi)$, among many other representations, are models that we employ for certain uses. In this book we use modeling primarily for the purpose of analysis and design of linear time-invariant systems.

We can model mechanical systems, for example, using the laws of Newton to write a set of differential equations. We can model electrical systems as well using similar laws to derive differential equations that relate different components in the system. We do similar things in chemical systems, mechanics and dynamics. Basically, we derive differential equations from existing systems. If various parameters in these systems are unknown, we can approximate these systems by finding their impulse response and then

derive the differential equations represented by these responses. Discrete systems can then be derived from these models; sometimes we find that there are systems that are inherently discrete in nature.

6.2 Five Ways of Representing Discrete Linear Systems

In this chapter we will consider representing linear time-invariant systems using all the techniques that we have presented so far in this book. We can represent linear time-invariant systems in many ways. The following five ways are considered.

1. Using linear difference equations with constant coefficients
2. Using the impulse response function, $h(n)$
3. Using the transfer function approach
4. Using block diagrams
5. Using state-space approach

To do so we will learn by example; we will consider many examples in this chapter and see how we can move from one representation to another. Given one representation from the five representations listed above, we should be able to deduce the other representations. We will also find the output, $y(n)$, given the input, $x(n)$, along with the necessary initial conditions and demonstrate that the output $y(n)$ will be the same for the same input $x(n)$ and the same given initial conditions for the same system.

6.2.1 From the Difference Equation to the Other Four Representations

6.2.1.1 The Difference Equation Representation

Consider the difference equation

$$y(n) + 3y(n-1) + 2y(n-2) = x(n) \text{ with } y(-1) = y(-2) = 0 \text{ and } x(n) = u(n)$$

The characteristic equation is

$$z^2 + 3z + 2 = 0$$

where $z = -2$ and $z = -1$ are the poles of the system. The solution has two terms, the homogeneous and the particular. The homogeneous solution is

$$y_h(n) = c_1(-2)^n + c_2(-1)^n$$

Since $x(n) = u(n)$, a constant, the particular solution is

$$y_p(n) = c_3$$

With $y_p(n) = c_3$, the original equation becomes

$$c_3 + 3c_3 + 2c_3 = 1$$

with $c_3 = 1/6$. Then the total solution is

$$y(n) = c_1(-2)^n + c_2(-1)^n + 1/6$$

Next we evaluate the constants. With the help of the initial conditions we get

$$y(-1) = 0 = -c_1/2 - c_2 + 1/6$$
$$y(-2) = 0 = c_1/4 + c_2 + 1/6$$

We can solve the above equations to get

$$c_1 = 4/3 \text{ and } c_2 = -1/2$$

The total solution is then

$$y(n) = 4/3(-2)^n + -1/2(-1)^n + 1/6$$

6.2.1.2 The Impulse Response Representation

With $x(n) = \delta(n)$, the difference equation becomes

$$y(n) + 3y(n-1) + 2y(n-2) = \delta(n)$$

The solution in this case has two parts. Since the input is a delta, the particular solution is zero. The total solution in this case is

$$h(n) = c_1(-2)^n + c_2(-1)^n$$

The impulse response is derived with zero initial conditions. Thus we will not be able to use the given initial conditions; we will find our own. From the difference equation with the delta signal as the input we have

$$h(0) = 1 \text{ and } h(1) = 0 - 3(1) - 2(0) = -3$$

From the solution we obtained we have

$$h(0) = c_1 + c_2 \text{ and } h(1) = -c_1 - 2c_2$$

So by equating these derived initial conditions we get

$$c_1 + c_2 - 1 \text{ and } -c_1 - 2c_2 = -3$$

Solving for c_1 and c_2 we have

$$c_1 = -1 \text{ and } c_2 = 2$$

The final solution is

$$h(n) = -1(-2)^n + 2(-1)^n$$

The output $y(n)$ for $x(n) = u(n)$ is

$$y(n) = \sum_{m=-\infty}^{\infty} \left((-1)(-2)^m + 2(-2)^m\right) u(m)u(n-m) = \sum_{m=0}^{n} (-1)(-1)^m + 2(-2)^m$$

$$y(n) = (-1)\left[\frac{1-(-1)^{n+1}}{1-(-1)}\right] + 2\left[\frac{1-(-2)^{n+1}}{1-(-2)}\right] \qquad n \geq 0$$

6.2.1.3 The z-Transform Representation

We can z-transform the difference equation term by term and get

$$Y(z) + 3z^{-1}Y(z) + 2z^{-2}Y(z) = X(z)$$

from which we have the transfer function representation as

$$H(z) = \frac{z^2}{z^2 + 3z + 2} = \frac{z^2}{(z+1)(z+2)}$$

The poles of the system are the roots of the denominator in the transfer function $H(z)$. With $x(n) = u(n)$ we have

$$Y(z) = \frac{zz^2}{(z-1)(z+2)(z+1)}$$

We can do partial fraction expansion to find $y(n)$. First we do partial fraction expansion on $Y(z)/z$.

$$\frac{Y(z)}{z} = \frac{z^2}{(z-1)(z+2)(z+1)} = \frac{A}{z-1} + \frac{B}{z+1} + \frac{C}{z+2}$$

where $A = 1/6$, $B = -1/2$ and $C = 4/3$.

Finally, the output in the z-domain is

$$Y(z) = \frac{z\frac{1}{6}}{z-1} + \frac{\frac{-1}{2}z}{z+1} + \frac{\frac{4}{3}z}{z+2}$$

The output $y(n)$ is the inverse transform of $Y(z)$ and is

$$y(n) = \frac{1}{6} - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.1.4 The State-Space Representation

With the difference equation

$$y(n) + 3y(n\text{-}1) + 2y(n-2) = x(n)$$

let

$$v_1(n) = y(n-2)$$

$$v_2(n) = y(n-1)$$

Then

$$v_1(n+1) = y(n-1) = v_2(n)$$

$$v_2(n+1) = y(n) = x(n) - 3y(n-1) - 2y(n-2) = x(n) - 3v_2(n) - 2v_1(n)$$

The output and the state-space equations are

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix}\mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(n)$$

$$y(n) = \begin{pmatrix} -2 & -3 \end{pmatrix}\mathbf{v}(n) + (1)\mathbf{x}(n)$$

The eigenvalues for the system matrix **A** are at –1 and –2, thus the system is unstable.

To find the output of the system using state-space we proceed as in the following. The state vector in the z-domain is

$$\mathbf{V}(z) = z(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{v}(0) + (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{X}(z) = \begin{pmatrix} z & -1 \\ 2 & z+3 \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \frac{z}{z-1}$$

$$\mathbf{V}(z) = \begin{pmatrix} z+3 & 1 \\ -2 & z \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \frac{z}{(z-1)(z^2+3z+2)} = \begin{pmatrix} \dfrac{z}{(z-1)(z^2+3z+2)} \\ \dfrac{z^2}{(z-1)(z^2+3z+2)} \end{pmatrix}$$

The output vector is

$$\mathbf{Y}(z) = \begin{pmatrix} -2 & -3 \end{pmatrix} \begin{pmatrix} \dfrac{z}{(z-1)(z^2+3z+2)} \\ \dfrac{Z^2}{(z-1)(z^2+3z+2)} \end{pmatrix} + \frac{z}{z-1}$$

$$Y(z) = \frac{-2z}{(z-1)(z^2+3z+2)} + \frac{-3z^2}{(z-1)(z^2+3z+2)} + \frac{z}{z-1} = \frac{z^2}{(z-1)(z^2+3z+2)}$$

This is what we have found earlier and the output $y(n)$ is

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \quad n \geq 0$$

6.2.1.5 The Block Diagram Representation

The original system is shown in Figure 6.1. From the block we have

$$v_1(n+1) = v_2(n)$$
$$v_2(n+1) = x(n) - 3v_2(n) - 2v_1(n)$$
$$y(n) = x(n) - 3v_2(n) - 2v_1(n)$$

Thus the state-space system is

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} -2 & -3 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

This representation is similar to the earlier state-space representation and the output $y(n)$ is obtained in a similar way.

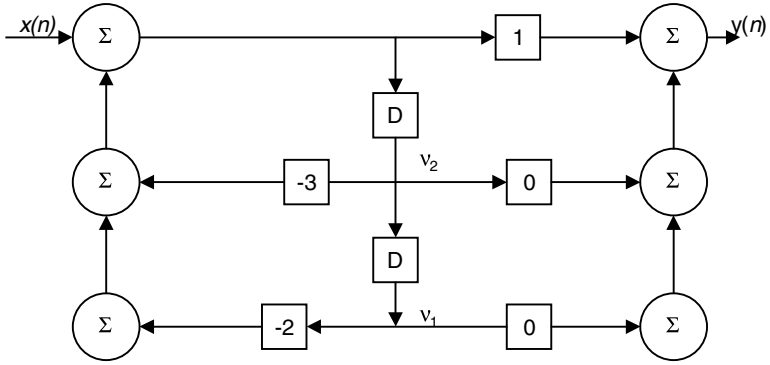

FIGURE 6.1 Bock diagram representation.

6.2.2 From the Impulse Response to the Other Four Representations

6.2.2.1 The Impulse Response Representation

We will consider the same system, namely,

$$h(n) = \left[-1(-2)^n + 2(-1)^n\right]u(n)$$

If the input is the unit step, the output can be evaluated using the convolution sum and the result is repeated here as

$$y(n) = (-1)\left[\frac{1-(-1)^{n+1}}{1-(-1)}\right] + 2\left[\frac{1-(-2)^{n+1}}{1-(-2)}\right] \qquad n \geq 0$$

6.2.2.2 The Transfer Function Representation

With

$$h(n) = \left[-1(-2)^n + 2(-1)^n\right]u(n)$$

The z-transform is obtained as

$$H(z) = \frac{-z}{z+1} + \frac{2z}{z+2} = \frac{-z(z+2)+2z(z+1)}{(z+1)(z+2)} = \frac{z^2}{z^2+3z+2}$$

With the unit step as an input, the output $y(n)$ is inverse z-transform of

$$Y(z) = \frac{z^2}{z^2+3z+2}\frac{z}{z-1}$$

and it was calculated earlier as

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.2.3 The Difference Equation Representation

From the transfer function representation

$$H(z) = \frac{z^2}{z^2 + 3z + 2} = \frac{Y(z)}{X(z)}$$

we can write

$$Y(z)\left[z^2 + 3z + 2\right] = z^2\left[X(z)\right]$$

By inverse transforming the above equation we arrive at the difference equation

$$y(n+2) + 3y(n+1) + 2y(n) = x(n+2)$$

For $n = n - 2$, we have the difference equation

$$y(n) + 3y(n-1) + 2y(n-2) = x(n)$$

The output for a step input was derived earlier and is

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.2.4 The State-Space Representation

With

$$H(z) = \frac{z^2}{z^2 + 3z + 2}$$

We have the block diagram as shown in Figure 6.1. The state and output equations are again

$$v_1(n+1) = v_2(n)$$
$$v_2(n+1) = x(n) - 3v_2(n) - 2v_1(n)$$
$$y(n) = x(n) - 3v_2(n) - 2v_1(n)$$

and the output $y(n)$ is as obtained earlier.

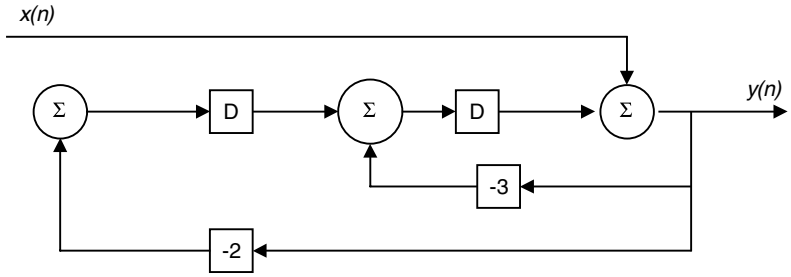

FIGURE 6.2 A different block representation.

6.2.2.5 The Block Diagram Representation

With

$$H(z) = \frac{z^2}{z^2 + 3z + 2}$$

We can draw a different block diagram as shown in Figure 6.2. Let $v_1(n)$ be the output of the first delay and $v_2(n)$ be the output of the second delay. Then

$$v_1(n+1) = -2y(n) = -2x(n) - 2v_2(n)$$

$$v_2(n+1) = -3y(n) + v_1(n) = -3x(n) - 3v_2(n) + v_1(n)$$

$$y(n) = x(n) + v_2(n)$$

The state matrix equations are then

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & -2 \\ 1 & -3 \end{pmatrix}\mathbf{v}(n) + \begin{pmatrix} -2 \\ -3 \end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 1 \end{pmatrix}\mathbf{v}(n) + (1)\mathbf{x}(n)$$

The eigenvalues for **A** are –1 and –2 and again the system is unstable. The solution is again

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.3 From the Transfer Function to the Other Four Representations

6.2.3.1 The Transfer Function Representation

Consider the same system

$$H(z) = \frac{z^2}{z^2 + 3z + 2}$$

With the unit step as an input, the output $y(n)$ is the inverse z-transform of

$$Y(z) = \frac{z^2}{z^2 + 3z + 2} \frac{z}{z - 1}$$

and it was calculated earlier as

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.3.2 The Impulse Response Representation

With

$$H(z) = \frac{z^2}{z^2 + 3z + 2}$$

The impulse response $h(n)$ is the inverse z-transform of $H(z)$. We will do partial fraction expansion on

$$\frac{H(z)}{z} = \frac{z}{z^2 + 3z + 2} = \frac{A}{z + 1} + \frac{B}{z + 2} = \frac{-1}{z + 1} + \frac{2}{z + 2}$$

Thus

$$H(z) = \frac{zA}{z + 1} + \frac{zB}{z + 2} = \frac{-1z}{z + 1} + \frac{2z}{z + 2}$$

and $h(n)$ is

$$h(n) = -(-1)^n + 2(-2)^n \qquad n \geq 0$$

The output $y(n)$ is the convolution sum between $u(n)$ and $h(n)$ and was calculated earlier.

6.2.3.3 The Difference Equation Representation

This was derived earlier from

$$H(z) = \frac{z^2}{z^2 + 3z + 2} = \frac{Y(z)}{X(z)}$$

and will not be repeated here.

6.2.3.4 The State-Space Representation

From

$$H(z)=\frac{z^2+0z+0}{z^2+3z+2}$$

we can obtain the state equations by inspection as we saw in Example 5.2. The state and the output equations are

$$\mathbf{v}(n+1)=\begin{pmatrix}0 & 1\\ -2 & -3\end{pmatrix}\mathbf{v}(n)+\begin{pmatrix}0\\ 1\end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n)=\begin{pmatrix}-2 & -3\end{pmatrix}\mathbf{v}(n)+(1)\mathbf{x}(n)$$

The eigenvalues for **A** are at –1 and –2 again and thus this system is unstable. The output $y(n)$ can easily be calculated and is left for you.

6.2.3.5 The Block Diagram Representation

With

$$H(z)=\frac{-1z}{z+1}+\frac{2z}{z+2}$$

We can draw the block diagram as in Figure 6.3. From Figure 6.3 we can obtain the state and the output equations as

$$v_1(n+1)=x(n)-v_1(n)$$

$$v_2(n+1)=-4x(n)-2v_2(n)$$

$$y(n)=x(n)+v_1(n)+v_2(n)$$

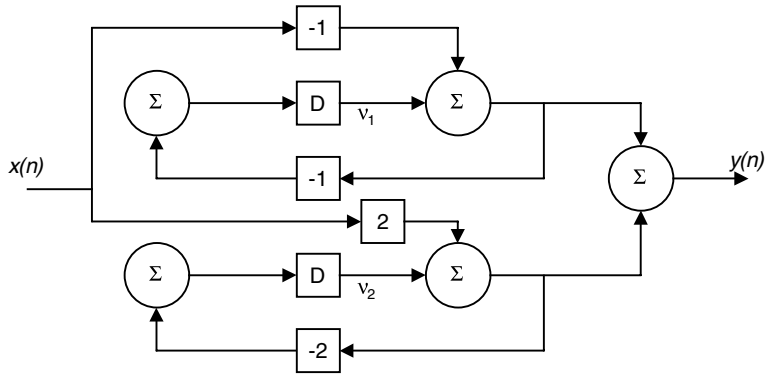

FIGURE 6.3 Another block representation.

and the matrix state equations are

$$\mathbf{v}(n+1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix} \mathbf{v}(\mathrm{n}) + \begin{pmatrix} 1 \\ -4 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 1 & 1 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

The given values of **A** in this case are at –1 and –2 and the system is again unstable. The same solution can be obtained again for $y(n)$.

6.2.4 From the State-Space to the Other Four Representations

6.2.4.1 The State-Space Representation

Consider the states and the output equations

$$\mathbf{v}(n+1) = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 1 \\ -4 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = (1 \quad 1)\mathbf{v}(n) + (1)\mathbf{x}(n)$$

It is clear here that the system is unstable and the output $y(n)$ is as calculated earlier.

6.2.4.2 The Transfer Function Representation

The transfer function $\mathbf{H}(z)$ can be calculated from the state matrix equations as

$$\mathbf{H}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} = (1 \quad 1)\begin{pmatrix} z+1 & 0 \\ 0 & z+2 \end{pmatrix}^{-1}\begin{pmatrix} 1 \\ -4 \end{pmatrix} + 1$$

$$= (1 \quad 1)\begin{pmatrix} \frac{1}{z+1} & 0 \\ 0 & \frac{1}{z+2} \end{pmatrix}\begin{pmatrix} 1 \\ -4 \end{pmatrix} + 1$$

$$\mathbf{H}(z) = \begin{pmatrix} \frac{1}{z+1} & \frac{1}{z+2} \end{pmatrix}\begin{pmatrix} 1 \\ -4 \end{pmatrix} + 1 = \frac{1}{z+1} - \frac{4}{z+2} + 1$$

$$= \frac{z+2-4(z+1)+z^2+3z+2}{z^2+3z+2}$$

After some simplifications we arrive at

$$H(z) = \frac{z^2}{z^2+3z+2}$$

The output $Y(z)$ is

$$Y(z) = \frac{z}{z-1}\ \frac{z^2}{z^2+3z+2}$$

and $y(n)$ is again

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.4.3 The Impulse Response Representation

From

$$H(z) = \frac{z^2}{z^2+3z+2}$$

The impulse response is the inverse transform of $H(z)$ and is

$$h(n) = -(-1)^n + 2(-2)^n \qquad n \geq 0$$

The output is the convolution between the step input signal and the impulse response. It was shown earlier and is presented again as

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.4.4 The Difference Equation Representation

We have the state and the output equations as

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix}\mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} -2 & -3 \end{pmatrix}\ \mathbf{v}(n) + (1)\mathbf{x}(n)$$

The states are

$$v_1(n+1) = v_2(n)$$

$$v_2(n+1) = -2v_1(n) - 3v_2(n)$$

Let

$$v_1(n) = y(n-2)$$

$$v_2(n) = y(n-1)$$

Then

$$v_1(n+1) = y(n-1)$$

$$v_2(n+1) = y(n)$$

The output equation is

$$y(n) = -2v_1(n) - 3v_2(n) + x(n) = -2y(n-2) - 3y(n-1) + x(n)$$

By arranging terms we get the difference equation as

$$y(n) + 2y(n-2) + 3y(n-1) = x(n)$$

The output due to a step input was calculated earlier.

6.2.4.5 The Block Diagram Representation

We have the states and the output equations as

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} -2 & -3 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

From the state and the output equation we can draw the block diagram as shown in Figure 6.4. The system is again unstable and the output is still

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

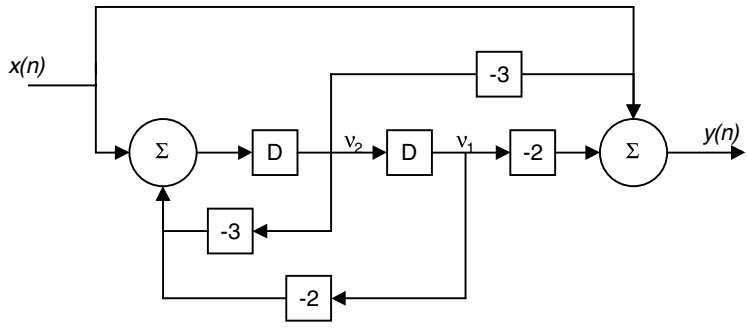

FIGURE 6.4 Another block representation.

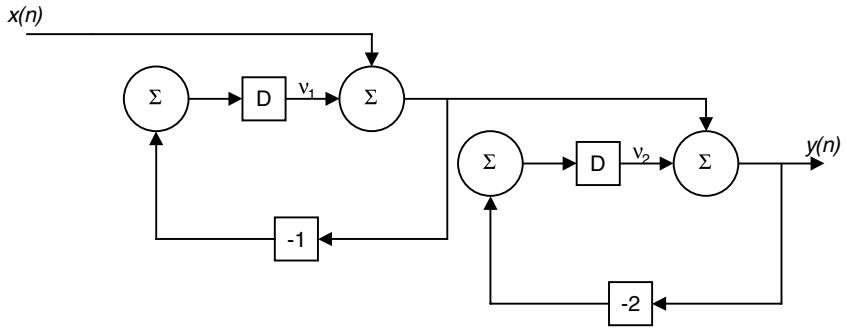

FIGURE 6.5 Another block representation.

6.2.5 From the Block Diagram to the Other Four Representations

Consider the block diagram in Figure 6.5.

6.2.5.1 The State-Space Representation

The states are written using the block in Figure 6.5 as

$$v_1(n+1) = -v_1(n) - x(n)$$
$$v_2(n+1) = -2v_1(n) - 2v_1(n) - 2x(n)$$

The output is

$$y(n) = v_1(n) + v_2(n) + x(n)$$

The state and matrix equations are then

$$\mathbf{v}(n+1) = \begin{pmatrix} -1 & 0 \\ -2 & -2 \end{pmatrix} \mathbf{v}(n) + \begin{pmatrix} -1 \\ -2 \end{pmatrix} \mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 1 & 1 \end{pmatrix} \mathbf{v}(n) + (1)\mathbf{x}(n)$$

The eigenvalues for **A** are the roots of the determinant of $(z\mathbf{I} - \mathbf{A})$.

$$(z\mathbf{I} - \mathbf{A}) = \begin{pmatrix} z & 0 \\ 0 & z \end{pmatrix} - \begin{pmatrix} -1 & 0 \\ -2 & -2 \end{pmatrix} = \begin{pmatrix} z+1 & 0 \\ 2 & z+2 \end{pmatrix}$$

The determinant of $(z\mathbf{I} - \mathbf{A})$ set to zero is

$$z^2 + 3z + 2 = 0$$

and has the roots at –1 and –2 which indicates that the system is unstable. To find the output we need the inverse of ($z\mathbf{I} - \mathbf{A}$).

$$(z\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{z+2}{z^2+3z+2} & 0 \\ \dfrac{-2}{z^2+3z+2} & \dfrac{z+1}{z^2+3z+2} \end{pmatrix} = \begin{pmatrix} \dfrac{1}{z+1} & 0 \\ \dfrac{-2}{(z+1)(z+2)} & \dfrac{1}{z+2} \end{pmatrix}$$

The output therefore is

$$\mathbf{Y}(z) = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} \dfrac{1}{z+1} & 0 \\ \dfrac{-2}{(z+1)(z+2)} & \dfrac{1}{z+2} \end{pmatrix} \begin{pmatrix} -1 \\ -2 \end{pmatrix} \left(\frac{z}{z-1}\right) + (1)\frac{z}{z-1}$$

$$\mathbf{Y}(z) = \begin{pmatrix} \dfrac{1}{z+1} - \dfrac{2}{(z+1)(z+2)} & \dfrac{1}{z+2} \end{pmatrix} \begin{pmatrix} -1 \\ -2 \end{pmatrix} \left(\frac{z}{z-1}\right) + (1)\frac{z}{z-1}$$

$$\mathbf{Y}(z) = \left(\frac{-1}{z+1} + \frac{2}{(z+1)(z+2)} - \frac{2}{z+2}\right)\left(\frac{z}{z-1}\right) + (1)\frac{z}{z-1}$$

$$\mathbf{Y}(z) = \left(\frac{-z-2+2-2(z+1)}{(z+1)(z+2)}\right)\left(\frac{z}{z-1}\right) + (1)\frac{z}{z-1}$$

$$= \left(\frac{-3z^2-2z}{(z-1)(z+1)(z+2)}\right) + \left(\frac{z}{z-1}\right)$$

Finally, the output in the z-domain is given by

$$\mathbf{Y}(z) = \frac{-3z^2-2z+z\left(z^2+3z+2\right)}{(z-1)(z+1)(z+2)} = \frac{z^3}{(z-1)(z+1)(z+2)}$$

The solution $y(n)$ for the above equation was obtained earlier as

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.5.2 The Transfer Function Representation

With

$$\mathbf{A} = \begin{pmatrix} -1 & 0 \\ -2 & -2 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} -1 \\ -2 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 1 & 1 \end{pmatrix} \text{ and } \mathbf{D} = (1),$$

the transfer function is

$$\mathbf{H}(z) = \mathbf{C}(z\mathbf{I}\text{-}\mathbf{A})^{-1}\,\mathbf{B} + \mathbf{D} = \begin{pmatrix}1 & 1\end{pmatrix}\begin{pmatrix}\dfrac{1}{z+1} & 0\\ \dfrac{-2}{(z+1)(z+2)} & \dfrac{1}{z+2}\end{pmatrix}\begin{pmatrix}-1\\-2\end{pmatrix} + 1$$

$$\mathbf{H}(z) = \left[\left(\frac{1}{z+1} \quad \frac{-2}{(z+1)(z+2)}\right)\frac{1}{z+2}\right]\begin{pmatrix}-1\\-2\end{pmatrix} + 1$$

$$= \frac{-1}{z+1} + \frac{2}{(z+1)(z+2)} - \frac{2}{z+2} + 1$$

$$\mathbf{H}(z) = \frac{-z-2+2-2z-2}{(z+1)(z+2)} + \frac{z^2+3z+2}{(z+1)(z+2)}$$

Finally,

$$\mathbf{H}(z) = \frac{-z-2+2-2z-2}{(z+1)(z+2)} + \frac{z^2+3z+2}{(z+1)(z+2)} = \frac{z^2}{z^2+3z+2}$$

6.2.5.3 The Impulse Response Representation

From

$$H(z) = \frac{z^2}{z^2+3z+2}$$

the impulse response is the inverse transform of $H(z)$ and is

$$h(n) = -(-1)^n + 2(-2)^n \qquad n \geq 0$$

The output is the convolution between the step input signal and the impulse response. It was shown earlier and is presented again as

$$y(n) = \frac{1}{6}(1)^n - \frac{1}{2}(-1)^n + \frac{4}{3}(-2)^n \qquad n \geq 0$$

6.2.5.4 The Difference Equation Representation

This was derived earlier from

$$H(z) = \frac{z^2}{z^2+3z+2} = \frac{Y(z)}{X(z)}$$

and will not be repeated here.

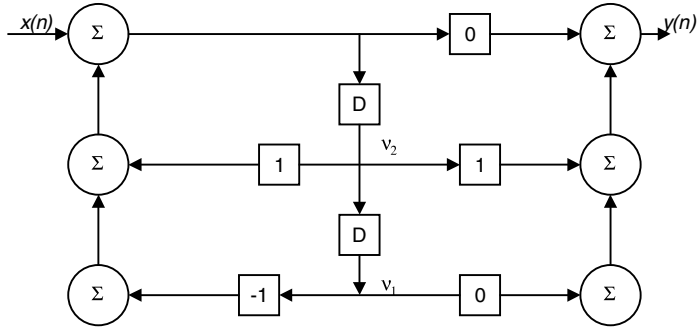

FIGURE 6.6 Block diagram for EOCE 6.1.

6.3 Some Insights: The Poles Considering Different Outputs within the Same System

Given a linear time-invariant system with multiple-inputs multiple-outputs, the poles of the system, regardless of what output you choose and what input you consider, will stay the same. This means that in such a system of many inputs and many outputs, the shape of the transients will be dominated by the fixed number poles of the system.

In summary, in a system of many inputs and many outputs, if you consider the transfer function relating any input to any output, the eigenvalues, or poles, will be the same. You may have different zeros for different transfer functions, but different transfer functions within the same system will have exactly the same poles.

6.4 End of Chapter Exercises

EOCE 6.1

Consider the following system as shown in Figure 6.6. Represent the system using all four representations and find the output $y(n)$ for $x(n) = u(n)$.

Solution

Using the state-space model

From the Figure 6.6, we see that

$$v_1(n+1) = v_2(n)$$

$$v_2(n+1) = x(n) + v_2(n) - v_1(n)$$

$$y(n) = v_2(n)$$

from which we get the state-space matrix system as

$$\mathbf{v}(n+1) = \begin{pmatrix} 0 & 1 \\ -1 & 1 \end{pmatrix}\mathbf{v}(n) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(n)$$

$$\mathbf{y}(n) = \begin{pmatrix} 0 & 1 \end{pmatrix}\mathbf{v}(n) + (0)\mathbf{x}(n)$$

The output $y(n)$ can be obtained with $x(n) = u(n)$ and with zero initial conditions with the help of the following MATLAB script:

```
n=0:25;
x = ones(size(n));
vin=[0 ; 0];
A = [0 1; -1 1];
B = [0; 1];
C =[0 1];
D = [0];
[y, v] = dlsim(A, B, C, D, x, vin);
stem (n, y), xlabel('n'), ylabel('The step response using the
 state-space model');
```

The transfer function representation

The transfer function $\mathbf{H}(z)$ is obtained as

$$\mathbf{H}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\,\mathbf{B} + \mathbf{D} = \begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} z & -1 \\ 1 & z-1 \end{pmatrix}^{-1}\begin{pmatrix} 0 \\ 1 \end{pmatrix} + (0)$$

By evaluating the inverse we get

$$\mathbf{H}(z) = \begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} \dfrac{z-1}{z^2-z+1} & \dfrac{1}{z^2-z+1} \\ \dfrac{-1}{z^2-z+1} & \dfrac{z}{z^2-z+1} \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \end{pmatrix}\begin{pmatrix} \dfrac{1}{z^2-z+1} \\ \dfrac{z}{z^2-z+1} \end{pmatrix}$$

Finally, the transfer function is obtained as

$$H(z) = \frac{z}{z^2 - z + 1}$$

Now we can use the MATLAB function `dstep` to find the step response and compare it with what we found using the state-spare representation. The following script is used.

```
num = [0 1 0] % numerator for H(z).
den = [1 -1 1]; % denominator for H(z).
n=0:25;
y=dstep(num, den,n);
stem(n,y); xlabel('n');
ylabel ('The step response using the transfer function model');
```

The difference equation representation

From

$$H(z) = \frac{z}{z^2 - z + 1} = \frac{Y(z)}{X(z)}$$

and by cross multiplication we have

$$Y(z)\left[z^2 - z + 1\right] = X(z)[z]$$

By inverse transforming the above equation we arrive at

$$y(n+2) - y(n+1) + y(n) = x(n+1)$$

If we substitute $n - 2$ for n, we will get

$$y(n) - y(n-1) + y(n-2) = x(n-1)$$

We then can use the MATLAB function `filter` to find the step response $y(n)$ as long as the coefficient of $y(n)$ in the difference equation is unity. The script is

```
b = [0 1 0] ; % input coefficients, x(n)
a = [1 -1 1]; % output coefficients, y(n)
n=0:25;
x=ones(size(n));
y=filter(b,a,x);
stem(n,y); xlabel('n');
ylabel ('The step response using the difference equation
 model');
```

The impulse response representation

We can take the inverse transform of

$$H(z) = \frac{z}{z^2 - z + 1}$$

to get $h(n)$. We can also use the MATLAB function `dimpulse` to find the impulse response $h(n)$ using the convolution equation

$$y(n) = h(n) * x(n)$$

with $x(n) = u(n)$. The next MATLAB script will do just that.

```
n=0:25;
x1=zeros(size(n));x1(1)=1;% impulse signal to find h(n)
x2=ones(size(n));% the input signal
b=[0 1 0];% input vector
a=[ 1 -1 1];% output vector
h=filter(b,a,x1);%the impulse response h(n)
y=conv(h,x2);%the output using convolution
stem(n,y(1:26));
title('The step response using the impulse response
 representation');
xlabel('n');
```

All the MATLAB scripts above for this first EOCE can be put together to produce one plot with multiple figures. By putting them together we have the script

```
%State space model
n=0:25;
x = ones(size(n));
vin=[0 ; 0];
A = [0 1; -1 1];
B = [0; 1];
C =[0 1];
D = [0];
[y, v] = dlsim(A, B, C, D, x, vin);
subplot(2,2,1);stem (n, y);
title('Step response: state-space model');
%Transfer function model
num = [0 1 0] % numerator for H(z).
den = [1 -1 1]; % denominator for H(z).
n=0:25;
y=dstep(num, den,n);
subplot(2,2,2);stem(n,y);
title('Step response: transfer function model');
%Difference equation model
b = [0 1 0] ; % input coefficients, x(n)
a = [1 -1 1]; % output coefficients, y(n)
n=0:25;
```

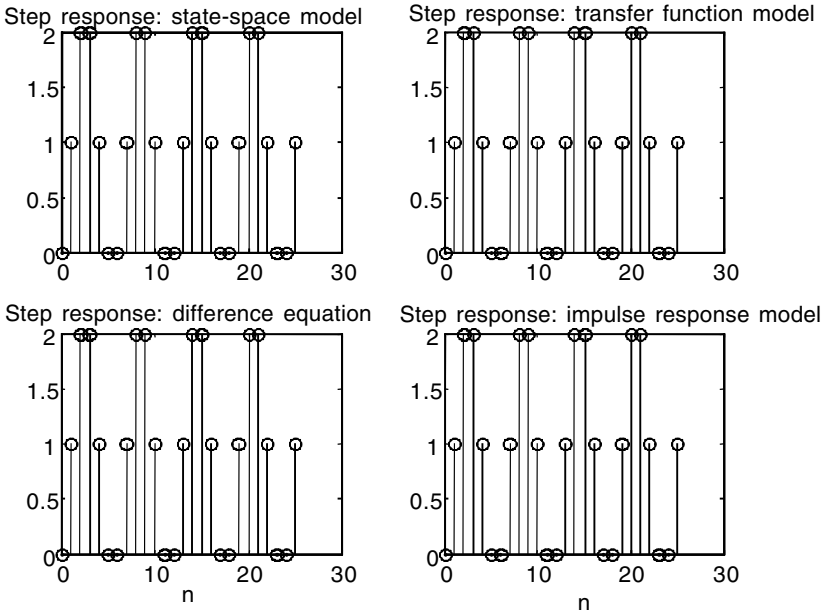

FIGURE 6.7 Plots for EOCE 6.1.

```
x=ones(size(n));
y=filter(b,a,x);
subplot(2,2,3);xlabel('n');stem(n,y);
title('Step response: difference equation');
%The impulse response model
n=0:25;
x1=zeros(size(n));x1(1)=1;% impulse signal to find h(n)
x2=ones(size(n));% the input signal
b=[0 1 0];% input vector
a=[ 1 -1 1];% output vector
h=filter(b,a,x1);%the impulse response h(n)
y=conv(h,x2);%the output using convolution
subplot(2,2,4);stem(n,y(1:26));
title('Step response: impulse response model');
xlabel('n');
```

The output using all the representations is shown in Figure 6.7.

EOCE 6.2

Consider the following difference equation representing the discrete system

$$y(n) + .81y(n-2) = x(n) - x(n-2)$$

Represent the system using all the models discussed. Find the impulse response of the system using all representation.

Solution

The difference equation representation

With $x(n) = \delta(n)$, we can find the impulse response using MATLAB as in the script

```
b = [1 0 -1] ; % input coefficients, x(n)
a = [1 0 .81]; % output coefficients, y(n)
n=0:25;
x=zeros(size(n));x(1)=1;%the impulse signal
y=filter(b,a,x);
stem(n,y); xlabel('n');
ylabel ('Impulse response: difference equation model');
```

The impulse response representation

We can use the MATLAB `filter` function directly on the difference equation to find the impulse response exactly as we did in the difference equation case. We can also use the MATLAB function `dimpulse` to find the impulse response as in the script

```
b = [1 0 -1] ; % input coefficients, x(n)
a = [1 0 .81]; % output coefficients, y(n)
n=0:25;
y=dimpulse(b,a,n);
stem(n,y); xlabel('n');
title('Impulse response: Impulse response model');
```

The transfer function representation

From the difference equation and by z-transforming term by term, we get

$$Y(z) + .81z^{-2}Y(z) = X(z) - z^{-2}X(z)$$

By multiplying all sides by z^2 we will get

$$z^2Y(z) + .81Y(z) = z^2X(z) - X(z)$$

Finally, the transfer function is given by

$$H(z) = \frac{z^2 - 1}{z^2 + .81}$$

$h(n)$ is the inverse z-transform of $H(z)$ and can be obtained using the filter function as in the script.

```
num = [1 0 -1] ; % input coefficients, x(n)
den = [1 0 .81]; % output coefficients, y(n)
n=0:25;
x=zeros(size(n));x(1)=1;%the impulse signal
y=filter(num,den,x);
stem(n,y); xlabel('n');
title('Impulse response: transfer function model');
```

The state-space representation

With

$$H(z) = \frac{z^2 - 1}{z^2 + .81}$$

the state-space system can be obtained with the following state matrices:

The system matrix $\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -.81 & 0 \end{pmatrix}$

The input matrix $\mathbf{B} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

The output matrix $\mathbf{C} = \begin{pmatrix} -1-(.81)(1) & 0-0(1) \end{pmatrix} = (-1.81 \quad 0)$

The transmission matrix $\mathbf{D} = (1)$

The output $y(n)$, the impulse response in this case, can be obtained as in the script.

```
n = 0:25;
x = zeros (size (n)); x(1)=1;
vin = [0 ; 0];
A = [0 1; -.81 0]
B = [0; 1]
C = [-1.81 0];
D = [1];
[y, v]= dlsim(A, B, C, D, x, vin);
stem (n, y); xlabel('n'); title('Impulse response:
 state-space');
```

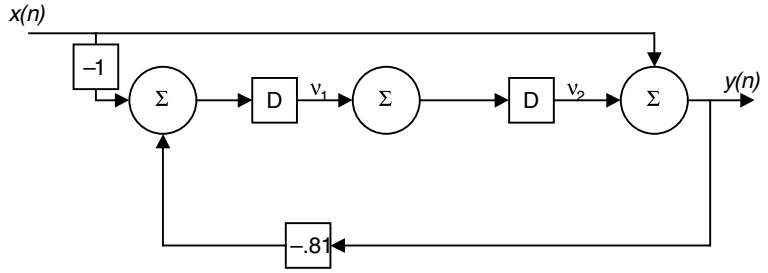

FIGURE 6.8 Block diagram for EOCE 6.2.

The block diagram representation

From

$$H(z) = \frac{z^2 - 1}{z^2 + .81}$$

we can obtain a different state-space representation. The block diagram is shown in Figure 6.8. From the block we have

$$v_1(n+1) = -x(n) - .81v_2(n) - .81x(n)$$
$$v_2(n) = v1(n)$$
$$y(n) = v_2(n) + x(n)$$

The state matrices are

$$\mathbf{A} = \begin{pmatrix} 0 & -.81 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} -1.81 \\ 0 \end{pmatrix}, \quad \mathbf{C} = (\,0 \quad 1), \quad \mathbf{D} = (1)$$

We will find the impulse response as in the following script:

```
%State space model
n=0:25;
x =zeros(size(n));x(1)=1;
vin=[0 ; 0];
A = [0 -.81; 1 0];
B = [-1.81; 0];
C =[0 1];
D = [1];
[y, v] = dlsim(A, B, C, D, x, vin);
stem (n, y);
title('Impulse response: state-space model');
```

The system is stable since the poles are at $z_{1,2} = \pm.9$. Again, we can combine the script in this exercise and produce one plot with multiple figures as in the following script.

```
%The difference equation representation
b = [1 0 -1] ; % input coefficients, x(n)
a = [1 0 .81]; % output coefficients, y(n)
n=0:25;
x=zeros(size(n));x(1)=1;%the impulse signal
y=filter(b,a,x);
subplot(2,2,1);stem(n,y);
title('From difference equation model');
ylabel('Impulse response');
%impulse response model
num = [1 0 -1] ; % input coefficients, x(n)
den = [1 0 .81]; % output coefficients, y(n)
n=0:25;
y=dimpulse(num,den,n);
subplot(2,2,2);stem(n,y);
title('From Impulse response model');
%State-space model
n = 0:25;
x = zeros (size (n)); x(1)=1;
vin = [0 ; 0];
A = [0 1; -.81 0]
B = [0; 1]
C = [-1.81 0];
D = [1];
[y, v]= dlsim(A, B, C, D, x, vin);
subplot(2,2,3);stem (n, y); xlabel('n');
title('State-space model #1');
ylabel('Impulse response');
%State-space model Different model
n=0:25;
x =zeros(size(n));x(1)=1;
vin=[0 ; 0];
A = [0 -.81; 1 0];
B = [-1.81; 0];
C =[0 1];
D = [1];
[y, v] = dlsim(A, B, C, D, x, vin);
subplot(2,2,4);stem (n, y);
title('State-space model #2');
```

The plots are shown in Figure 6.9.

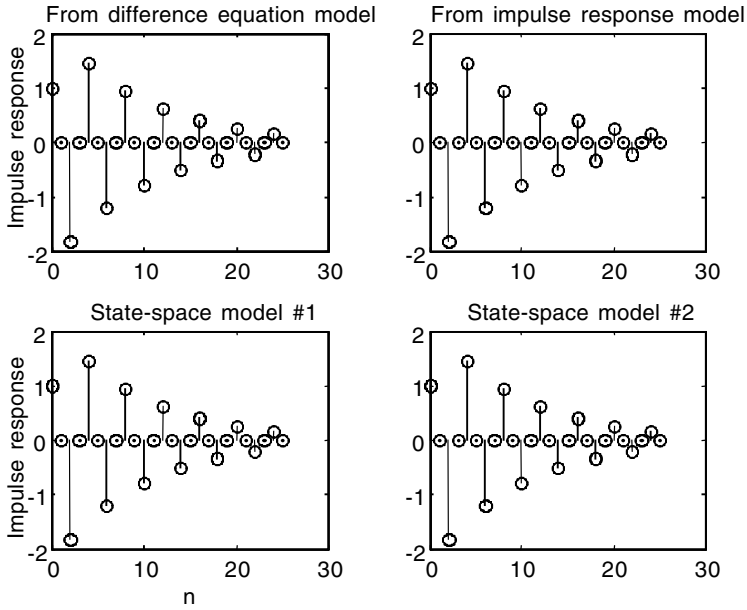

FIGURE 6.9 Plots for EOCE 6.2.

EOCE 6.3

Consider the state-space system

$$v_1(n+1) = -v_2(n) + x_1(n) + x_3(n)$$
$$v_2(n+1) = v_1(n) + x_2(n)$$
$$v_3(n+1) = v_2(n) + x_3(n)$$
$$y(n) = v_3(n)$$

Find the other representations and the impulse response.

Solution

The state-space representation

From the state and output equations we have, we can write the state-space matrices as

$$\mathbf{A} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{D} = (0)$$

The impulse response is obtained using the following MATLAB script.

```
n = 0:25;
x1=zeros(size(n));x1(1)=1;
x2=x1; x3=x2;
x=[x1' x2' x3'];
vin = [0; 0; 0];
A = [0 -1 0; 1 0 0; 0 1 0];
B = [1 0 1; 0 1 0; 0 0 1];
C = [0 0 1];
D = [0];
[y, v]= dlsim(A, B, C, D, x, vin);
stem (n, y); xlabel('n'); title('Impulse response:
 state-space');
```

The transfer function representation

We start with finding ($z\mathbf{I} - \mathbf{A}$) and write

$$(z\mathbf{I} - \mathbf{A}) = \begin{pmatrix} z & 0 & 0 \\ 0 & z & 0 \\ 0 & 0 & z \end{pmatrix} - \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} z & 1 & 0 \\ -1 & z & 0 \\ 0 & -1 & z \end{pmatrix}$$

The inverse of ($z\mathbf{I} - \mathbf{A}$) is

$$(z\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{z^2}{z(z^2+1)} & \dfrac{-z}{z(z^2+1)} & 0 \\ \dfrac{z}{z(z^2+1)} & \dfrac{z^2}{z(z^2+1)} & 0 \\ \dfrac{1}{z(z^2+1)} & \dfrac{z}{z(z^2+1)} & \dfrac{z^2+1}{z(z^2+1)} \end{pmatrix}$$

The transfer function $\mathbf{H}(z)$ is then

$$\mathbf{H}(z) = (0 \quad 0 \quad 1)\,(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$$

$$= \begin{pmatrix} \dfrac{1}{z(z^2+1)} & \dfrac{z}{z(z^2+1)} & \dfrac{z^2}{z(z^2+1)} \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

or $\mathbf{H}(z)$ is finally

$$\mathbf{H}(z)=\left(\frac{1}{z\left(z^2+1\right)} \quad \frac{z}{z\left(z^2+1\right)} \quad \frac{1}{z\left(z^2+1\right)}+\frac{z^2+1}{z\left(z^2+1\right)}\right)$$

We can also obtain $\mathbf{H}(z)$ using MATLAB by writing the next MATLAB script.

```
A = [0 -1 0; 1 0 0; 0 1 0];
B = [1 0 1; 0 1 0; 0 0 1];
C = [0 0 1];
D = [0 0 0];
[n1 d]=ss2tf(A, B, C, D, 1)%column 1 coefficients
[n2 d]=ss2tf(A, B, C, D, 2)%column 2 coefficients
[n3 d]=ss2tf(A, B, C, D, 3)%column 3 coefficients
```

The output is

```
n1 = 0   0.0000   0.0000   1.0000
d = 1   0   1   0
n2 = 0   0.0000   1.0000   0.0000
d = 1   0   1   0
n3 = 0   1.0000   –0.0000   2.0000
d = 1   0   1   0
```

With $X_1(z)=X_2(z)=X_3(z)=1$, we can find the impulse response in the z-domain as

$$\mathbf{Y}(z)=\left(\frac{1}{z\left(z^2+1\right)} \quad \frac{z}{z\left(z^2+1\right)} \quad \frac{1}{z\left(z^2+1\right)}+\frac{z^2+1}{z\left(z^2+1\right)}\right)\begin{pmatrix}1\\1\\1\end{pmatrix}$$

$$\mathbf{Y}(z)=\left(\frac{1}{z\left(z^2+1\right)}+\frac{z}{z\left(z^2+1\right)}+\frac{z^2+2}{z\left(z^2+1\right)}\right)$$

or

$$\mathbf{Y}(z)=\frac{z^2+z+3}{z^3+z}$$

At this point we can do partial fraction expansion on $\mathbf{Y}(z)/z$ and get $y(n)$. However, we can use MATLAB and find the impulse response as in the script.

```
num = [0 1 1 3] ; % input coefficients, x(n)
den = [1 0 1 0]; % output coefficients, y(n)
```

```
n=0:25;
y=dimpulse(num,den,n);
stem(n,y); xlabel('n');
title('Impulse response: transfer function model');
```

The impulse response representation

It looks trivial but we can use MATLAB to find $h(n)$ from $H(z)$ then use MATLAB again to find the impulse response, the convolution between $h(n)$ and $x(n)$, the impulse input, just to get more practice with MATLAB. Remember that $h(n)$ convolved with the impulse signal is just $h(n)$.

```
n=0:25;
x=zeros(size(n));x(1)=1;% impulse signal to find h(n)
b=[0 1 1 3];% input vector
a=[ 1 0 1 0];% output vector
h=filter(b,a,x1);%the impulse response h(n)
y=conv(h,x);%the output using convolution
stem(n,y(1:26));title('Impulse response: impulse response
 model');
xlabel('n');
```

The difference equation representation

We can z-transform the state and output equations to get

$$zV_1(z) = -V_2(z) + X_1(z) + X_3(z)$$

$$zV_2(z) = V_1(z) + X_2(z)$$

$$zV_3(z) = V_2(z) + X_3(z)$$

and the output $y(n)$ becomes

$$Y(z) = V_3(z)$$

Solving the above equations we get

$$V_2(z) = \frac{z^{-2}X_1(z) + z^{-2}X_3(z) + z^{-1}X_2(z)}{1 + z^{-2}}$$

If we substitute $V_2(z)$ in the last of the state equations, we arrive at

$$V_3(z) = \frac{z^{-3}X_1(z) + z^{-3}X_3(z) + z^{-2}X_2(z) + z^{-1}X_3(z) + z^{-3}X_3(z)}{1 + z^{-2}}$$

If we multiply the numerator and the denominator to the right in the above equation by z^3 we will get

$$V_3(z) = \frac{X_1(z) + X_3(z) + zX_2(z) + z^2X_3(z) + X_3(z)}{z^3 + z}$$

or finally

$$Y(z) = V_3(z) = \frac{X_1(z)}{z\left(z^2+1\right)} + \frac{zX_2(z)}{z\left(z^2+1\right)} + \frac{X_3\left(2+z^2\right)}{z\left(z^2+1\right)}$$

We can inverse transform the last equation and get the difference equation representation. We will have

$$y(n+3) + y(n+1) = x_1(n) + x_2(n+1) + 2x_3(n) + x_3(n+2)$$

We can at this point use superposition and find the impulse response for

$$\frac{Y(z)}{X_1(z)} = \frac{1}{z\left(z^2+1\right)}, \quad \frac{Y(z)}{X_2(z)} = \frac{z}{z\left(z^2+1\right)}, \text{ and } \frac{Y(z)}{X_3(z)} = \frac{z^2+2}{z\left(z^2+1\right)}$$

individually, then sum up the responses to get the total impulse response. We will do that using the following MATLAB script.

```
n=0:25;x=zeros(size(n)); x(1)=1;
n1=[ 0 0 0 1]; n2=[0 0 1 0];n3=[ 0 1 0 2];
d=[ 1 0 1 0];
y1=filter(n1,d,x); y2=filter(n2,d,x); y3=filter(n3,d,x);
y=y1+y2+y3;
stem(n,y); title('Impulse response: difference equation
 model');
```

The block diagram representation

The block diagram is implemented directly from the state and the output equation, and is shown in Figure 6.10. We can consider all the MATLAB scripts that we wrote for this exercise and generate one plot with multiple figures illustrating the impulse response using all representations. The script is

```
%The state-space model
n = 0:25;
x1=zeros(size(n));x1(1)=1;
x2=x1; x3=x2;
x=[x1' x2' x3'];
vin = [0 ; 0; 0];
```

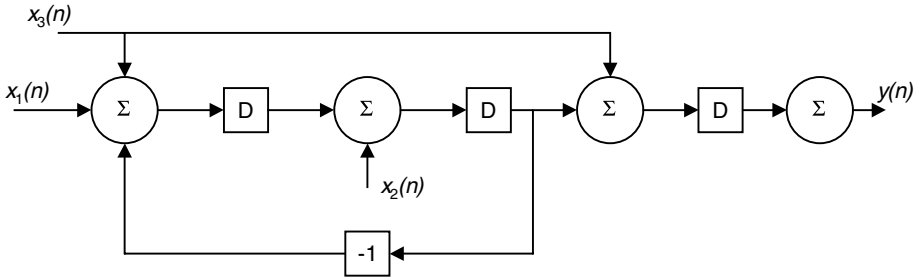

FIGURE 6.10 Block diagram for EOCE 6.3.

```
A = [0 -1 0; 1 0 0; 0 1 0];
B = [1 0 1; 0 1 0; 0 0 1];
C = [0 0 1];
D = [0];
[y, v]= dlsim(A, B, C, D, x, vin);
subplot(2,2,1);
stem (n, y); title('From state-space');
ylabel('Impulse response');
%The transfer function model
num = [0 1 1 3] ; % input coefficients, x(n)
den = [1 0 1 0]; % output coefficients, y(n)
n=0:25;
y=dimpulse(num,den,n);
subplot(2,2,2);stem(n,y); xlabel('n');
title('From transfer function model');
%The impulse response model
n=0:25;
x=zeros(size(n));x(1)=1;% impulse signal to find h(n)
b=[0 1 1 3];% input vector
a=[ 1 0 1 0];% output vector
h=filter(b,a,x1);%the impulse response h(n)
y=conv(h,x);%the output using convolution
subplot(2,2,3);
stem(n,y(1:26));title('From impulse response model');
xlabel('n');ylabel('Impulse response');
n=0:25;x=zeros(size(n)); x(1)=1;
n1=[ 0 0 0 1]; n2=[0 0 1 0];n3=[ 0 1 0 2];
d=[ 1 0 1 0];
y1=filter(n1,d,x); y2=filter(n2,d,x); y3=filter(n3,d,x);
y=y1+y2+y3; subplot(2,2,4);
stem(n,y); title('From difference equation model');
xlabel('n');
```

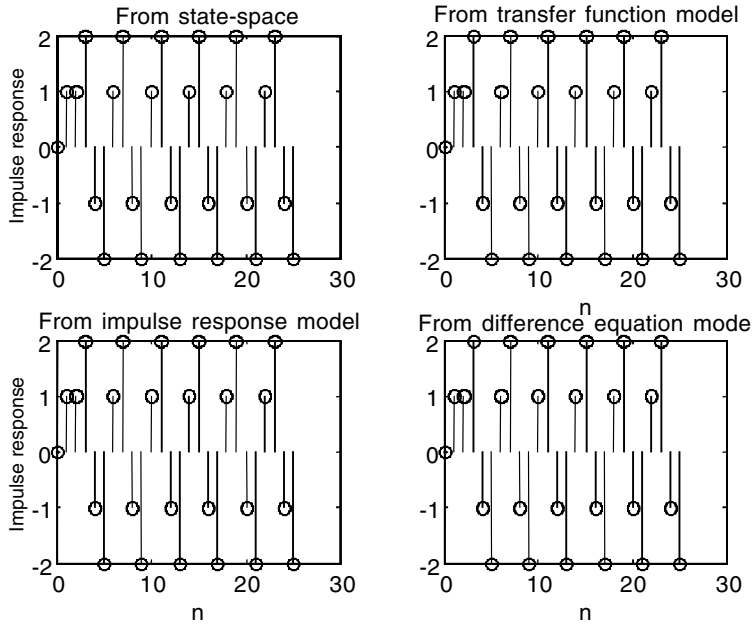

FIGURE 6.11 Plots for EOCE 6.3.

The plot is shown in Figure 6.11.

6.5 End of Chapter Problems

EOCP 6.1

Consider the following systems

1. $y(n) + y(n-2) = 3x(n)$
2. $y(n) + y(n-3) = x(n)$
3. $y(n) + 2y(n-1) + 2y(n-2) = x(n-1)$
4. $y_1(n) - 2y_2(n-1) = 0$ and $y_2(n) - 3\,y_1(n-1) = x(n)$
5. $y(n) = x(n) + 2x(n-1) + 3x(n-3) + 4x(n-4)$

a) Find the other four representations and indicate if the system is stable or not.
b) Find the step and the impulse responses using the other representation.

EOCP 6.2

Consider the systems

1. $H(z) = \dfrac{z}{z(z+1)}$
2. $H(z) = \dfrac{z^2+4}{z(z^2+1)}$
3. $H(z) = \dfrac{z+1}{z^3-1}$
4. $H(z) = \begin{pmatrix} \dfrac{z}{z(z+1)} \\ \dfrac{z^2}{z(z+1)} \end{pmatrix}$
5. $H(z) = \begin{pmatrix} \dfrac{z}{z^2+z+1} & \dfrac{1}{z^2+z+1} & \dfrac{z+1}{z^2+z+1} \end{pmatrix}$

a) Find the other four representations for the first three systems.

b) Find the step response for the last two systems.

EOCP 6.3

Consider the systems represented by the block diagrams as shown in Figures 6.12 through 6.16.

1. Give the other four representations for these systems.
2. Use state-space method to find the step response for the last two systems.
3. Find the impulse response for the first system using the state-space representations.

EOCP 6.4

Consider the systems

1. $h(n) = (.7)^n\, u(n)$
2. $h(n) = (.7)^n\, u(n) + (.8)^n\, u(n)$
3. $h(n) = u(n-1) - u(n-2)$

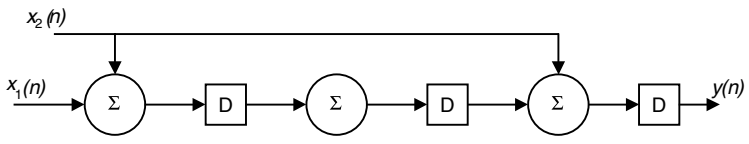

FIGURE 6.12 Block for EOCP 6.3.

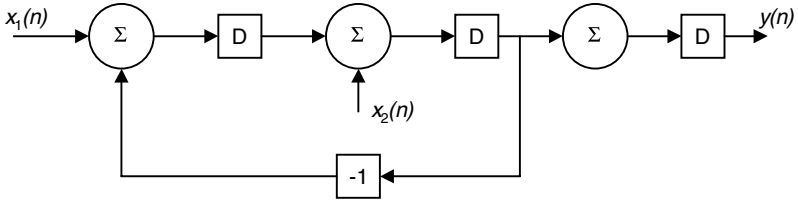

FIGURE 6.13 Block diagram for EOCP 6.3.

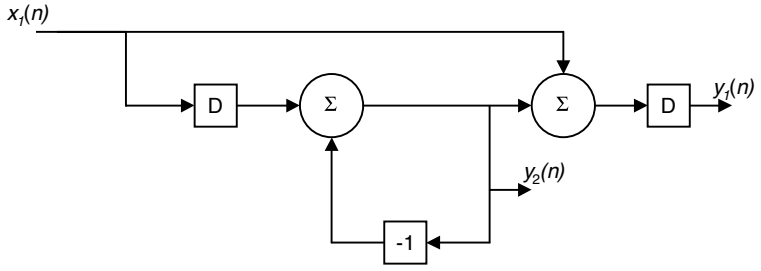

FIGURE 6.14 Block diagram for EOCP 6.3.

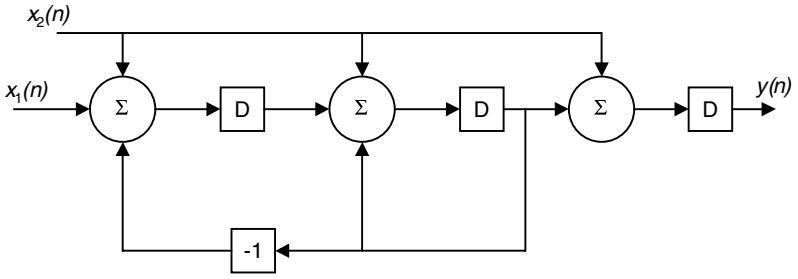

FIGURE 6.15 Block diagram for EOCP 6.3.

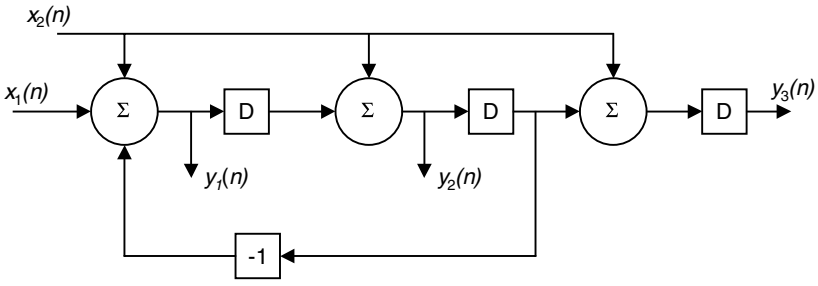

FIGURE 6.16 Block diagram for EOCP 6.3.

4. $h(n) = n(.6)^n\, u(n)$
5. $\mathrm{h}(n) = \begin{pmatrix} (.5)^n u(n) \\ \\ (.3)^n u(n) \end{pmatrix}$

1. Find the other representation.
2. Find the step response for system 5.
3. Find the steady-state response for system 2.

EOCP 6.5

Consider the systems

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -1 & -2 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 0 & 1 \end{pmatrix}, \quad \mathbf{D} = (0)$$

$$\mathbf{A} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -3 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{D} = (0)$$

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & -2 & -3 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}, \quad \mathbf{D} = (1)$$

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & -2 \\ 1 & 0 & -3 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{D} = (1)$$

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & -2 & -1 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad \mathbf{D} = (0)$$

1. Find all other representations.
2. Find $\mathbf{H}(z)$ for the last three systems.
3. What are the poles for the last three systems?

7

The Discrete Fourier Transform and Discrete Systems

7.1 Introduction

We have two types of linear systems: the continuous linear system and the discrete linear system. The Fourier transform of the continuous signal $x(t)$ is

$$X(w) = \int_{-\infty}^{+\infty} x(t)e^{-jwt}dt \tag{7.1}$$

By observing the signal $x(t)$ in the frequency domain, we are able to see all the frequencies that $x(t)$ contains, and based on that we can further process the signal $x(t)$ and alter its frequency components in any way we wish.

But the Fourier transform equation given above is not suitable for computer evaluation since the bounds on the integral are infinite. By *discretizing* the signal $x(t)$ and obtaining the discrete signal $x(n)$, we can Fourier transform the signal $x(n)$ into the frequency domain in order to see its frequency components hoping that we can utilize the computer in all calculations. The Fourier transform of the signal $x(n)$ is

$$X(e^{j\theta}) = \sum_{-\infty}^{+\infty} x(n)e^{-j\theta n} \tag{7.2}$$

where θ is the digital frequency and is related to the continuous frequency by the equation

$$\theta = wT_s \tag{7.3}$$

where T_s is the sampling period used to sample $x(t)$ to get $x(n)$. $x(n)$ now is discrete but is of infinite duration and, again, it is not possible to evaluate $X(e^{j\theta})$ on the computer. This problem will be discussed next.

7.2 The Discrete Fourier Transform and the Finite-Duration Discrete Signals

Continuing with the discussion above, let us assume that $x(n)$ is zero for all the integer n values less than zero, and for n greater than or equal to some integer value N. In other words suppose $x(n) = 0$ for $n < 0$ and $n \geq N$. Then we can define the N-points Fourier transform for the truncated $x(n)$ as

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi nk}{N}} \qquad k = 0, 1, \ldots N-1 \tag{7.4}$$

The above equation is the discrete Fourier transform (DFT) of the finite duration signal $x(n)$.

In this defining equation,

1. $x(n)$ in the truncated signal that is zero for $n < 0$ and $n \geq N$.
2. N is the length of the truncated signal $x(n)$.
3. n is the index for the samples in $x(n)$.
4. k is the frequency index for the DFT.

We can see in the above equation for the DFT that $X(k)$ is finite, $x(n)$ is finite and the number of points in $x(n)$ is the same as the number of points in $X(k)$. Now it is easy to implement the DFT equation using the computer.

To go back to the discrete time signal $x(n)$ from the $X(k)$ values, we use the inverse relation

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{\frac{j2\pi nk}{N}} \qquad n = 0, 1, \ldots, N-1 \tag{7.5}$$

Example 7.1

Consider the discrete signal given by

$$x(0) = 1 \text{ and } x(1) = 0$$

Find the DFT of $x(n)$ and do the inverse DFT to obtain $x(n)$ from $X(k)$ for $n = 0, 1$.

Solution

In this example, N is 2 and $X(k)$ is

$$X(k) = \sum_{n=0}^{1} x(n) e^{\frac{-j2\pi nk}{N}} = \sum_{n=0}^{1} x(n) \left(W_n\right)^{nk}$$

Before we evaluate $X(k)$, let us look at the defining equation for $X(k)$. We have substituted $W_N = e^{\frac{-j2\pi}{N}}$ in Equation (7.5) where

$$W_N^{nk} = e^{\frac{-j2\pi nk}{N}} = \left(W_N\right)^{nk}$$

Based on this relation, in this example we have

$$W_N = W_2 = e^{-j\frac{2\pi}{2}} = e^{-j\pi} = \cos(\pi) - j\sin(\pi) = -1$$

Now by expanding the $X(k)$ equation we get

$$X(0) = x(0)(-1)^0 + x(1)(-1)^0 = 1$$

$$X(1) = x(0)(-1)^0 + x(1)(-1)^1 = 1$$

To calculate $x(n)$ from $X(k)$ we use the inverse relation

$$x(n) = \frac{1}{N}\sum_{k=0}^{1} X(k)e^{\frac{j2\pi nk}{N}}$$

to get

$$x(0) = \frac{1}{2}X(0)e^{j\pi 0} + \frac{1}{2}X(1)e^{j\pi 0} = \frac{1}{2}(1) + \frac{1}{2}(1) = 1$$

and

$$x(1) = \frac{1}{2}X(0)e^{j\pi 0} + \frac{1}{2}X(1)e^{j\pi} = \frac{1}{2}(1) + \frac{1}{2}(1)(-1) = 0$$

7.3 Properties of the Discrete Fourier Transform

Following are some of the characteristics that the DFT possesses.

7.3.1 How Does the Defining Equation Work?

If we look at the defining equation in its trigonometric form we have

$$X(k) = \sum_{n=0}^{N-1} x(n)\left[\cos\left(\frac{2\pi nk}{N}\right) - j\sin\left(\frac{2\pi nk}{N}\right)\right] \tag{7.6}$$

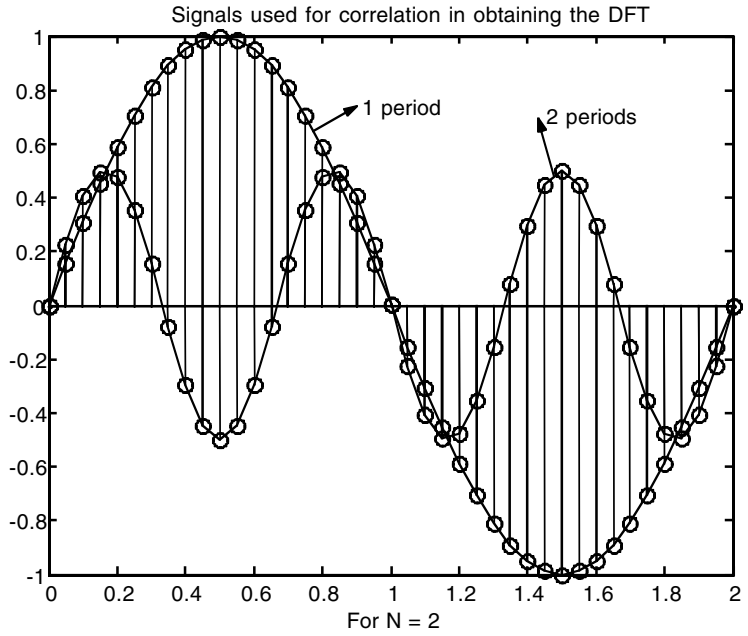

FIGURE 7.1 Correlation signals for the DFT.

You can see that each component of $X(k)$ is a correlation of all the samples in $x(n)$ with cosine and sine signals that have k complete periods in the N samples interval. This can be seen in the example that we considered earlier, Example 7.1, where with $N = 2$

$$X(k) = \sum_{n=0}^{1} x(n)\left[\cos(\pi nk) - j\sin(\pi nk)\right]$$

If we plot $\cos(\pi nk)$ or $\sin(\pi nk)$ on a fine grid, we will observe a complete cycle for $k = 1$ and 2 cycles for $k = 2$ in the $N = 2$ interval. This can be seen in Figure 7.1.

7.3.2 The DFT Symmetry

For a real $x(n)$ signal, the DFT is symmetrical around the point $k = N/2$. The magnitude of $X(k)$ will have even symmetry and the phase of $X(k)$ will have odd symmetry. To see this we can show that

$$|X(k)| = |X(N-k)|$$

and that the phase of $X(k)$ is the negative of the phase of $X(N - k)$. The magnitude of $X(k)$ is

$$|X(k)| = \left| \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}} \right|$$

and the magnitude of $X(N-k)$ is

$$|X(N-k)| = \left| \sum_{n=0}^{N-1} x(n) e^{-j2\pi n\frac{(N-k)}{N}} \right| = \left| \sum_{n=0}^{N-1} x(n) e^{j\frac{2\pi nk}{N}} \right|$$

since $e^{-j\frac{2\pi nN}{N}} = 1$.

To look at this in a different way we have

$$X(k) = \sum_{n=0}^{N-1} x(n) \left[\cos\left(\frac{2\pi nk}{N}\right) - j \sin\left(\frac{2\pi nk}{N}\right) \right]$$

But

$$|X(k)| = |X(N-k)| \text{ if } \text{Real}\left[X(k)\right] = \text{Real}\left[X(N-k)\right].$$

$$\text{Real}\left[X(k)\right] = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi nk}{N}\right)$$

and

$$\text{Real}\left[X(N-k)\right] = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi n(N-k)}{N}\right)$$

We also know that

$$\cos\left(2\pi nk - \frac{2\pi nk}{N}\right) = \cos\left(\frac{-2\pi nk}{N}\right) = \cos\left(\frac{2\pi nk}{N}\right)$$

Thus

$$\text{Real}\left[X(k)\right] = \text{Real}\left[X(N-k)\right]$$

and consequently

$$|X(k)| = |X(N-k)|$$

To show that

$$\angle X(k) = -\angle X(N-K)$$

we need to show that the imaginary

$$\text{Imag}\left[X(k)\right] = -\text{Imag}\left[\left(X(N-k)\right)\right]$$

But

$$\text{Imag}\left[X(k)\right] = -\sum_{n=0}^{N-1} x(n)\sin\left(\frac{2\pi nk}{N}\right)$$

and

$$\text{Imag}\left[X(N-k)\right] = -\sum_{n=0}^{N-1} x(n)\sin\left(2\pi n\frac{(N-k)}{N}\right)$$

But again

$$\sin\left(2\pi n + -\frac{2\pi nk}{N}\right) = \sin\left(\frac{-2\pi nk}{N}\right) = -\sin\left(\frac{2\pi nk}{N}\right)$$

Thus

$$\text{Imag}\left[X(N-k)\right] = \sum_{n=0}^{N-1} x(n)\sin\left(\frac{2\pi nk}{N}\right)$$

Therefore,

$$\text{Imag}\left[X(k)\right] = -\text{Imag}\left[\left(X(N-k)\right)\right]$$

and thus

$$\angle X(k) = -\angle X(N-k)$$

7.3.3 The DFT Linearity

Given the signal

$$x(n) = \alpha x_1(n) + \beta x_2(n)$$

the DFT of $x(n)$ is then

$$X(k) = \alpha X_1(k) + \beta X_2(k)$$

To see this result, we have

$$X(k) = \sum_{n=0}^{N-1} \left(\alpha x_1(n) + \beta x_2(n)\right) e^{-j\frac{2\pi nk}{N}} = \alpha \sum_{n=0}^{N-1} x_1(n) e^{\frac{-j2\pi nk}{N}} + \beta \sum_{n=0}^{N-1} x_2(n) e^{\frac{-j2\pi nk}{N}}$$

Finally we have

$$X(k) = \alpha X_1(k) + \beta X_2(k)$$

This linearity property is very important because we usually have signals that have more than one frequency component.

7.3.4 The Magnitude of the DFT

If we are calculating the DFT of the signal $x(n)$ that was obtained by taking N samples of a real sinusoid of amplitude M_r then the magnitude of the DFT is

$$|X(k)| = \frac{M_r N}{2}$$

If the N samples in $x(n)$ were obtained by sampling a complex sinusoid of the form

$$x(t) = M_c\left[\cos(wt) + j\sin(wt)\right] = M_c e^{jwt}$$

then

$$|X(k)| = M_c N$$

If, however, $x(n)$ has a dc component M_d, then

$$|X(k)| = M_d N$$

In the relation above, the input to the DFT system should have a sinusoidal component that makes an integer number of periods over the N samples.

7.3.5 What Does *k* in *X*(*k*), the DFT, Mean?

After we calculate the DFT of the signal $x(n)$ which is usually samples of the continuous signal $x(t)$, we may be interested in the value of the frequency when the magnitude of $X(k)$ is maximum. The plot of $|X(k)|$ is usually drawn vs. the frequency index k. The distance between the successive values of k is given by the frequency resolution f_s/N where f_s is the sampling frequency. If the maximum value of $|X(k)|$ is at $k = k_m$, for example, then the frequency at which $|X(k)|$ is maximum is $k_m f_s/N$ Hz. If we let $\Delta f = f_s/N$ be the frequency resolution, then we have

$$\Delta f = \frac{f_s}{N} = \frac{1}{NT_s} = \frac{1}{T_r} \tag{7.7}$$

where T_r is the time interval along which the samples are taken.

TABLE 7.1

Some Properties of the DFT

Linearity	$\alpha x_1(n) + \beta x_2(n) \leftrightarrow \alpha X_1(k) + \beta X_2(k)$
Time shifting	$x(n-m) \leftrightarrow X(k)e^{-j\frac{2\pi km}{N}}$
Frequency shifting	$x(n)e^{\frac{j2\pi nm}{N}} \leftrightarrow X(k-m)$
Modulation	$x_1(n)x_2(n) \leftrightarrow \frac{1}{N}X_1(k) \otimes X_2(k)$
Circular convolution	$x_1(n) \otimes x_2(n) \leftrightarrow X_1(k)X_2(k)$
Parseval's theorem	$\sum_{k=0}^{N-1} \lvert x(n)\rvert^2 = \frac{1}{N}\sum_{k=0}^{N-1} \lvert X(k)\rvert^2$
Duality	$\frac{1}{N}X(n) \leftrightarrow x(-k)$

$\otimes$ indicates circular convolution

Example 7.2

Consider the magnitude DFT of the signal $x(n)$ that was obtained by sampling a continuous sinusoid at f_s = 1000 Hz as shown in Figure 7.2.

1. What is the frequency at which $|X(k)|$ is maximum?
2. What is the frequency at which $|X(k)|$ is minimum?
3. If only the value at $k = 2$ is known for $|X(k)|$, can you find $|X(3)|$?

Solution

1. From the plot we see that $|X(k)|$ is maximum at $k = 0$ and $k = 4$ and that $N = 5$. Thus we have $k_m f_s/N = 0(1000)/5 = 0$ Hz and $4(1000)/5 = 800$ Hz as the two frequencies.

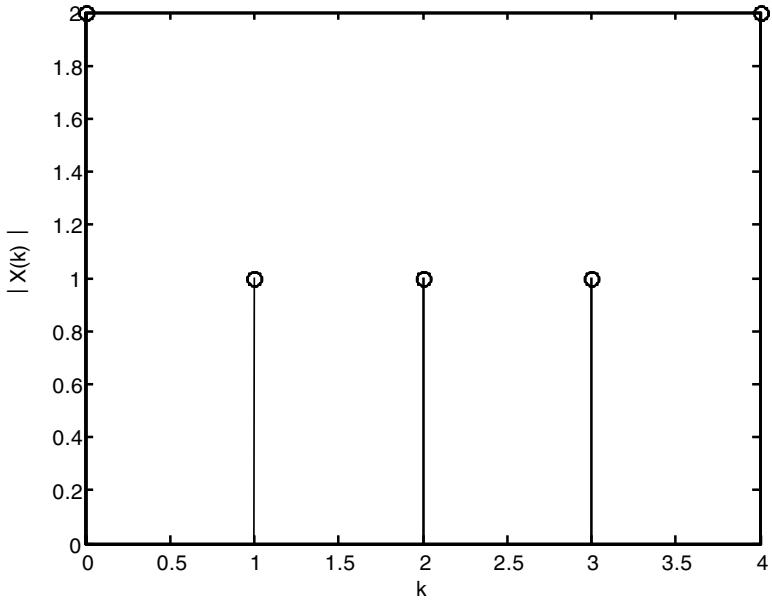

FIGURE 7.2 Signal for Example 7.2.

2. The minimum magnitude is at $k = 1$, 2 and 3. Thus

$$\frac{1(1000)}{5} = 200\ ,\ \frac{2(1000)}{5} = 400 \text{ and } \frac{3(1000)}{5} = 600 \text{ Hz}$$

   are the frequencies.
3. With only $X(2)$ known we have

$$\frac{N}{2} = \frac{5}{2} = 2.5$$

   Thus from the symmetry property we would know $X(3) = X(2) = 1$.

7.4 The Relation the DFT Has with the Fourier Transform of Discrete Signals, the *z*-Transform and the Continuous Fourier Transform

7.4.1 The DFT and the Fourier Transform of $x(n)$

Given $x(n)$ for $-\infty \le n \le +\infty$. The Fourier transform is defined as

$$X\left(e^{j\theta}\right) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n}$$

where $\theta = wT_s$, w is the continuous radian frequency and T_s is the sampling interval used to sample $x(t)$ to get $x(n)$. If the signal $x(n)$ is truncated and is limited to the lower $n = 0$ and the upper $n = N - 1$ interval where N is some integer value, then

$$X\left(e^{j\theta}\right) = \sum_{n=0}^{N-1} x(n)e^{-j\theta n}$$

Now if we let

$$\theta = \frac{2\pi k}{N}$$

we will have

$$X\left(e^{\frac{j2\pi k}{N}}\right) = \sum_{n=0}^{N-1} x(n)e^{\frac{-j2\pi kn}{N}}$$

But the term $\frac{2\pi k}{N}$ is a function of k only since N is known. Thus we can write

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{\frac{-j2\pi kn}{N}} = X(e^{j\theta})\Big|_{\theta=\frac{2\pi k}{N}} \tag{7.8}$$

The above equation is nothing but the DFT of the finite signal $x(n)$ defined for $n = 0, 1,\ldots, N - 1$. Therefore, from this development you can see that if $x(n)$ is limited to the interval $n = 0, 1,\ldots, N - 1$, and $X(e^{j\theta})$ is sampled starting at values of $\theta = 0$ and then regularly spaced along the unit circle by $2\pi k/N$, the DFT will be the sampled version of $X(e^{j\theta})$.

7.4.2 The DFT and the *z*-Transform of *x*(*n*)

The z-transform for a finite length signal $x(n)$ is

$$X(z) = \sum_{n=0}^{N-1} x(n)z^{-n}$$

where N is the number of samples in $x(n)$. If we set $z = e^{\frac{j2\pi k}{N}}$ then

$$X\left(e^{\frac{j2\pi k}{N}}\right) = \sum_{n=0}^{N-1} x(n)e^{\frac{-j2\pi nk}{N}}$$

But again, the term $2\pi k/N$ is a function of k only since N is known. Thus we can write

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi kn}{N}} = X(z)\Big|_{z=\frac{j2\pi k}{N}} \tag{7.9}$$

7.4.3 The DFT and the Continuous Fourier Transform of $x(t)$

Consider the signal $x(t)$ that is assumed to be zero for $t < 0$. The Fourier transform of $x(t)$ is

$$X_c(w) = \int_0^{\infty} x(t) e^{-jwt} dt$$

Let T_s be a small positive number. Then the above equation can be approximated as

$$X_c(w) = \sum_{n=0}^{\infty} \int_{nT_s}^{nT_s+T_s} x(t) e^{-jwt} dt = \int_0^{T_s} x(t) e^{-jwt} dt + \int_{T_s}^{2T_s} x(t) e^{-jwt} dt + \ldots$$

In the interval $nT_s \le t \le nT_s + T_s$, if T_s is chosen small enough such that $x(t)$ in this interval will be considered constant, then

$$X_c(w) = \sum_{n=0}^{\infty} \int_{nT_s}^{nT_s+T_s} e^{-jwt} dt \, x\left(nT_s\right)$$

With

$$\int_{nT_s}^{nT_s+T_s} e^{-jwt} dt = \frac{e^{-jwnT_s} - e^{-(1+n)jwT_s}}{jw}$$

we have

$$X_c(w) = \frac{1 - e^{-jwT_s}}{jw} \sum_{n=0}^{\infty} e^{-jwnT_s} x\left(nT_s\right)$$

If $x(nT_s)$ is very small for $n \ge N$, then the continuous Fourier transform can be simplified further as

$$X_c(w) = \frac{1 - e^{-jwT_s}}{jw} \sum_{n=0}^{N-1} e^{-jwnT_s} x(nT_s)$$

If we set $w = 2\pi k/NT_s$ we will have

$$X_c\left(\frac{2\pi k}{NT_s}\right) = \frac{1 - e^{-jwT_s}}{j\left(\frac{2\pi k}{NT_s}\right)} \sum_{n=0}^{N-1} e^{-j\frac{2\pi k}{N}n} x(nT_s)$$

But

$$\sum_{n=0}^{N-1} e^{-j\frac{2\pi nk}{N}} x(nT_s) = X(k)$$

which is the DFT of $x(nT_s)$. Thus the continuous Fourier transform can be approximated at $2\pi/NT_s$ k points where $k = 0, 1, \ldots, N - 1$. Finally we write

$$X_c\left(\frac{2\pi k}{NT_s}\right) = \frac{1 - e^{-jwT_s}}{j\left(\frac{2\pi k}{NT_s}\right)} X(k) \tag{7.10}$$

Example 7.3

Consider the signal $x(n)$ where $x(0) = 1$, $x(1) = 0$ and $x(2) = 1$. Use the z-transform method to find $X(k)$.

Solution

This signal can easily be put in the z-domain. It is

$$X(z) = 1z^{-0} + 0z^{-1} + 1z^{-2} = 1 + z^{-2}$$

With $z = e^{j\theta}$ we have

$$X\left(e^{j\theta}\right) = 1 + e^{-2j\theta}$$

But

$$X(k) = X\left(e^{j\theta}\right)\Big|_{\theta=\frac{2\pi k}{N}}$$

With $N = 3$ as is the case in this example we have

$$X(k) = 1 + e^{-2j\left(\frac{2\pi k}{3}\right)}$$

with

$$X(0) = 1 + e^{-2j(0)} = 2$$

$$X(1) = 1 + e^{-2j\left(\frac{2\pi}{3}\right)}$$

$$X(2) = 1 + e^{-2j\left(\frac{2\pi 2}{3}\right)} = 1 + e^{-j\left(\frac{8\pi}{3}\right)}$$

Using the defining equation of the DFT we have

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi nk}{N}} = X(k) = \sum_{n=0}^{2} x(n)\left(W_N\right)^{nk}$$

with

$$e^{\frac{-j2\pi}{N}} = e^{\frac{-j\pi 2}{3}}$$

With this simplification we can get the first few samples of the DFT.

$$X(0) = x(0)e^{\frac{-j2\pi}{3}(0)} + x(1)e^{\frac{-j2\pi}{3}(0)} + x(2)e^{\frac{-j2\pi}{3}(0)} = 2$$

$$X(1) = x(0)e^{\frac{-j2\pi}{3}(0)} + x(1)e^{\frac{-j2\pi}{3}(1)} + x(2)e^{\frac{-j2\pi}{3}(2)} = 1 + e^{\frac{-j4\pi}{3}}$$

$$X(2) = x(0)e^{\frac{-j2\pi}{3}(0)} + x(1)e^{\frac{-j2\pi}{3}(2)} + x(2)e^{\frac{-j2\pi}{3}(4)} = 1 + e^{\frac{-j8\pi}{3}}$$

7.5 Numerical Computation of the DFT

The DFT is a computer program that can be used to transform the signal $x(n)$ defined for $n = 0, 1, \ldots, N-1$, where N is the number of samples in $x(n)$, to $X(k)$, a set of N values defined for the frequency index $k = 0, 1, \ldots, N-1$. In other words, the DFT can be thought of as a system (a computer program) whose input is $x(n)$ and whose output is $X(k)$.

To implement the DFT equation on a digital computer, MATLAB will be used. To do that we will try to find the DFT of the signal in Example 7.3 where $x(0) = 1$, $x(1) = 0$ and $x(2) = 1$. The following MATLAB script is used.

```
% supply the x(n) array in a column form
N = 3;
```

```
xn = [1 0 1];
% initialize x(k) to a column of N zeros
Xk = zeros(N,1);
%supply the index n
n = 0: N-1;
%starting the loop to calculate X(k);
for k = 0: N-1
Xk =(exp(-j*2*k*pi/N).^n)*xn'
end
```

and the results are the same as those obtained in Example 7.3, where $x(0) = 2$, $x(1) = 1 + e^{-j\frac{4\pi}{3}}$ and $x(2) = 1 + e^{-j\frac{8\pi}{3}}$. To find $x(n)$ from $X(k)$ we use the inverse DFT and write the following MATLAB script:

```
% Supply the X(k) column array
Xk = [2 1 + exp(-j * 4 *pi/3) 1 + exp(-j *8 *pi/3)];
N = 3;
%fill the discrete signal x(n) with zeros
xn= zeros(N,1);
% supply the frequency index k.
k = 0: N-1;
% start the loop to compute x(n)
for n = 0: N-1;
xn = 1/N*(exp(-j * 2 * pi *n/N).^k) * Xk'
end
```

The results are again similar to the original signal $x(n)$, with $x(0) = 1$, $x(1) = 0$ and $x(2) = 1$.

7.6 The Fast Fourier Transform: A Faster Way of Computing the DFT

If we look at the following equation to calculate $X(k)$ from $x(n)$

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}}$$

we can see that to evaluate $X(k)$ for a single k value we need to perform N multiplications. Yet to calculate the N values for k we will need N^2 multipli-

cations. The goal is to reduce this number of multiplications. For that reason the FFT, a fast way of computing the DFT, was developed in which the number of multiplications, N^2, was reduced to $\frac{N(\log_2 N)}{2}$ multiplications. Thus if $N = 1024$, for example, it will take 1048576 multiplications using the DFT. Using the FFT, it will take 5120 multiplications, a drastic reduction in the number of multiplications.

To see how this FFT was developed, let us look back at the DFT equation. We have

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}}$$

If we define $W_N = e^{\frac{-j2\pi}{N}}$ then the DFT equation becomes

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}$$

Here we will derive what is called the radix-2 FFT and use what is called decimation in time. The last equation can be written as

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) W_N^{2nk} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^k W_N^{2nk}$$

With W_N^k not depending on the index n, this term can be pulled out and $X(k)$ becomes

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) W_N^{2nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^{2nk}$$

Notice that we are dividing $x(n)$ into even and odd parts. Notice also that

$$W_{\frac{N}{2}} = e^{\frac{-j2\pi}{N/2}} = e^{\frac{-j2\pi 2}{N}} = W_N^2$$

With this observation $X(k)$ becomes

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) W_{\frac{N}{2}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_{\frac{N}{2}}^{nk} \tag{7.11}$$

Now consider $X\left(k+\frac{N}{2}\right)$. With $W_{\frac{N}{2}}^{n\left(k+\frac{N}{2}\right)} = W_{\frac{N}{2}}^{nk}$ and $W_N^{k+\frac{N}{2}} = -W_N^k$ we can write $X\left(k+\frac{N}{2}\right)$ as

$$X\left(k+\frac{N}{2}\right) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)W_{\frac{N}{2}}^{nk} - W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(2n+1)W_{\frac{N}{2}}^{nk} \tag{7.12}$$

with $X(k)$ repeated as

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n)W_{\frac{N}{2}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(2n+1)W_{\frac{N}{2}}^{nk} \tag{7.13}$$

In Equation (7.13) we can find the first $N/2$ points. We can use Equation (7.12) for $X(k + N/2)$ to find the other $N/2$. Notice that the last two equations are similar with only the sign of W_N^k reversed.

To make things more clear, let us consider the case where $N = 8$, an 8-point DFT. With the last two equations, we can see that the 8-point DFT is reduced to the 4-point DFT. Remember that a 2-point DFT can be implemented without any complex multiplication. So if we subdivide the 4-point DFT further, we will get the 2-point DFT.

In general, if we are given N samples of $x(n)$ and are asked to find $X(k)$, we will first subdivide the N-point DFT into $N/2$-point DFT. We continue with this process until we get to the 2-point DFT.

The FFT is implemented in MATLAB and can be used directly to evaluate the DFT of the signal $x(n)$. We need to remind the reader here that the FFT will give the same result as the DFT. The FFT is not an approximation to the DFT in any way. Also, the properties for the DFT will hold for the FFT.

7.7 Applications of the DFT

We will look next at some of the important applications of the DFT.

7.7.1 Circular Convolution

Let us look at the input-output relationship between the input $x(n)$ and the output $y(n)$ of the discrete linear system.

$$y(n) = x(n) * h(n) = \sum_{m=-\infty}^{\infty} x(m)h(n-m)$$

where * indicates linear convolution. If we define the signals $x(n)$ and $h(n)$ to have zero values for $n < 0$ and $n \geq N$, then the linear convolution relationship becomes

$$y(n) = \sum_{m=0}^{N-1} x(m)h(n-m)$$

But we can see that if $n < 0$ and $n > 2N - 1$, $h(n - m)$ is zero. Hence

$$y(n) = 0 \quad \text{for} \quad n < 0 \text{ and } n > 2N - 1$$

Notice that $2N - 1$ is the length of the convolution result. However, $y(n)$ is generally not zero for $n \geq N$. Note also that if we take the N-points DFT of $y(n)$ we will miss the nonzero values of $y(n)$ as the result of the linear convolution. This indicates a need to define another form of convolution so that $y(n) = x(n) * h(n)$ is zero for $n < 0$ and $n \geq N$.

To arrive at this conclusion in a different approach, let $X(k)$, $H(k)$ and $Y(k)$ be the DFTs of the input $x(n)$, the impulse response $h(n)$ and the output $y(n)$ of the linear discrete system.

Let us claim now that

$$Y(k) = X(k)H(k) \tag{7.14}$$

Does this mean that

$$y(n) = x(n) * h(n)$$

where * indicates convolution?

Taking the inverse DFT of $Y(k) = X(k)H(k)$ leads to

$$y(n) = \frac{1}{N} \sum_{n=0}^{N-1} \left[\sum_{m=0}^{N-1} x(m) e^{\frac{-j2\pi mk}{N}} \sum_{p=0}^{N-1} h(p) e^{\frac{-j2\pi pk}{N}} \right] e^{\frac{j2\pi nk}{N}}$$

By rewriting the above equation we get

$$y(n) = \frac{1}{N} \sum_{m=0}^{N-1} x(m) \sum_{p=0}^{N-1} h(p) \sum_{k=0}^{N-1} e^{\frac{j2\pi k(n-m-p)}{N}}$$

But

$$\sum_{k=0}^{N-1} e^{\frac{j2\pi k(n-m-p)}{N}} = \begin{cases} N & n-m-p = 0 \\ 0 & \text{otherwise} \end{cases}$$

Thus, we can write

$$\sum_{k=0}^{N-1} e^{\frac{j2\pi k(n-m-p)}{N}} = N\delta(n-m-p)$$

where $N\delta(n)$ is the impulse signal with a strength of N at $n = 0$ and a strength of zero for $n \neq 0$. With the relation $\delta(n-m)\,f(n) = f(m)$ we can simplify the equation for $y(n)$ to get

$$y(n) = \sum_{m=0}^{N-1} x(m)h(n-m)$$

The above equation is similar to the linear convolution equation when it is not limited to one period N. Based on what has been presented we will define circular convolution as

$$y(n) = \sum_{m=0}^{N-1} x(m)h(n-m) \tag{7.15}$$

and use the notation

$$y(n) = x(n) \otimes h(n)$$

to indicate circular convolution. With this result we can say now that circular convolution in the discrete domain is equal to multiplication in the DFT domain. That is

$$y(n) = x(n) \otimes h(n) \leftrightarrow Y(k) = X(k)H(k) \tag{7.16}$$

Example 7.4

Consider the signals $x(n)$ and $h(n)$, each of length $N = 4$.

Solution

To calculate the output $y(n)$ using circular convolution we will write first the signals as

$$x(n) = \{x(0) \quad x(1) \quad x(2) \quad x(3)\}$$

$$h(n) = \{h(0) \quad h(1) \quad h(2) \quad h(3)\}$$

The output $y(n)$ will be of length 4 as well, and

$$y(0) = x(0)h(0) + x(1)h(1) + x(2)h(2) + x(3)h(3)$$

To find $y(1)$ we will move (shift) the first value in $h(n)$ and append it to the end of the array $h(n)$. The two arrays are now

$$x(n) = \{x(0) \quad x(1) \quad x(2) \quad x(3)\}$$

$$h(n) = \{h(1) \quad h(2) \quad h(3) \quad h(0)\}$$

and

$$y(1) = x(0)h(1) + x(1)h(2) + x(2)h(3) + x(3)h(0)$$

To find $y(2)$ we will list the two arrays by again performing the same shifting to get

$$x(n) = \{x(0) \quad x(1) \quad x(2) \quad x(3)\}$$

$$h(n) = \{h(2) \quad h(3) \quad h(0) \quad h(1)\}$$

with

$$y(2) = x(0)h(2) + x(1)h(3) + x(2)h(0) + x(3)h(1)$$

For the final value, $y(3)$, we will have

$$x(n) = \{x(0) \quad x(1) \quad x(2) \quad x(3)\}$$

$$h(n) = \{h(3) \quad h(0) \quad h(1) \quad h(2)\}$$

and

$$y(3) = x(0)h(3) + x(1)h(0) + x(2)h(1) + x(3)h(2)$$

Example 7.5

Consider the signals $x(n)$ and $h(n)$ with $N = 2$ and $x(0) = h(0) = 1$ and $x(1) = h(1) = 0$. Find the output $y(n)$ of the system given by this $h(n)$.

Solution

Using circular convolution we need to find $y(0)$ and $y(1)$. For y(0) we have

$$x(n) = \{1 \quad 0\}$$

$$h(n) = \{1 \quad 0\}$$

$$y(0) = (1)(1) + 0(0) = 1$$

For $y(1)$ we have

$$x(n) = \{1 \quad 0\}$$

$$h(n) = \{0 \quad 1\}$$

$$y(1) = (1)(0) + 0(1) = 0$$

But we also know that y(n) is the inverse DFT of $Y(k) = X(k)H(k)$. To find $X(k)$ and $H(k)$, we need to find $X(0)$, $X(1)$, $H(0)$ and $H(1)$.

$$X(0) = \sum_{n=0}^{1} x(n)e^{\frac{-j2\pi(0)}{N}} = x(0)(1) + x(1)(1) = 1 + 0 = 1$$

$$X(1) = \sum_{n=0}^{1} x(n)e^{\frac{-j2\pi(1)}{N}} = x(0)(1) + x(1)(-1) = 1$$

In the same way we have $H(0) = 1$ and $H(1) = 1$. $Y(k)$ is the term-by-term multiplication of $X(k)$ and $H(k)$ and is

$$Y(0) = X(0)H(0) = 1(1) = 1$$

$$Y(1) = X(1)H(1) = 0(0) = 0$$

$y(n)$ is the inverse DFT of $Y(k)$ and is given by

$$y(n) = \frac{1}{N}\sum_{k=0}^{N-1} Y(k)e^{\frac{j2\pi nk}{N}}$$

with

$$y(0) = \frac{1}{2}\left[Y(0)(1) + Y(1)(1)\right] = \frac{1}{2}\left[1 + 1\right] = 1$$

$$y(1) = \frac{1}{2}\left[Y(0)(1) + Y(1)(-1)\right] = \frac{1}{2}\left[1 - 1\right] = 0$$

7.7.2 Linear Convolution

Consider the two signals, $x(n)$ and $h(n)$, each of length N_1 and N_2, respectively. If we append zeros to the end of $x(n)$ and $h(n)$ to make the length of both equal to $N_1 + N_2 - 1$, then the linear convolution is the same as the circular convolution. It means that we can use the DFT to find linear convolution if both signals have the length $N_1 + N_2 - 1$.

Example 7.6

Consider $x(n)$ where $x(0)$ =1 and $x(1) = 0$ and $h(n)$ where $h(0) = 1$ and $h(1) = 0$. Find $y(n) = x(n) * h(n)$ and $y(n) = x(n) \otimes h(n)$.

Solution

First we will make $x(n)$ and $h(n)$ each of length $N_1 + N_2 - 1 = 2 + 2 - 1 = 3$ by appending zeros at the end and write

$$x(n) = \{1 \quad 0 \quad 0\}$$

$$h(n) = \{1 \quad 0 \quad 0\}$$

With circular convolution we have

$$y(0) = 1(1) + 0(0) + 0(0) = 1$$

To find $y(1)$ we list the arrays as

$$x(n) = \{1 \quad 0 \quad 0\}$$

$$h(n) = \{0 \quad 0 \quad 1\}$$

$$\text{with } y = 1(0) + 0(0) + 0(1) = 0$$

To find $y(2)$ we list the arrays again as

$$x(n) = \{1 \quad 0 \quad 0\}$$

$$h(n) = \{0 \quad 1 \quad 0\}$$

$$\text{with } y(2) = 1(0) + 0(1) + 0(0) = 0$$

So $y(n)$ with circular convolution is

$$y(n) = \{1 \quad 0 \quad 0\}$$

Using linear convolution we get the same result.

7.7.3 Approximation to the Continuous Fourier Transform

We have seen in Section 7.4.3 that the approximation to the Fourier transform of the signal $x(t)$ is related to the DFT $X(k)$ as in the relation

$$X_c\left(\frac{2\pi}{NT_s}k\right) = \frac{1 - e^{\frac{-j2\pi k}{N}}}{\frac{j2\pi k}{N}} X(k)$$

We can implement this approximation in MATLAB. Let us look at an example.

Example 7.7

Consider the signal

$$x(t) = \begin{cases} 1 & 0 < t < 2 \\ 0 & \text{otherwise} \end{cases}$$

Plot the actual $X(w)$ and its approximation using the DFT.

Solution

The actual Fourier transform of $x(t)$ is

$$X(w) = 2\text{sinc}(w)e^{-jw} = 2\frac{\sin(w)}{w}e^{-jw}$$

First we need to sample $x(t)$ to get $X(k)$ and then we need to scale $X(k)$ to get the approximation to $X(w)$, the actual continuous Fourier transform. In the following MATLAB script we will use a sampling period of 0.1 sec and a number of samples $N = 2^7 = 128$. Note that to use the `fft` function with Matlab, N must be 2 raised to an integer power.

```
integer=8;
N = 2.^integer; %number of samples in x(n)
Ts = 0.1;fs=1/Ts; %sampling period
NTs = N * Ts;
t = 0:Ts:2;
xn = [ones(1, length(t)) zeros(1, N-length(t))];% fill zeros
 to get N sample
Xk=fft(xn); % the fft used to calculate the DFT
exponent = 2 * pi / N;
% Next choose a k range
k = 1:2.^integer-1;
sf = ((1 - exp(-j*2*pi*k/N))./( j*2*pi*k/NTs));% the scaling
 factor
%we have avoided division by zero. Xwappr at k=0 is Ts
Xwappr = [Ts sf].* Xk;
k=0:2.^integer-1;
% transforming into the frequency axis: the spacing is fs/N
wappr=k*2*pi*fs/N;
% above is the approximate of X(w).
% next is the actual X(w)
wact = wappr(1, 2:length(wappr));%avoiding the zero for Xactual
% trying to avoid w = 0 (sin(w)/w)
```

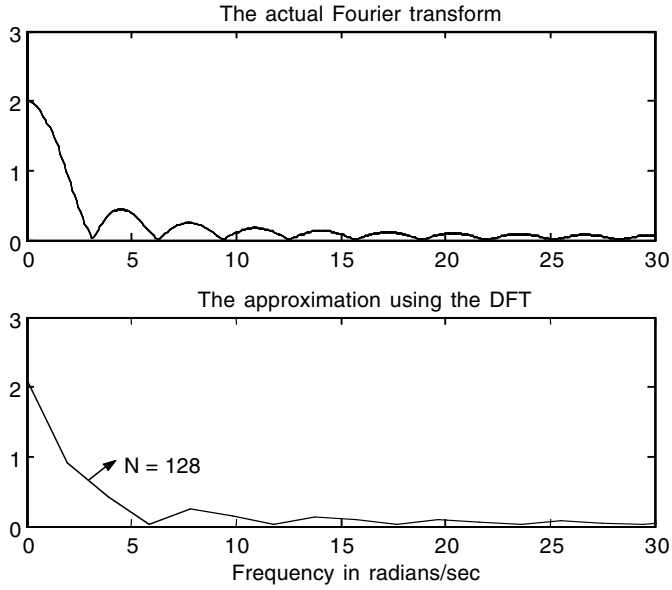

FIGURE 7.3 Plots for Example 7.7.

```
Xwact = (2 * sin(wact) ./wact).*exp(-j*wact); % point by point
 division (./)
Xwact = [2 Xwact]; % X(w) at w = 0 is 2
wact = wappr;
subplot(2,1,1);plot (wact, abs(Xwact));title('The actual
 Fourier transform');
axis([0 30 0 3]);
subplot(2,1,2); plot(wappr, abs(Xwappr));
title('The approximation using the DFT');
xlabel('Frequency in radians/sec');axis([0 30 0 3]);
```

The plots are shown in Figure 7.3. If we increase the number of samples N to $2^8 = 256$, we have the better approximation as seen in Figure 7.4.

7.7.4 Approximation to the Coefficients of the Fourier Series and the Average Power of the Periodic Signal $x(t)$

Given a periodic continuous signal, $x(t)$ with the period T the Fourier series approximation is calculated first by evaluating the Fourier series coefficients

$$c(k) = \frac{1}{T}\int_T x(t)e^{-j\mathrm{w}_0 kt}dt \tag{7.17}$$

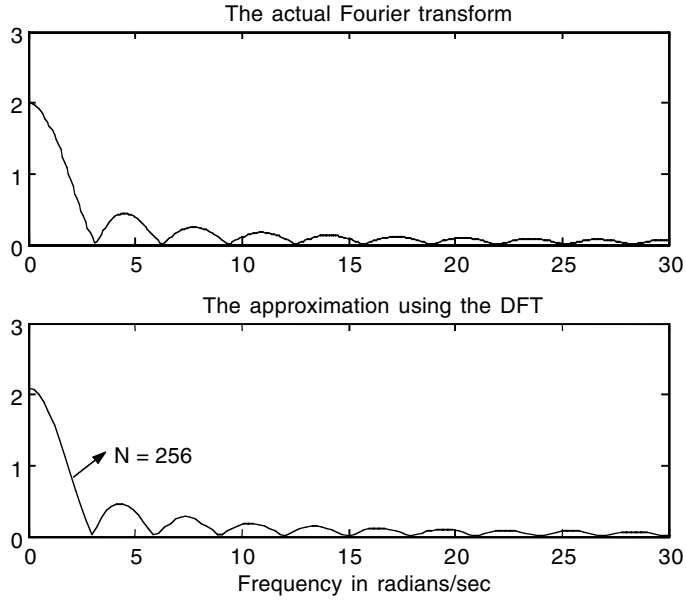

FIGURE 7.4 Plots for Example 7.7.

For the computer to approximate this integration we need to represent $x(t)$ in a discrete form. Let us sample $x(t)$ at $t = nT_s$ where T_s is the sampling interval and n is an integer. Thus we will have

$$x(n) = x(t)\Big|_{t=nT_s}$$

Remember that we are integrating over the period T in continuous time. In discrete time we need to subdivide this period T into intervals of width T_s and we will have N of these intervals. Thus we write $T = NT_s$. If we make T_s very small (N very large) and approximate the area under $x(t)e^{-jw_0kt}$ in these intervals we can have a good approximation to the integral equation given.

With this process, w_0 will be

$$w_0 = \frac{2\pi}{T} = \frac{2\pi}{NT_s}$$

and $c(k)$ will be

$$c(k) = \frac{1}{NT_s}\left[\sum_{n=0}^{N-1} x(nT_s)e^{-j\frac{2\pi}{NT_s}knT_s}\right]T_s$$

where the integration becomes summation and hence dt becomes T_s. Finally, the equation for $c(k)$ becomes

$$c(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(nT_s)e^{-j\frac{2\pi kn}{N}} \tag{7.18}$$

The above equation is $1/N$ times the DFT of $x(nT_s)$. Thus

$$X(k) = Nc(k)$$

where $X(k)$ is the DFT of $x(nT_s)$. Remember also that

$$c(0) = \frac{1}{T}\int_0^{T_s} x(t)dt$$

is the average value of $x(t)$. Hence $X(0)/N$ is an approximation to the average value of $x(t)$. The average power in the periodic signal $x(t)$ is given as

$$P_a = \frac{1}{T}\int_0^{T} |x(t)|^2 dt \tag{7.19}$$

and its approximation using the Fourier series coefficients is

$$P_a \approx \sum_{k=-\infty}^{\infty} |c(k)|^2 \tag{7.20}$$

But with $X(k) = Nc(k)$ we have the average power approximated as

$$P_a \approx \frac{1}{N^2}\sum_{k=0}^{N-1} |X(k)|^2 \tag{7.21}$$

Example 7.8

Consider the signal

$$x(t) = \sin(2\pi t)$$

Find the average value of the signal, its average power and its Fourier series coefficients as well as their approximations.

Solution

The signal is periodic with period $T = 1$ sec. The average value is

$$c(0) = \frac{1}{1}\int_0^1 \sin(2\pi t)dt = -\left(\frac{1}{2\pi} - \frac{1}{2\pi}\right) = 0$$

The average power is given as

$$P_a = \frac{1}{1}\int_0^1 |\sin(2\pi t)|^2 dt = 1/2$$

The other Fourier series coefficients can be found by using MATLAB as in the following script:

```
syms t;% symbolic constants
T=1; %period of the signal
w=2*pi/T;
for k=0:3
  %performing the integration to calculate the coefficients
  ck=1/T*int(sin(w*t)*exp(-j*w*k*t),t,0,1)
end
```

The result is

ck = 0

ck = –1/2*i

ck = 0

ck = 0

You can see from this result, as expected, that the sine wave has only one frequency component, that $c(0) = 0$, the average value and the $|c(1)| = 1/2$ is the only frequency component that is nonzero.

We can use the DFT to find an approximation to $c(0)$, the average power and the other frequency components. First we see that $x(t)$ is periodic with $T = 1$ sec. We will divide this T interval into equally spaced intervals. With the frequency of the signal $x(t)$ equal to 1 Hz, we choose f_s to be 10 Hz. Thus, with $f_s = 10$ Hz, $T_s = .1$ sec. The number of samples we will take from $x(t)$ in the T interval is

$$N = \frac{T}{T_s} = \frac{1}{0.1} = 10 \text{ samples}$$

The MATLAB script is given next.

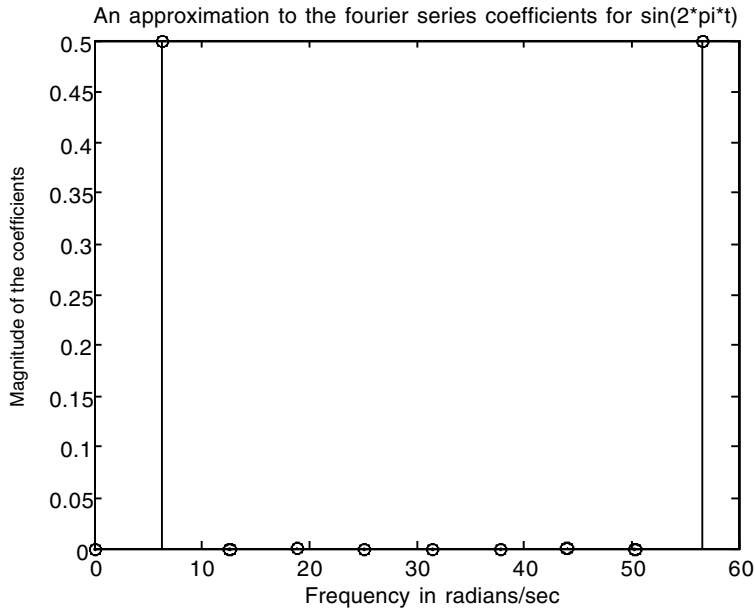

FIGURE 7.5 Plots for Example 7.8.

```
Ts=0.1; fs=1/Ts;T=1; N=T/Ts;
n=0:N-1;
xn=sin(2*pi*n*Ts);
Xk=fft(xn);
ck=Xk/N;
average=Xk(1)/N
avpower=(1/(N*N))*sum(abs(Xk).^2)
k=0:N-1;
waxis=k*2*pi*fs/N;
stem(waxis,abs(ck));xlabel('Frequency in radians/sec');
title('An approximation to the fourier series coefficients
 for sin(2*pi*t)');
ylabel('Magnitude of the coefficients');
```

The average value as expected is zero and the average power is 1/2. The plot is shown in Figure 7.5.

7.7.5 Total Energy in the Signal $x(n)$ and $x(t)$

The total energy in the signal $x(n)$ for $n = 0, 1, \ldots, N-1$ is given as

$$E_t = \sum_{n=0}^{N-1} \left|x(n)\right|^2 = \frac{1}{N} \sum_{k=0}^{N-1} \left|X(k)\right|^2 \tag{7.22}$$

If we are given a nonperiodic signal $x(t)$, we can talk about its total energy. If we sample $x(t)$ at the sampling rate f_s satisfying the Nyquist sampling rate, then the approximate energy in the signal is

$$E_t = \sum_{n=n_1}^{n_2} \left|x(nT_s)\right|^2 T_s = \frac{T_s}{N} \sum_{k=0}^{N-1} \left|X(k)\right|^2$$

n_1 and n_2 are the integer values for which $x(t)$ is approximately zero for $t < n_1$ and $t > n_2$.

Example 7.9

Consider the signal $x(t) = e^{-t}$ $t > 0$. What is the total energy in the signal?

Solution

The total energy in the signal is

$$E_t = \int_0^\infty \left|\left(e^{-t}\right)\right|^2 dt = 1/2$$

We can compute this value of 1/2 using the DFT. But first let us look at the continuous Fourier transform of $x(t)$

$$X(w) = \frac{1}{1 + jw}$$

The magnitude of $X(w)$ is given as

$$\left|X(w)\right| = \frac{1}{\sqrt{1 + w^2}}$$

We can see that for $w > 2\pi(50)$, the magnitude of $X(w)$ approaches zero. Thus let f_s = 2(50) =100 Hz be the sampling frequency. We can also see that for $t > 5$, the signal $x(t)$ approaches zero as well. Therefore we will choose the time interval of 5 sec and sample $x(t)$ in this interval. In this case the number of samples N is T/T_s = 5(100) = 500.

We can use MATLAB to calculate the total energy in $x(t)$ as in the following script:

```
fs=100; Ts =1/fs; T=5;% T is the time interval for sampling
 x(t)
N=T/Ts;
n=0:1:500; %n*Ts= 500*.01=5=the time interval T
```

```
xn=exp(-n*Ts);
Xk=fft(xn);
Etotal=Ts*sum(abs(xn).*abs(xn))%total energy in real time
Effttotal=Ts/N*sum(abs(Xk).*abs(Xk))%total energy in x(t)
 using fft
```

to get

Etotal = 0.505 using the time domain
Effttotal = 0.506 using the DFT

Note that if we were given a discrete signal $x(n)$ such that

$$x(0)=1 \quad x(1)=0 \quad x(2)=1 \quad x(3)=1 \quad x(4)=0$$

then

$$E_t=\sum_{n=0}^{N-1}\left|x(n)\right|^2=\left(x(0)\right)^2+\left(x(1)\right)^2+\left(x(2)\right)^2+\left(x(3)\right)^2+\left(x(4)\right)^2=1+0+1+1+0=3$$

MATLAB can be used also to compute the total energy in the signal as in the following script:

```
xn=[1 0 1 1 0];
Xk=fft(xn);
Etotal=sum(abs(xn).*abs(xn))%total energy in real time
Effttotal=1/5*sum(abs(Xk).*abs(Xk))%total energy in
 x(n)using fft
```

The result will be

Etotal = 3
Effttotal = 3

7.7.6 Block Filtering

A digital filter can be described by the impulse response $h(n)$. $h(n)$ can change the magnitude of the input signal as well as produce a phase shift. In using the DFT to calculate $y(n)$, the output of the filter $h(n)$, notice that you need the entire $x(n)$ present. In case of huge $x(n)$, this will cause a considerable delay in obtaining $y(n)$. If we break $x(n)$ into blocks of data, we can use the DFT to produce blocks of the output $y(n)$. These blocks then can be arranged to produce the total output $y(n)$. This method will produce outputs faster

and will reduce the waiting time for incoming $x(n)$ samples. An example of how to perform this method will be given in the EOCE section later.

7.7.7 Correlation

Correlation is often used in the detection of a target in a radar signal. It can also be used in the estimation of the frequency content of a certain signal. Cross-correlation is the correlation between two different signals and auto-correlation is the correlation with the signal itself. The cross-correlation is given by the relation

$$R_{x1x2}(p) = \sum_{n=-\infty}^{+\infty} x_1(n)x_2(p+n) \tag{7.23}$$

The above equation is similar to the convolution equation with the only difference being that $x_2(n)$ is shifted but not reflected.

In dealing with discrete signals we use N samples and assume that the signals are periodic with period NT_s. With periodic signals and if $0 \le n \le N-1$ we have

$$x(-n) = x(-n+N)$$

The cross-correlation equation can be written then as

$$R_{x_1x_2}(p) = \sum_{n=-\infty}^{+\infty} x_1(n)x_2\big(p-(-n)\big) = x_1(-n) * x_2(n) \tag{7.24}$$

If we take the DFT of the above result we will have

$$R_{x_1x_2} \leftrightarrow X_1(-k)X_2(k) \tag{7.25}$$

and then the inverse DFT of $X_1(-k)$ $X_2(k)$ is $R_{x1x2}(p)$. If $x_2(n)$ is real then

$$X_1(-k) = X_1{}^*(k)$$

Finally, we have

$$R_{x_1x_2}(p) \leftrightarrow X_1(k)^* X_2(k) \tag{7.26}$$

Note that $R_{x_1x_2}(p) \neq R_{x_2x_1}(p)$. It can be shown that

$$R_{X_2X_1}(p) \leftrightarrow x_1(k)x_2(k)^* \tag{7.27}$$

One important application of auto-correlation is the estimation of the energy spectrum density of the signal $x(n)$. The auto-correlation process helps to eliminate the noise if it is present in a certain signal. The energy spectrum density estimate is

$$R_{xx}(p) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)x(p+n) \qquad 0 \le p \le N-1 \tag{7.28}$$

From the above relation we have

$$R_{xx}(p) \leftrightarrow \frac{1}{N} X(k)X(k)^* \tag{7.29}$$

We will give an example on how to use the above relation later in the EOCE.

7.8 Some Insights

7.8.1 The DFT Is the Same as the fft

The DFT is not an approximation to the fft; it is the same as the fft. The fft is a fast and an efficient way of calculating the DFT.

7.8.2 The DFT Points Are the Samples of the Fourier Transform of $x(n)$

The Fourier transform of the discrete signal $x(n)$ for $n = 0, 1, \ldots, N-1$ is

$$X\left(e^{j\theta}\right) = \sum_{n=0}^{N-1} x(n)e^{-j\theta n}$$

If we sample $X(e^{j\theta})$ on the unit circle with $\theta = 2\pi k/N$ for $k = 0, 1, \ldots, N-1$, then

$$X(k) = X\left(e^{j\theta}\right)\Big|_{\theta=\frac{2\pi k}{N}}$$

7.8.3 How Can We Be Certain That Most of the Frequency Contents of $x(t)$ Are in the DFT?

To get $X(k)$ from x(t) we need first to sample $x(t)$ making sure that f_s, the sampling frequency used to produce $x(n)$, is at least twice f_m, the maximum frequency in $x(t)$. This is necessary to avoid aliasing. Next, N samples should be collected from $x(t)$ in NT_s sec, where $T_s = 2\pi f_s$. The number of samples N is inversely related to the frequency resolution Δf according to

$$\Delta f = \frac{2\pi}{N}$$

The smaller the frequency resolution is, the better is the process of detecting most of the frequency components in $x(t)$.

7.8.4 Is the Circular Convolution the Same as the Linear Convolution?

The circular convolution between the two signals $x_1(n)$ and $x_2(n)$ will be the same as the linear convolution if we append zeros to the end of $x_1(n)$ and $x_2(n)$ so that the number of samples in $x_1(n)$ and $x_2(n)$ will be $N_1 + N_2 - 1$ where N_1 is the number of samples in $x_1(n)$ and N_2 is the number of samples in $x_2(n)$.

7.8.5 Is $|X(w)| \cong |X(k)|$?

Due to the inherent factor $1/T_s$ in calculating the Fourier transform of discrete signals, the magnitude of the DFT, $|X(k)|$ should be multiplied by T_s to get the approximation to $|X(w)|$. Note that this approximation is not as good as the approximation we discussed earlier in this chapter.

7.8.6 Frequency Leakage and the DFT

In the development of the DFT, the signal $x(nT_s)$ is made periodic first and then multiplied by the rectangular window of unity magnitude that extends from $n = 0$ to $n = N - 1$. The result of this multiplication is the signal $x(n)$. To understand this better consider the signal $x(t) = \sin(t)$. This signal is periodic and there is no need to make it periodic. Its Fourier transform consists of one component at $w = 1$. If we truncate $x(t)$ by multiplying it with a rectangular window that is centered at $t = 0$, the Fourier transform of $\sin(t)$ multiplied by this window will result in a sinc-shape graph centered at $w = 1$. From this you can see that the frequency content of $x(t)$ after this multiplication is distorted. This is referred to as frequency leakage.

To have better approximation, different windows can be used. Two well-known windows are the Hanning and the Hamming windows. These windows do not have sharp cuts at the edges; they gradually approach zero. The two windows are implemented in Matlab.

7.9 End of Chapter Exercises

EOCE 7.1

Consider the discrete time system represented by

$$h(n) = (0.5)^n \quad n = 0, 1, 2, \ldots, 10$$

and consider an input to the system given by

$$x(n) = u(n) \quad n = 0, 1, 2, \ldots, 5$$

Find the output $y(n)$ using the DFT method.

Solution

We have $N_h = 11$ samples for $h(n)$ and $N_x = 6$ samples for $x(n)$. So we need to add $6 - 1 = 5$ zeros to $h(n)$ and $11 - 1 = 10$ zeros to $x(n)$ so that each signal will have $(11 + 6) - 1 = 15$ samples. At this point we can use the DFT to calculate $y(n)$. The MATLAB script that follows will compute $y(n)$ using the function `conv` and it will also compute $y(n)$ by inverse transforming $Y(k)$ for comparison.

```
nh=0:10; nx=0:5;
hn=.5.^nh; xn=1.^nx; %original signals
yn=conv(xn,hn)%output using direct convolution
hn=[hn zeros(1,length(nx)-1)];%zero padding
xn=[xn zeros(1,length(nh)-1)];
Hk=fft(hn); Xk=fft(xn); Yk=Hk.*Xk;
Yn=ifft(Yk)%inverse fft to get y(n)
n=0:15;
subplot(2,1,1);stem(n,yn);title('Output using direct
 convolution');
subplot(2,1,2);stem(n,Yn); xlabel('n');
title('Output using the inverse DFT');
```

The result is plotted in Figure 7.6. Note that we have $h(n)$ and $x(n)$ each of length 16. This means that the radix-2 fft was used. If we want the faster and more efficient radix-2 to be used, each signal should have 2^p as its length where p is a positive integer.

EOCE 7.2

Find an approximation to the magnitude of the Fourier transform of

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

Solution

To use the DFT to approximate the Fourier transform of $x(t)$ we need to sample $x(t)$ first. Let us look at the Fourier transform of $e^{-10t}u(t)$ so that we can find the frequency at which $X(w)$ is approaching zero. The Fourier transform is

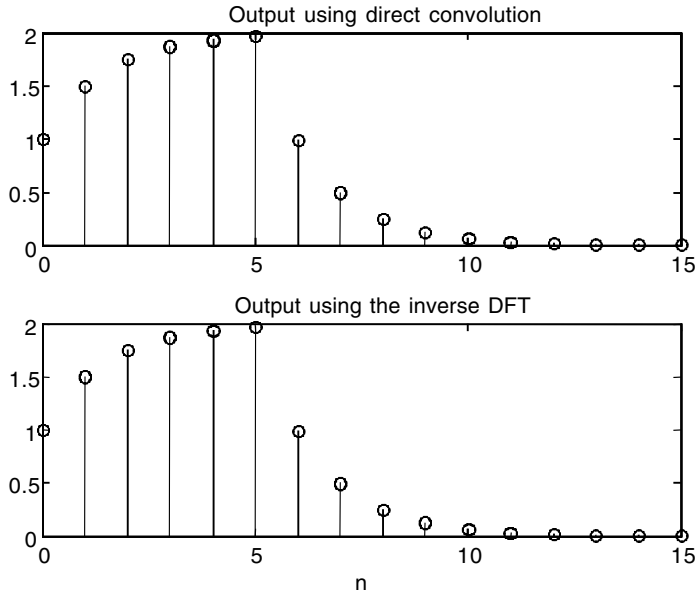

FIGURE 7.6 Plots for EOCE 7.1.

$$X(w) = \frac{1}{jw + 20}$$

and the magnitude of $X(w)$ is

$$|X(w)| = \frac{1}{\sqrt{w^2 + 100}}$$

For $w = 30\pi$ rad/sec, the magnitude of $X(w)$ is approximately zero. Thus we choose $w_s \geq 2w = 60\pi$. Let us choose $w_s = 300\pi$ with $f_s = 150$ Hz and $T_s = 1/150$ sec. Note that at $t = 1$ sec, $x(t)$ is approximately zero. Let us choose $NT_s = 1$ or $N = 1/T_s = 150$ samples. We have established a lower limit on N of 150. To use the radix-2 fft, N must be an integer power of 2. The next N then will be $2^8 = 256$.

We have seen in this chapter two ways to perform this approximation. In the first method the approximation was given by

$$\frac{1 - e^{\frac{-j2\pi k}{N}}}{j\dfrac{2\pi k}{NT_s}} X(k)$$

and in the second method, the approximation to $|X(w)|$ was T_s ($X(k)$). In the following script we use the two methods.

```
integer=8;
N = 2.^integer; %number of samples in x(n)
Ts = 1/150;fs=1/Ts; %sampling period
NTs = N * Ts;
n=0:150; %n*Ts=1second for n=150.The record length taken from
 x(t)
xn = [exp(-10*n*Ts) zeros(1, N-length(n))];% fill with zeros
 to get N samples
Xk=fft(xn); % the fft used to calculate the DFT
% Next choose a k range
k = 1:2.^integer-1;
sf = ((1 - exp(-j*2*pi*k/N))./( j*2*pi*k/NTs));% the scaling
 factor
Xwappr1 = [Ts sf].* Xk; %First method of approximation
k=0:2.^integer-1;
% transforming into the frequency axis: the spacing is fs/N
wappr=k*2*pi*fs/N;
% above is the approximate of X(w).
% next is the actual X(w)
Xwappr2=Ts*Xk;% Second method of approximation
wact = wappr;
Xwact =1./(sqrt(wact.^2+100)); % point by point division (./)
subplot(3,1,1);plot (wact, Xwact);title('The actual Fourier
 transform');
axis([0 30 0 .15]);
subplot(3,1,2);plot(wappr,abs(Xwappr1));
title('First method: approximation using the DFT');
axis([0 30 0 .15]);
subplot(3,1,3); plot(wappr, abs(Xwappr2));
title('Second method: approximation using the DFT');
xlabel('Frequency in radians/sec');axis([0 30 0 .15]);
```

The plots are shown in Figure 7.7.

EOCE 7.3

Consider the signal

$$x(t) = \sin(2\pi 2000t) + \frac{1}{2}\sin(2\pi 4000t)$$

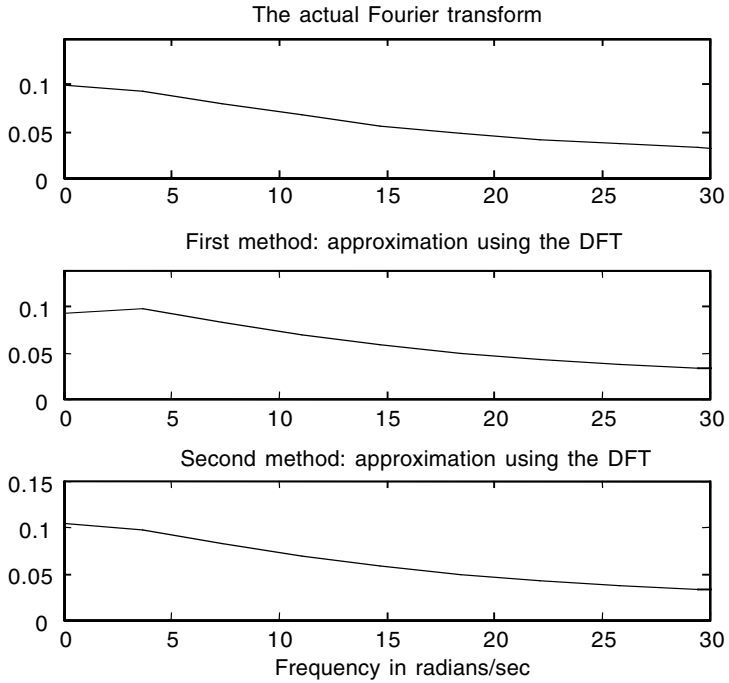

FIGURE 7.7 Plots for EOCE 7.2.

1. Use the DFT to approximate $|X(w)|$ with N =16, and choose f_s such that the Nyquist criteria is observed and $f_s/N = f_1/k_1 = f_2/k_2$ where k_1 and k_2 are integers and f_1 and f_2 are the frequencies of the two components in $x(t)$, 2000 Hz and 4000 Hz.
2. Repeat while $f_s/N \neq m_1 f_1 \neq m_2 f_2$.
3. Can you do a correction to the approximation?

Solution

1. The highest frequency in $x(t)$ is f_2 = 4000 Hz. With f_s = 16000 Hz,

$$\frac{f_s}{N} = \frac{16000}{16} = 1000 = \frac{f_1}{2} = \frac{f_2}{4}$$

Therefore, on the frequency axis of the magnitude of $|X(k)|$, we will see that the frequency content of the first component at f_1 = 2000 Hz corresponds to k_1 = 2, and the frequency content of the second component at f_2 = 4000 Hz corresponds to k_2 = 4. In this case the Fourier transform of $x(t)$, $X(w)$, will match exactly the DFT of $x(n)$, $X(k)$. The MATLAB script to prove that follows.

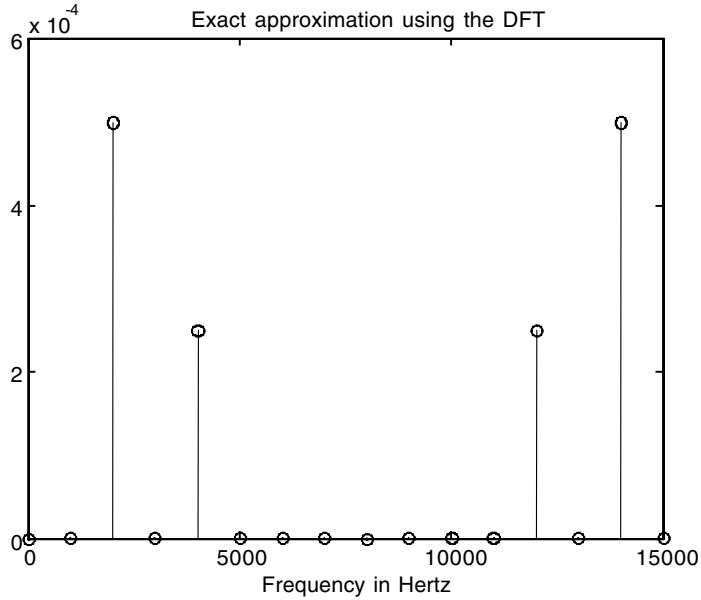

FIGURE 7.8 Plot for EOCE 7.3.

```
fs=16000;Ts=1/fs;
N=16;
n=0:15; % 16-points fft
t=n*Ts;
xn=sin(2*pi*2000*t)+0.5*sin(2*pi*4000*t);
Xk=fft(xn);
Xkappr=Ts*Xk;
k=n*fs/N; % the frequency axis
stem(k,abs(Xkappr));
xlabel('Frequency in Hertz');
title('Exact approximation using the DFT')
```

The plot is shown in Figure 7.8. The magnitude of $X(2) = 8$ is the magnitude of the 2000 Hz term times N divided by 2. This is true because the first term in $x(t)$ has no dc components.

2. If we sample at $f_s = 10000$ Hz (satisfying the Nyquist rate) and with $N = 16$ we have

$$\frac{f_s}{N} = \frac{10000}{16} \neq \frac{f_1}{k_1} \neq \frac{f_2}{k_2}$$

for any k_1 or k_2. In this case there will be distortion in the frequency spectrum using the DFT. This is to say that the 2000- and 4000-Hz

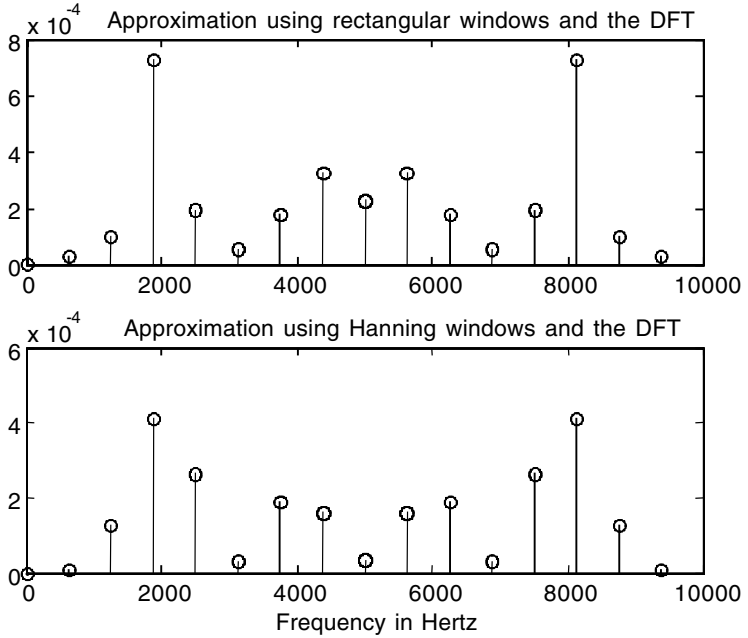

FIGURE 7.9 Plot for EOCE 7.3.

frequencies will not be observed at any value of k in the DFT magnitude plot. For $f_s = 10{,}000$ Hz the approximation to $x(t)$ will be as seen in Figure 7.9.

3. If we use the Hanning window instead of the rectangular window inherent in the DFT development, we will get some improvements as seen in Figure 7.9. The MATLAB script for using the Hanning and the rectangular windows follows.

```
fs=10000;Ts=1/fs;
N=16;
n=0:15; % 16-points fft
t=n*Ts;
xn=sin(2*pi*2000*t)+0.5*sin(2*pi*4000*t);
hn=hanning(16); % 16-points Hanning window
wn=xn.*hn'; % truncation using Hanning window
Xkrec=fft(xn);
Xkhan=fft(wn);
Xkrecappr=Ts*Xkrec;
Xkhanappr=Ts*Xkhan;
k=n*fs/N; % the frequency axis
subplot(2,1,1);stem(k,abs(Xkrecappr));
```

```
title('Approximation using rectangular windows and the
 DFT');
subplot(2,1,2);stem(k,abs(Xkhanappr));
xlabel('Frequency in Hertz');
title('Approximation using Hanning windows and the DFT');
```

EOCE 7.4

Consider the signals $x(t)$ and $h(t)$ that are the input and the impulse response of a linear system

$$h(t) = e^{-100t}$$

$$x(t) = \sin(2\pi(1000)t)$$

Find the output $y(n)$ and display the frequency spectrum of the input and the output using the DFT.

Solution

The input $x(t)$ is periodic with period $T = 0.001$. We will sample $x(t)$ at f_s = 8000 Hz satisfying the Nyquist rate. For f_s = 8000 Hz we have $T_s = 1/8000$ sec. We will sample $x(t)$ for $T = 0.001$ sec. To choose a value for N we use the relation $NT_s = T$ or $N = T/T_s = 0.001/1/8000 = 8$ samples.

For $h(t) = e^{-100t}$, we see that if $t = 0.1$ sec $h(t)$ will be very close to zero in value. The Fourier transform of $h(t)$ is $H(w)$ and is

$$H(w) = \frac{1}{jw + 100}$$

with a magnitude of

$$|H(w)| = \frac{1}{\sqrt{w^2 + (100)^2}}$$

With $w = 2\pi(20)$ rad/sec, $|H(w)|$ will approach zero. So we choose w_s as $w_s > 2(2\pi)(20)$ to satisfy the Nyquist rate. Let us make $w_s > 2\pi(200)$. Thus $f_s = 200$ and $T_s = 1/200$ sec. So for a time length of 0.1 sec (at which $h(t) \approx 0$) and with $T_s = 1/200$ we have

$$N\left(\frac{1}{200}\right) = 0.1 \text{ or } N = 0.1(200) = 20 \text{ samples}$$

Now we will find the DFT for $x(t)$ with $N = 8$ and the DFT for $h(t)$ with $N = 20$. To find $y(n)$ we will use convolution by making $x(n)$ and $h(n)$ both of length $n_1 + n_2 - 1$ where n_1 and n_2 are the number of samples in $x(n)$ and $h(n)$, respectively. To do that we will pad $x(n)$ and $h(n)$ by zeros. The following script will do that.

```
%For x(n) we will sample for 0.001 seconds
Ts1=1/8000; n1= 0:8;t1=n1*Ts1; N1=8;
xn=sin(2*pi*1000*t1);
%For h(n) we will sample for .1 seconds
Ts2=1/200;n2=0:20; t2=n2*Ts2; N2=20;
hn=exp(-100*t2);
subplot(1,3,1); stem(n1,xn);title('x(n)');
xlabel('n');
subplot(1,3,2); stem(n2,hn);title('h(n)');
xlabel('n');
%next we make both signals of size n1+n2 -1
N=length(xn)+length(hn)-1;
xn=[xn zeros(1, N-length(xn))];
hn=[hn zeros(1, N-length(hn))];
Xk=fft(xn); Hk=fft(hn);
Yk=Xk.*Hk;
yn=ifft(Yk);
%Plotting
n=0:1:N-1;
subplot(1,3,3); stem(n,yn);title('y(n)');
xlabel('n');
```

The plots are shown in Figure 7.10. Notice in the above MATLAB script that the DFT of $x(n)$, $h(n)$ and the inverse transform of $Y(k)$ were calculated using the relation

```
N = length(xn) + length(hn) -1;
```

But 27 is not an integer power of 2 and hence the radix-2 fft was not used. To use the radix-2 fft we need not pad $x(n)$ or $h(n)$ with zeros since padding will be made by calling the `fft` function

```
Xk = fft(xn, N);
```

In this case $x(n)$ will be made of length N where (N – length (xn)) is the number of the padded zeros. We can use this method with $h(n)$ as well. When we use the command

```
Yk = Xk * Hk';
```

`Yk` will be of length N as well.

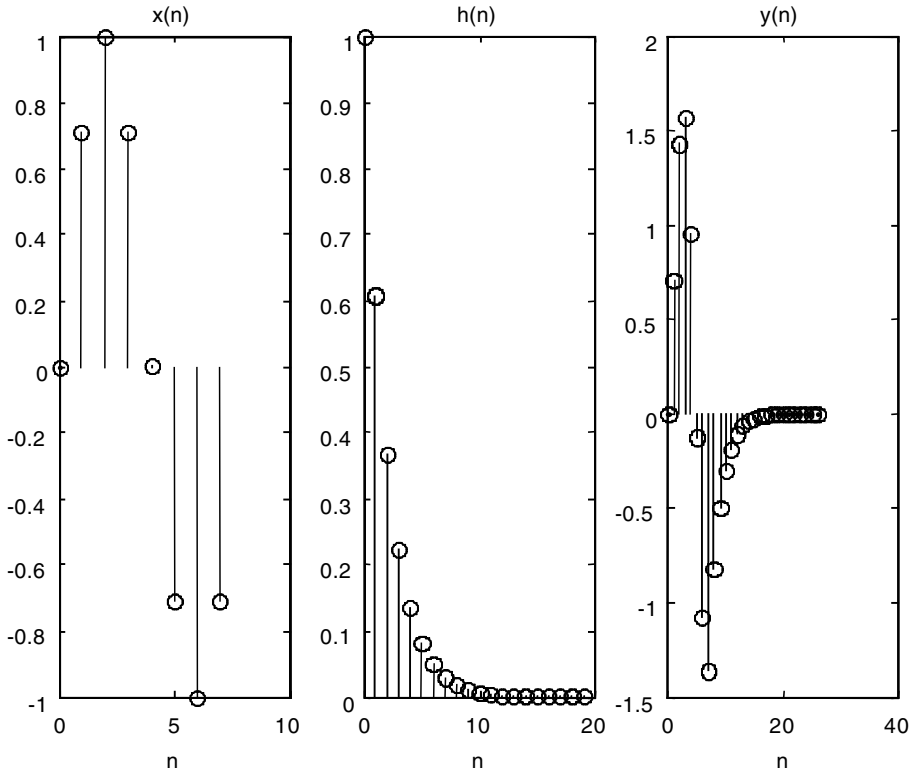

FIGURE 7.10 Plots for EOCE 7.4.

EOCE 7.5

Consider the discrete system where the input $x(n)$ is the pulse defined as

$$x(n) = \begin{cases} 1 & 0 \le n \le 5 \\ 0 & \text{otherwise} \end{cases}$$

and the impulse response is defined as

$$h(n) = (.2)^n u(n)$$

Find the output $y(n)$ using the DFT.

Solution

The signals here are already sampled. We can see that $x(n)$ is zero for $n \ge 6$. Thus the number of samples in $x(n)$ is 6. For $h(n)$, there is no real integer that makes $h(n)$ zero. But if $n = 5$, $h(n)$ is close to zero. Thus $h(n)$ will have

a value close to zero for $n \geq 5$. For this $h(n)$ we will have five samples. To use radix-2 fft, $N_1 + N_2 - 1$ should be an integer multiple of 2. $N_1 + N_2 - 1 = 6 + 5 - 1 = 10 \neq 2^p$ for any integer p. The next integer 2^p greater than 10 is $16 = 2^4$. So we need to use at least 16-point DFT to find $y(n)$ using the radix-2 fft. The MATLAB script follows.

```
nx=0:5; nh=0:4;
xn=ones(1,6); hn=.2.^nh;
Xk=fft(xn,16); Hk=fft(hn,16);
Yk=Xk.*Hk;
yn=ifft(Yk);
% for plotting
n=0:15;
subplot(1,3,1);stem(nx,xn);title('The input signal');
xlabel('n');
subplot(1,3,2);stem(nh,hn);title('The impulse response
 signal');
xlabel('n');
subplot(1,3,3);stem(n,yn);title('The output signal');
xlabel('n');axis([0 15 0 1.4]);
```

The plot is shown in Figure 7.11.

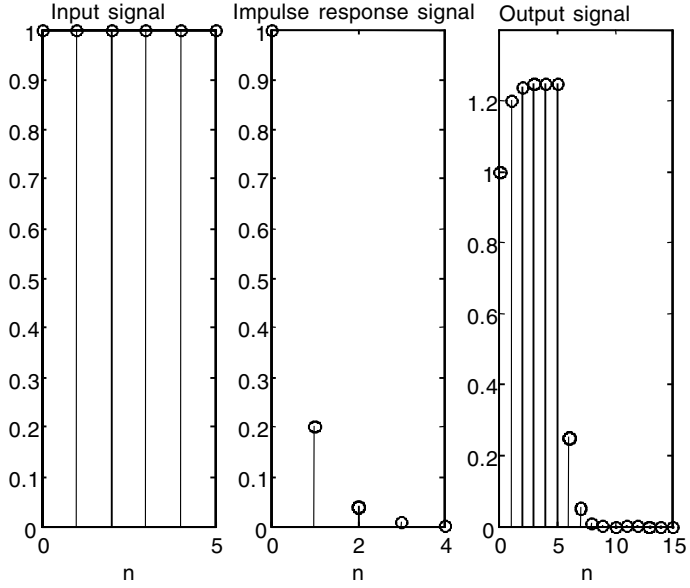

FIGURE 7.11 Plots for EOCE 7.5.

EOCE 7.6

Suppose that we have 2 sec of the signal $x(t)$. Let us sample $x(t)$ at $f_s = 1000$ Hz.

1. Find the maximum frequency that can be present if there is to be no aliasing.
2. What analogue frequencies are present in the DFT?
3. What is the frequency resolution?

Solution

1. With f_s = 1000 Hz, from the Nyquist condition we have

$$f_s \geq 2f_m \text{ or } f_m = \frac{f_s}{2} = \frac{1000}{2} = 500\text{Hz}$$

2. The frequency spacing (the frequency resolution) is given by

$$\frac{f_s}{N} = \frac{1}{T_r} = \frac{1}{2} = 0.5\text{Hz}$$

   where T_r is the total time of observing the signal and is called the record length.
3. From (2) with $f_s/N = 0.5$ we get

$$N = \frac{f_s}{.5} = \frac{1000}{.5} = 2000$$

   The frequencies in the DFT are at

$$\frac{mf_s}{N} = m\frac{1000}{2000} = \frac{1}{2}m \text{ Hz}$$

   where m is an integer. So they are at

$$0,\ \frac{1}{2},\ 1\ ,\ \frac{3}{2},\ 2, \ldots, 500, -499, -\frac{1}{2}$$

EOCE 7.7

Consider the two signals $x_1(n) = x_2(n)$ with

$$x_1(0) = 1,\ x_1(1) = 1,\ x_1(2) = 0, \text{ and } x_1(3) = 1$$

Find the cross-correlation between $x_1(n)$ and $x_2(n)$, Rx_1x_2 and Rx_2x_1.

Solution

Note that $x_1(n)$ has four samples as well as $x_2(n)$. To use the DFT to find the cross-correlation, we need to make both $x_1(n)$ and $x_2(n)$ of length $4 + 4 - 1 = 7$ samples. We will pad $x_1(n)$ with 3 zeros and we will do the same for $x_2(n)$. The following MATLAB script will accomplish this.

```
x1n=[1 1 0 1 0 0 0];n1x=0:6;
x2n=[1 1 0 1 0 0 0];n2x=0:6;
X1k=fft(x1n); X2k=fft(x2n);
X1kconj=conj(X1k);
Rx1x2=X1kconj.*X2k;
X2kconj=conj(X2k);
Rx2x1=X1k.*X2kconj;
rx1x2=ifft(Rx1x2);
rx2x1=ifft(Rx2x1);
n=0:6;
subplot(2,2,1);stem(n1x,x1n);title('x1(n)');
xlabel('n');
subplot(2,2,2);stem(n2x,x2n);title('x2(n)');
xlabel('n');
subplot(2,2,3);stem(n,rx1x2);title('cross-correlation
 between x1 and x2');
xlabel('n');
subplot(2,2,4);stem(n,rx2x1);title('cross-correlation
 between x2 and x1');
xlabel('n');
```

The plots are shown in Figure 7.12. We could have used the MATLAB function conv to find the same result. The next script can be used.

```
x1n=[1 1 0 1];n1x=0:3;
x2n=[1 1 0 1];n2x=0:3;
N=length(x1n);
n=1:N;
x1nref=x1n(N+1-n);
n1x = -fliplr(n1x);%the index changes too
x2nref=x2n(N+1-n);
n2x = -fliplr(n2x);
rx1x2=conv(x1nref,x2n);
rx2x1=conv(x1n,x2nref);
n=-3:3;
subplot(2,2,1);stem(n1x,x1n);title('x1(n)');
```

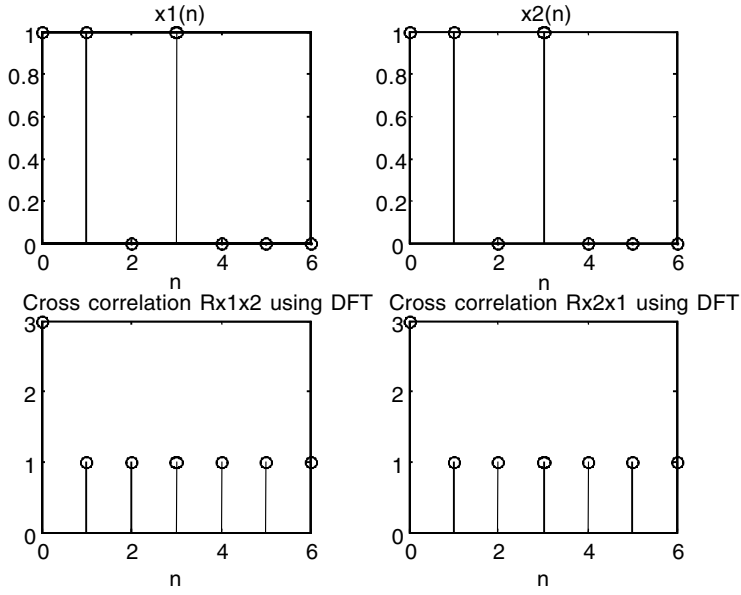

FIGURE 7.12 Plots for EOCE 7.7.

```
xlabel('n');
subplot(2,2,2);stem(n2x,x2n);title('x2(n)');
xlabel('n');
subplot(2,2,3);stem(n,rx1x2);title('cross-correlation
 between x1 and x2');
xlabel('n');
subplot(2,2,4);stem(n,rx2x1);title('cross-correlation
 between x2 and x1');
xlabel('n');
```

The plots are shown in Figure 7.13.

EOCE 7.8

Use MATLAB to generate an N-points random signal $x_1(n)$, then add to it the N-points samples of the 1000-Hz signal $x_2(n)$ where

$$x_2(n) = \sin\left(2\pi(1000)n/10000\right)$$

Find the energy spectral estimate of $x(n)$ where

$$x(n) = x_1(n) + x_2(x)$$

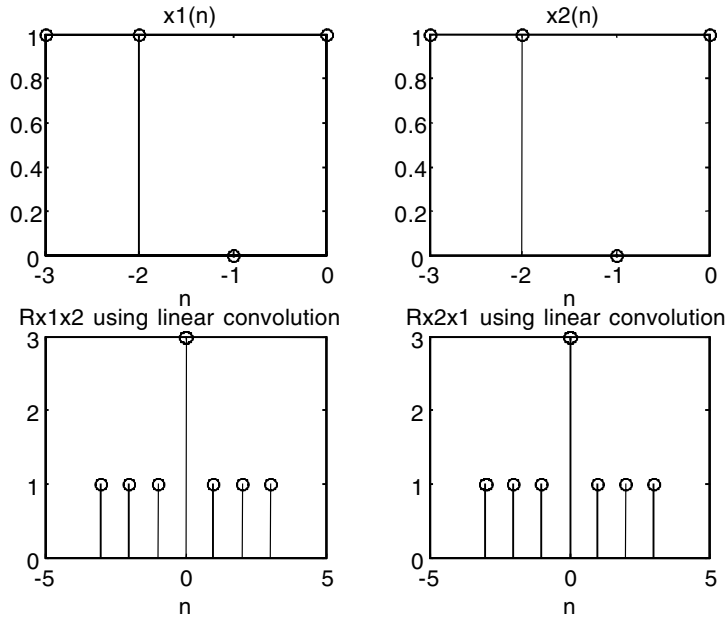

FIGURE 7.13 Plots for EOCE 7.7.

Solution

We will use MATLAB to generate 51 random values that are uniformly distributed between zero and one. The MATLAB script to calculate the spectral energy estimate follows.

```
%Generating the signal x(n)
n=0:50; %51 samples
x1n=rand(1,length(n));
X1k=fft(x1n);
x2n=sin(2*pi*1000*n/10000);% fs=10000Hz
X2k=fft(x2n);
xn=x1n+x2n;% the signal x(n)
Xk=fft(xn);
Xkconj=conj(Xk);
k=0:50;
ESDk=(Xk.*Xkconj)/length(n);% energy spectral density estimate
subplot(4,1,1);stem(k,X1k);ylabel('X1(k)');
subplot(4,1,2);stem(k,X2k);ylabel('X2(k)');
subplot(4,1,3);stem(k,Xk);ylabel('X1k(k)+X2k(k)');
subplot(4,1,4);stem(k,ESDk); xlabel('k');ylabel('Energy
 Density');
```

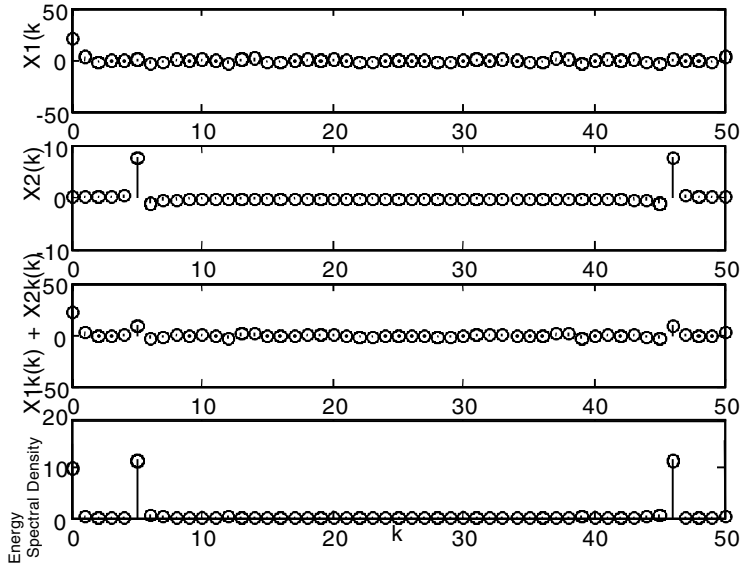

FIGURE 7.14 Plots for EOCE 7.8.

The plots are shown in Figure 7.14. You can clearly see that the 1-kHz signal is standing out while the noise is attenuated.

EOCE 7.9

Consider a discrete system with input

$$x(n) = e^{-n} \qquad 0 \le n \le 20$$

and impulse response

$$h(n) = 1 \qquad 0 \le n \le 1$$

Find the output $y(n)$ using block filtering.

Solution

The impulse response has 2 values and the input has 21 values. If we want to use 8-point DFT then we can divide $x(n)$ into 3 blocks each having 7 values in it. Thus 7 + 2 − 1 = 8 is the length of the DFT output as the result of convolving the 7-values input with the 2-values impulse response. Let us call the input blocks $x_1(n)$, $x_2(n)$ and $x_3(n)$. The output blocks are then $y_1(n)$, $y_2(n)$ and $y_3(n)$. Note that for 7-values input we will have 8-values output.

To find out the final output we will form the array $y(n)$ as a series of the subarrays $y_1(n)$, $y_2(n)$ and $y_3(n)$, where the eighth value of $y_1(n)$ will be added to the first value of $y_2(n)$ and the eighth value of $y_2(n)$ will be added to the first value of $y_3(n)$. This array of $y(n)$ can be viewed as

$$y(n) = \left[y_1(0) \ldots y_1(6)\ \left(y_1(7) + y_2(0)\right)\ y_2(1) \ldots y_2(6)\ \left(y_2(7) + y_3(0)\right)\ y_3(1) \ldots y_3(7)\right]$$

The following MATLAB script will produce $y(n)$ using block filtering with the DFT and regular linear convolution.

```
n=0:20;
xn=exp(-n);%the signal x(n) with 21 samples
x1n=xn(1:7);x2n=xn(8:14); x3n=xn(15:21);% the three blocks
% next we will make each block of length 8 by padding zeros
x1n=[x1n 0]; x2n=[x2n 0]; x3n=[x3n 0];
hn=[1 1]; %the impulse response
hne=[hn zeros(1,8-length(hn))]; %extending h(n) to of length 8
X1k=fft(x1n); X2k=fft(x2n); X3k=fft(x3n); Hk=fft(hne);
Y1k=X1k.*Hk; Y2k=X2k.*Hk; Y3k=X3k.*Hk;
y1n=ifft(Y1k); y2n=ifft(Y2k); y3n=ifft(Y3k);
yn=[y1n(1:7) y2n(1:7) y3n(1:8)];
yn(8)=y1n(8)+y2n(1); yn(15)=y2n(8)+y3n(1); % overlapping
%next we will use regular convolution to find y(n)
y=conv(xn,hn);
subplot(2,1,1);stem(n,y(1:21)); xlabel('n');
title('Output using linear convolution');
subplot(2,1,2);stem(n,yn(1:21)); xlabel('n');
title('Output using block filtering');
```

The plots are shown in Figure 7.15.

EOCE 7.10

High resolution and *dense spectrum* are two misleading terms. Next we consider an example to show you the difference. Consider the signal

$$x(n) = \sin(.37\pi n) + \sin(.55\pi n)$$

It is clear that $x(n)$ has the two frequencies at $.37\pi$ and $.55\pi$. Let us take only 20 samples of the 100 samples from $x(n)$ and find the spectrum. Then let us pad the 20-sample signal with 80 zeros and find the spectrum again. Next let us take all 100 samples from $x(n)$ and find the spectrum.

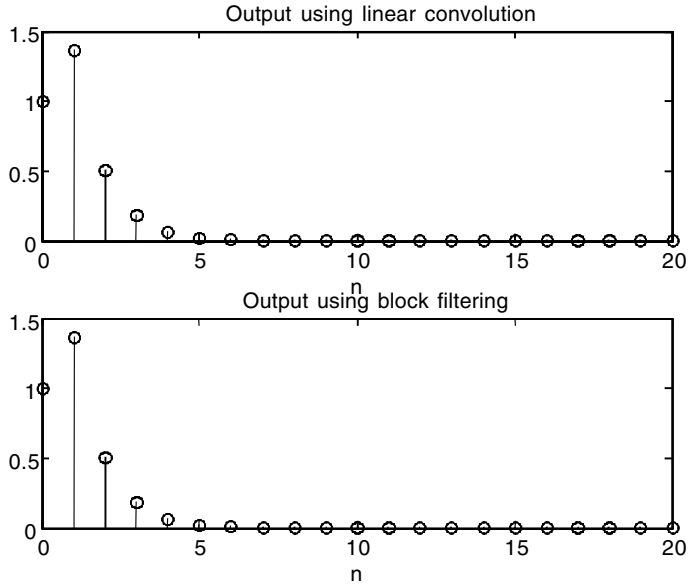

FIGURE 7.15 Plots for EOCE 7.9.

Solution

We will use MATLAB to accomplish this task.

```
n=0:1:99; %generating 100 samples
xn=sin(.37*pi*n)+sin(.55*pi*n);
x1n=xn(1:20); %taking only few points from x(n)
X1k=fft(xn);
k1=0:10; w1=k1*2*pi/20; %range from zero to 3.14 rad
%let us now padd x1(n) with 80 zeros
x2n=[x1n zeros(1,80)];
X2k=fft(x2n);
k2=0:50; w2=k2*pi*2/100;
%Now we take all 100 points
Xk=fft(xn);
k=0:1:50; w=k*pi*2/100;
n1=0:19;
subplot(2,3,1);stem(n,xn);xlabel('n');title('100 samples No
 padding')
subplot(2,3,2);stem(n1,x1n);xlabel('n');title('20 samples No
 padding')
subplot(2,3,3); stem(n, x2n);xlabel('n');title('20 samples 80
 zeros')
```

```
subplot(2,3,4); stem(w,abs(Xk(1:51)));xlabel('Frequency in
 radians');
title('Corresponding spectra');
subplot(2,3,5); stem(w1,abs(X1k(1:11)));xlabel('Frequency in
 radians');
title('Corresponding spectra');
subplot(2,3,6); stem(w2,abs(X2k(1:51)));xlabel('Frequency in
 radians');
title('Corresponding spectra');
```

The plots are shown in Figure 7.16. Notice that with only 20 samples of $x(n)$ and no zero padding, the spectrum is not dense as it is in the case when 20 samples are taken with 80 zeros padded. Padding with zeros will make the spectrum more dense. This does not mean that the DFT with zero padding will tell more about the frequency contents of the signal. As it is seen in the plot, with few samples and no zero padding and with few samples and zero padding, the magnitude plots are distorted and do not have values of the exact two frequencies of .37π and .55π. However, by taking more points from $x(n)$, we can see the spectrum emphasized at the .37π and.55π frequencies.

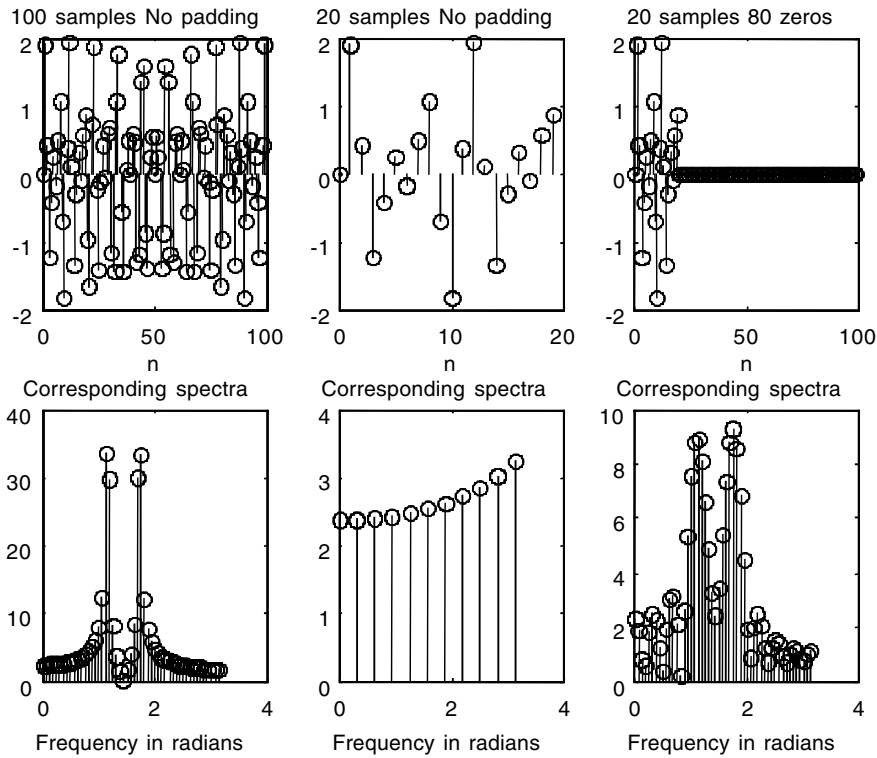

FIGURE 7.16 Plots for EOCE 7.10.

Thus we can say that zero padding can only make spectrum more dense which does not mean good frequency resolution. More points or samples will produce more frequency components and yet good frequency resolution. This is not to say that good frequency resolution comes from more samples only. You need to consider a good time span over which the signal is well known. If the signal is periodic, you can sample over one period. Taking more samples within this period gives good frequency resolution and yet more frequencies will be detected using the DFT. If the signal is not periodic but its amplitude approaches zero as time reaches a certain limit, then you need to sample the signal for the period of time up to that limit. This is what is known as the record length in digital signal processing.

7.10 End of Chapter Problems

EOCP 7.1

A continuous time signal has f_m as its highest frequency. If f_m = 1 kHz and we desire sampling the signal at 10 times f_m, what would be the record length and the number of samples if the frequency resolution is to be 10 Hz?

EOCP 7.2

Find the circular convolution between the signals

1. $u(n)$ for $0 \le n \le 5$ and $e^{-n/.1}$ for $0 \le n \le 5$
2. $5\sin(n\pi/3)$ for $0 \le n \le 4$ and e^{-n} for $0 \le n \le 4$
3. $e^{-n/2}$ and itself for $0 \le n \le 10$
4. $x(n) = 1$ and itself for $0 \le n \le 5$
5. $\cos(n\pi/6)$ for $0 \le n \le 6$ and $\delta(n)$ for $0 \le n \le 6$

EOCP 7.3

1. Find the discrete Fourier transform $X(e^{j\theta})$ of $x(n) = e^{-n/3}$ for $0 \le n \le 7$.
2. Sample $X(e^{j\theta})$ at $\theta = 2\pi k/N$ with $N = 16$.
3. Find the first 4 values $X(0)$,…, $X(3)$ using the DFT equation.
4. Find all eight values for $X(k)$ and compare with the values found in part 2.
5. If $x(n)$ in part 1 is an input to the linear system given by $h(n) = e^{-n/6}$ for $0 \le n \le 3$, use convolution to find $y(n)$, the output of the system.
6. Use the DFT to find $y(n)$ in part 5.

EOCP 7.4

Consider the signal

$$x(n) = \begin{cases} \cos(n\pi/6) & 0 \le n \le 5 \\ 0 & \text{otherwise} \end{cases}$$

1. Let N = 8, 16, 32 and 64. Find the spectrum for $x(n)$ using the DFT.
2. What conclusion can you draw by completing part 1?
3. If $x(n) = \cos(n\pi/6)$ for $0 \le n \le 63$, plot the spectrum for $x(n)$ in this case.
4. Compare the results of part 1 and part 3.

EOCP 7.5

Find the DFT of the following signals where n is taken in the interval $0 \le n \le N - 1$. A is a constant.

1. $A\delta(n)$
2. A
3. $A\sin(2\pi n/N)$
4. $A\cos(2\pi n/N)$

EOCP 7.6

Consider the signal

$$x(t) = \sin(600\pi t) + \sin(1000\pi t)$$

1. What is the period of $x(t)$?
2. What is the minimum sampling frequency?
3. Sample $x(t)$ for one period at $f_s = 10\, f_m$ and plot $x(n)$.
4. Find the DFT of $x(n)$ in 3.
5. Repeat 3 over two periods. What do you observe? Keep N as in 3.
6. Find the DFT of $x(n)$ in 5. What do you notice?

EOCP 7.7

Use the DFT to find the energy spectrum density for the signals

1. $x(n) = \sin\left(\dfrac{2\pi n}{11}\right) \quad 0 \le n \le 15$
2. $x(n) = e^{\frac{-n}{3}} \sin\left(\dfrac{2\pi n}{11}\right) 0 \le n \le 15$

EOCP 7.8

Use the DFT to find the cross-correlation between the signals given in EOCP 7.7.

EOCP 7.9

Use the DFT to approximate the Fourier transform for the signals

1. $x(t) = e^{-t}u(t)$
2. $x(t) = 20\cos\left(\dfrac{2\pi t}{13}\right)$
3. $x(t) = \begin{cases} 1 & 0 \le t \le 1 \\ 0 & \text{otherwise} \end{cases}$

EOCP 7.10

Consider the system

$$h(n) = e^{-n} \qquad 0 \le n \le 10$$

1. If $x(n) = u(n)$ for $0 \le n \le 5$, find $y(n)$ using linear convolution via the DFT.
2. Find the impulse response for the system.

EOCP 7.11

Consider the system

$$h(t) = e^{\frac{-t}{3}}u(t)$$

1. If the input is $x(t) = e^{-5t}u(t)$, use the DFT to find $y(n)$.
2. If the input is $x(t) = 10sin(2\pi(500)t) + 10$, find the output $y(n)$ using the DFT.
3. Is there any dc component in the output? Use the DFT to check.

EOCP 7.12

Consider the signals

$$x(t) = \sin(2\pi t) + \cos\left(\left(3\pi/4\right)t\right)$$

$$x(t) = e^{-10t}\sin(2\pi t) + \cos\left(3/4\,t\right)$$

1. Are the signals periodic?
2. Find the Fourier series coefficient and/or the Fourier transform (approximation) using the DFT for the signals.
3. What is the average power/total energy in the signals $x(t)$?

EOCP 7.13

Consider the signal

$$x(t) = e^{\frac{-t}{10}}u(t)$$

1. Find the total energy in the signal using the DFT.
2. If $x(t)$ is multiplied by $\sin(t)$, what would be the approximation to the total energy in the signal using the DFT.

EOCP 7.14

Consider the signal

$$x(t) = e^{-10t}\sin(t)u(t)$$

as an input to the system

$$h(t) = e^{-4t}u(t)$$

1. Find $y(n)$ using convolution and the DFT.
2. Subdivide the input signal into blocks and use block filtering to find $y(n)$ again.
3. Compare the results in 1 and 2.
4. Find the total energy in both signals using the DFT.

EOCP 7.15

Consider the signal

$$x(t) = \cos\left(2\pi(300)t\right)$$

1. Find the approximation to the Fourier transform of $x(t)$.
2. Use the Hamming windowing method and repeat part 1.
3. Use the Hamming windowing method and repeat part 1.
4. Comment on the results.

EOCP 7.16

Consider the signal

$$x(t) = 10\sin(2\pi(700)t) + \sin(2\pi(300)t)$$

1. Choose an f_s and a suitable N so that only the 700 and the 300 Hz will appear in the DFT magnitude plot. Plot the DFT magnitude.
2. Choose f_s (you should satisfy the sampling Nyquist rate in this part, too) and a suitable N so that the frequencies in $x(t)$ will appear distorted on the DFT magnitude plot. Plot the DFT magnitude.
3. How can you get a more accurate plot in part 2 to suppress the frequency distortion? Plot the DFT magnitude.

EOCP 7.17

A continuous signal has a duration of 2 sec and is sampled at 64 equally spaced values.

1. What is the distance between successive frequency points?
2. What is the highest frequency in the spectrum?
3. What is the maximum frequency in the continuous signal if there is to be no aliasing?

8

Analogue Filter Design

With Khaled Younis, Ph.D.

8.1 Introduction

The goal of this chapter is to review analogue filter design before we talk about digital filter design. This is important because most of the techniques for designing discrete time infinite impulse response (IIR) filters require the availability of suitable analogue lowpass filter (LPF) approximations. There are many established approximations for the design of analogue LPF. We will focus on five well-known approximation methods, namely, Butterworth, Chebyshev Type I, Chebyshev Type II, elliptic and Bessel filters.

The detailed derivation of these filters and their transfer functions can be found in texts on analogue filter design. First, we describe the specifications of the analogue lowpass filters. For each filter approximation we give the magnitude response, transfer function and the minimum order N needed to satisfy the analogue specifications. Then, the analogue frequency transformation is discussed. At the end, we describe the MATLAB functions that are useful for designing each of these filters and we provide some design examples. But first let us gain some understanding of the filter concept.

At this point in the course of this book we should have a very good understanding of the concept of input, output and a linear time-invariant system. We will stress again that if $x(t)$ is an input to a linear time-invariant system, the output will reshape the input due to the characteristics of the system itself. Let us look at the following system transfer function.

$$H(jw) = \frac{Y(jw)}{X(jw)} = \frac{1}{jw+1} \tag{8.1}$$

The magnitude of this transfer function is

$$\left|\frac{Y(jw)}{X(jw)}\right| = \left|H(jw)\right| = \frac{1}{\sqrt{1+w^2}} \tag{8.2}$$

At low frequencies (close to zero), the magnitude of the transfer function is close to unity. As the frequency increases, the magnitude decreases to a value close to zero.

It seems that this system will pass low frequencies and prevent high frequencies from passing through the system. This is similar to the process of filtering frequencies. Most input signals have frequencies. It seems to us that the impulse signal has no frequency, but if we take the Laplace transform of the impulse signal we see that the transform is a constant that is spread over all frequency values; thus the impulse signal has a constant value at all frequencies. If a signal contains both high and low frequencies, only frequencies in the low range will pass.

From the discussion above we see that the words *system* and *filter* are the same. In reality, the word system, and not filter, is used for electrical, mechanical and any dynamical systems.

Filters can be built using either active or passive circuit elements. The use of active or passive elements depends on the kind of the application being designed. In most communication systems we use active filters because inductors are heavy, very noisy and quite expensive. Passive filters do not produce an output of magnitude greater than the magnitude of the input. If high gain is required, we use active filters. Active filters are limited in terms of frequency. If we require filters that will operate on signals with frequencies very much higher than megahertz, we tend to use passive filters.

The phase angle of any transfer function has the lead-lag information that relates the input to the output. If the phase angle is positive, we say the output leads the input, and if the phase angle is negative, it means that the output lags behind the input. If the phase angle is zero, then the input and the output appear at the same time and we say the input and output are in-phase. The gain of any filter (the dc gain) can be calculated at the zero frequency.

Remember that we are dealing with steady-state analysis ($s = jw$) when we are dealing with the problem of designing filters. As the frequency approaches zero, w will approach zero, and thus s approaches zero. To increase or decrease the gain of a filter we simply multiply the transfer function by the desired constant. If the constant is positive, it will have no effect on the phase angle of the filter. If the constant is negative, 180° will be added to the phase of the filter. Thus the sign of the constant is important. Filters can have linear phase, meaning that the phase is a linear function of frequency.

8.2 Analogue Filter Specifications

An analogue filter can be represented by the equation

$$\frac{Y(s)}{X(s)} = H(s) = \frac{N(s)}{D(s)} \tag{8.3}$$

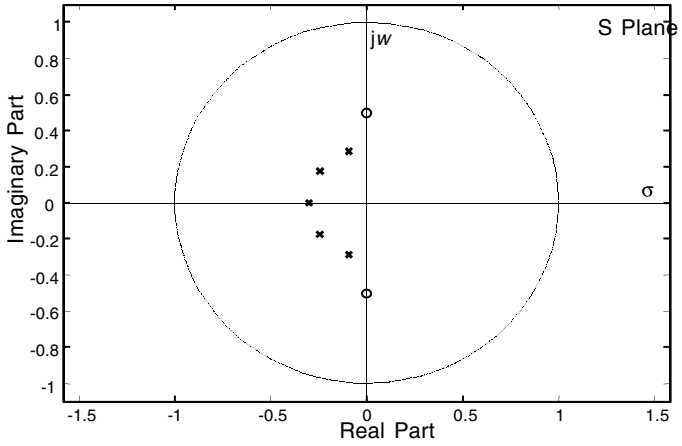

FIGURE 8.1 Typical analogue filter zero-pole plot.

where s is the complex variable in the Laplace transform ($s = \sigma + jw$), and $X(s)$ and $Y(s)$ are the Laplace transforms of the input and output signals $x(t)$ and $y(t)$, respectively. $H(s)$ is the transfer function of the analogue filter and $N(s)$ and $D(s)$ are polynomials in s. To obtain the specifications in analogue frequency w, we evaluate the transfer function at $s = jw$ to obtain $H(jw)$. The phase shift and group delay of the filter are given by

$$\theta(w) = \arg H(jw) \tag{8.4}$$

and

$$\tau(w) = \frac{d\theta(w)}{dw} \tag{8.5}$$

respectively. $\theta(w)$ and $\tau(w)$ are the phase and delay characteristics. A plot that shows the location of zeros and poles of an example $H(s)$ is shown in Figure 8.1. There are four common configurations for frequency-selective filters: a lowpass filter (LPF), a highpass filter (HPF), a passband filter (BPF), and a bandstop filter (BSF). The magnitude responses for these filter types are shown in Figure 8.2. Looking at the magnitude response we can easily see why these are called brick wall (or ideal) filters. For each of these filters only selected frequencies will pass — above w_c for the LPF and below w_c for the HPF. In the case of the BPF, frequencies above the lower cut-off frequency, w_{cl}, and below the upper cut-off frequency, w_{cu}, will pass. On the other hand, this frequency range will be rejected in the BSF case. In the next section we will focus on the design of common lowpass Butterworth filters. At the end of the chapter we will describe a method for transforming a lowpass filter with $w_c = 1$ rad/sec to other filter configurations.

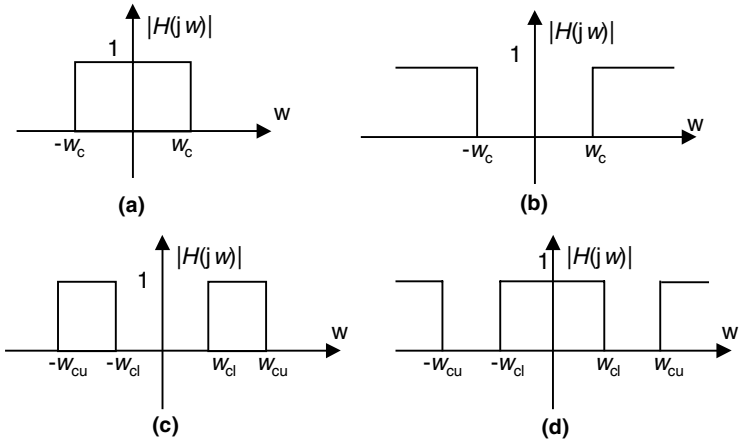

FIGURE 8.2 Ideal (brick wall) filters (a) LPF, (b) HPF, (c) BPF and (d) BSF.

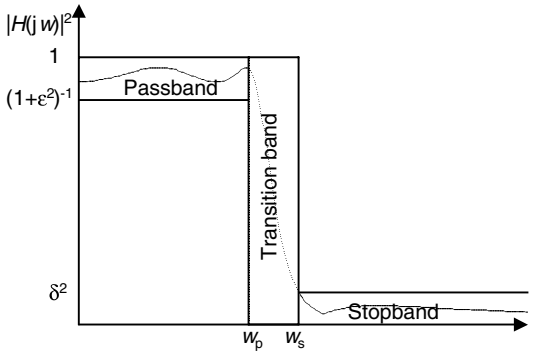

FIGURE 8.3 Practical lowpass filter specifications.

Since we cannot achieve the ideal lowpass filter shown in Figure 8.2(a), we need to allow for some ripple in the passband and the stopband. Therefore a magnitude response similar to the one shown in Figure 8.3 is usually obtained. Note that the magnitude-squared response can assume any value in the dashed area. In other words, the analogue lowpass filter should satisfy the following specifications

$$\begin{aligned} &\left(1+\varepsilon^2\right)^{-1} \le \left|H(jw)\right|^2 \le 1 \quad &&\text{for } 0 \le w \le w_p \quad &&\text{rad/sec} \\ &0 \le \left|H(jw)\right|^2 \le \delta^2 &&\text{for } w_s \le w \le \infty &&\text{rad/sec} \end{aligned} \tag{8.6}$$

where w_p and w_s are the passband and stopband edge frequencies, respectively. The passband ripple, denoted as $(1 + \varepsilon^2)^{-1}$ is given by the minimum value of the magnitude-squared in the passband. In the normalized form of the magnitude response, the maximum passband gain is unity. The maximum stopband ripple is given by δ^2.

Usually the ripples of the analogue filter are specified in decibels in terms of the peak passband ripple, R_p, and the minimum stopband attenuation, R_s, defined by

$$R_p = 10\log\left(1+\varepsilon^2\right) \tag{8.7}$$

$$R_s = -10\log\delta^2 \tag{8.8}$$

So far we have seen that there are four parameters of interest in the analogue filter specifications, namely, w_p, w_s, R_p and R_s. There are two other parameters that will be shown to be useful when we derive the equation to calculate the minimum order of the analogue filter to satisfy certain specifications. These are the lowpass filter transition ratio, or selectivity parameter, k, which is given by

$$k = \frac{w_p}{w_s} \tag{8.9}$$

The other parameter is called the discrimination parameter, d, and is given by

$$d = \frac{\varepsilon}{\sqrt{\delta^{-2}-1}} \tag{8.10}$$

8.3 Butterworth Filter Approximation

The magnitude-squared response of an N^{th} order analogue lowpass Butterworth filter is given by

$$\left|H(jw)\right|^2 = \frac{1}{1+\left(w/w_c\right)^{2N}} \tag{8.11}$$

where w_c is the frequency where the magnitude-squared drops by 3 dB. Therefore, it is called the 3-dB cut-off frequency. In other words, the magnitude response at w_c is always $\sqrt{1/2}$, regardless of the filter order N.

The Butterworth filter provides the best Taylor Series approximation to the ideal lowpass filter response at analogue frequencies $w = 0$ and $w = \infty$. The magnitude-squared response has $2N - 1$ zero derivatives at these locations. Hence it is called maximally flat at $w = 0$ and $w = \infty$. The response is monotonic overall, decreasing smoothly from $w = 0$ to $w = \infty$. Figure 8.4 shows the magnitude response $|H(jw)|$ of a lowpass Butterworth filter of order

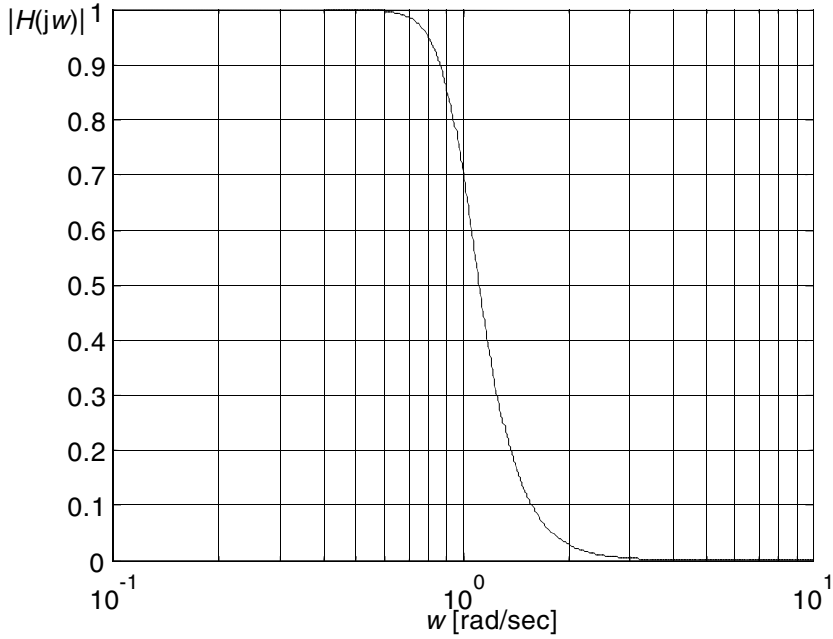

FIGURE 8.4 Magnitude response of a fifth-order Butterworth lowpass filter.

5. As can be seen in Figure 8.4, $|H(jw)| = \sqrt{1/2}$ at $w = 1$ and therefore the cut-off frequency of this filter is 1 rad/sec. The transfer function of an N^{th} order analogue Butterworth filter is given by

$$H(s) = \frac{c}{D_N(s)} = \frac{w_c^N}{\prod_{i=1}^{N}(s - p_i)} \tag{8.12}$$

where

$$p_i = w_c e^{\frac{j\pi}{2}[1+(2i-1)/N]} \tag{8.13}$$

and c normalizes the magnitude so that the maximum magnitude is unity. The transfer function has $2N$ poles equally spaced in angle on a circle of radius w_c in the left half s-plane that exist in complex conjugate pairs. The poles are symmetrically located with respect to the imaginary axis. The denominator polynomials have been tabulated in the analogue filter design texts for reference. However, as we shall see in Section 8.8, we can use MATLAB for designing such filters easily.

As can be seen from Equation (8.11), the two parameters that define the Butterworth LPF are the order N and the cut-off frequency w_c. Therefore we need to derive these parameters from the specifications. Recall that the main

constraints in Equation (8.6) are the value of the magnitude-squared response at w_p and w_s. In other words, at $w = w_p$

$$\left|H(jw)\right|^2 = \frac{1}{1+\left(w_p/w_c\right)^{2N}} = \frac{1}{1+\varepsilon^2} \tag{8.14}$$

and at $w = w_s$

$$\left|H(jw)\right|^2 = \frac{1}{1+\left(w_s/w_c\right)^{2N}} = \delta^2 \tag{8.15}$$

Equations (8.14) and (8.15) can be arranged into the equations

$$1+\left(w_s/w_c\right)^{2N} = \delta^{-2}$$

$$1+\left(w_p/w_c\right)^{2N} = 1+\varepsilon^2$$

or we write the equations as

$$\left(w_s/w_c\right)^{2N} = \delta^{-2}-1$$

$$\left(w_p/w_c\right)^{2N} = \varepsilon^2$$

We can divide the above equations to get

$$\left(w_s/w_p\right)^{2N} = \left(\delta^{-2}-1\right)/\varepsilon^2 = \left(\sqrt{\left(\delta^{-2}-1\right)}\right)^2/\varepsilon^2$$

If we solve the above equation for N we get

$$N \geq \frac{\log\left(\left[\sqrt{\left(\delta^{-2}-1\right)}\right]^2/\varepsilon^2\right)}{2\log\left[w_s/w_p\right]} = \frac{2\log\left(\left[\sqrt{\left(\delta^{-2}-1\right)}\right]/\varepsilon\right)}{2\log\left[w_s/w_p\right]} = \frac{\log\left(1/d\right)}{\log\left[1/k\right]} \tag{8.16}$$

Note that since N must be an integer value we round up the results of the right-hand expression to the next higher integer. The value of w_c can then be chosen anywhere from the following range of values

$$w_p\varepsilon^{-1/N} \leq w_c \leq w_s\left(\delta^{-2}-1\right)^{-1/(2N)} \tag{8.17}$$

The lower limit was found by solving Equation (8.14) while the upper limit was obtained by solving Equation (8.15).

8.4 Chebyshev Filters

Looking at the frequency response of the Butterworth filter given in Figure 8.4 we can obviously see that specifications are exceeded at frequencies close to $w = 0$ and $w = \infty$. This is due to the fact that the response is monotonic overall. Thus, to meet the filter specifications at w_c with minimum order, we need to far exceed the requirements at frequencies away from w_c. Therefore, the mean approximation error (defined as the difference between the ideal [brick wall] response and the actual frequency response) is minimal at $w = 0$ and $w = \infty$, and is high at w_c. Another approach is to distribute the approximation error over the entire passband or stopband regions maintaining the same maximum passband error or minimum stopband attenuation with lower filter order. There are approximation methods that have an equiripple behavior instead of a monotonic behavior. In the next section, we will describe Chebyshev Type I approximation which has the property that the magnitude of the frequency response is equiripple in the passband and monotonic in the stopband. On the other hand, Type II Chebyshev filter approximation is monotonic in the passband and equiripple in the stopband.

8.4.1 Type I Chebyshev Approximation

The Chebyshev Type I filter incorporates an equiripple of R_p dB in the passband and a monotonic behavior in the stopband. Hence, the absolute approximation error over the entire passband is minimized. Moreover, the transition from passband to stopband is more rapid than for the Butterworth filter.

The magnitude response for Type I Chebyshev filter is given by

$$\left|H(jw)\right|^2 = \frac{1}{1+\varepsilon^2 T_N^2\left(w/w_c\right)} \tag{8.18}$$

where $T_N(w)$ is the N^{th} order Chebyshev polynomial of the first kind

$$T_N(w) = \begin{cases} \cos\left(N\cos^{-1} w\right) & |w| \le 1 \\ \cosh\left(N\cosh^{-1} w\right) & |w| > 1 \end{cases} \tag{8.19}$$

The above polynomial can also be derived via the recurrence relation

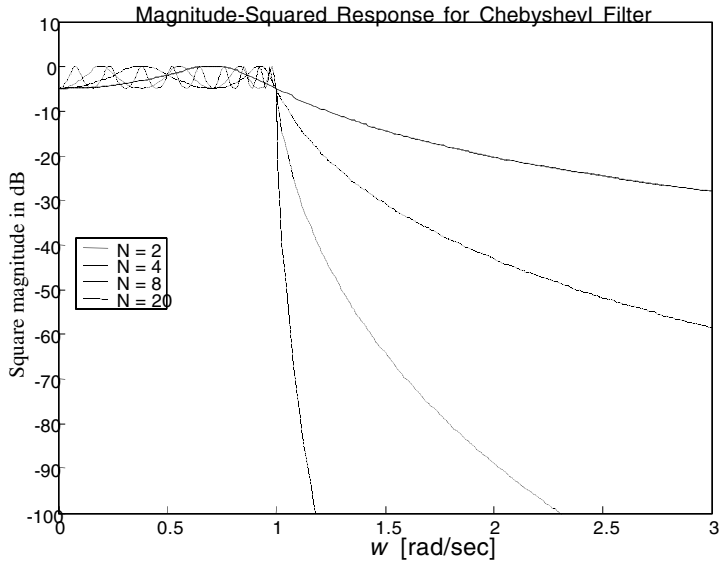

FIGURE 8.5 Magnitude-squared response of a Chebyshev Type I filter with different N values.

$$T_r(w) = 2wT_{r-1}(w) - T_{r-2}(w) \qquad r \geq 2 \tag{8.20}$$

with $T_0(w) = 1$ and $T_1(w) = w$. The Chebyshev Type I cut-off frequency w_c is the frequency at which the passband ends and the filter has magnitude response of $10^{-R_P/20}$. Typical plots of the magnitude responses of the Type I Chebyshev lowpass filter with w_c= 1 rad/sec are shown in Figure 8.5 for four different values of the filter order N with the same passband ripple ε. From these plots it is seen that the magnitude response is equiripple between w = 0 and w =1, and it is maximally flat for w > 1 rad/sec. The N^{th}-order analogue Type I Chebyshev filter has a transfer function given by

$$H(s) = \frac{c}{\prod_{i=1}^{N}(s - p_i)} \tag{8.21}$$

where

$$p_i = -w_p \sinh(\phi)\sin\left(\frac{2i-1}{2N}\pi\right) + jw_p \cosh(\phi)\cos\left(\frac{2i-1}{2N}\pi\right) \tag{8.22}$$

and

$$\phi = \frac{1}{N}\ln\left[\frac{1+\left(1+\varepsilon^2\right)^{\frac{1}{2}}}{\varepsilon}\right] \tag{8.23}$$

In Equation (8.22) the constant is given by

$$c = -\Pi_{i=1}^{N} p_i \tag{8.24}$$

when N is odd and

$$c = \left(1+\varepsilon^2\right)^{-\frac{1}{2}} \Pi_{i=1}^{N} p_i \tag{8.25}$$

when N is even. Note that c normalizes the magnitude so that the maximum magnitude is unity. The poles are evenly spaced about an ellipse in the left half s-plane.

Looking at Equation (8.18) we can see that the filter is completely specified by three parameters: ε, w_c and N. In a typical design, ε is specified by the allowable passband ripple and w_c is specified by the desired passband cut-off frequency. Then we choose the order N to meet the stopband specifications exactly at $w = w_s$ in Equation (8.6). This yields that the filter order required is

$$N \geq \frac{\log\left(d^{-1} + \sqrt{d^{-2}-1}\right)}{\log\left(k^{-1} + \sqrt{k^{-2}-1}\right)} = \frac{\cosh^{-1}\left(d^{-1}\right)}{\cosh^{-1}\left(k^{-1}\right)} \tag{8.26}$$

Example 8.1

Estimate the minimum order and cut-off frequency for a Chebyshev Type I analogue filter having the following specifications:

w_p = 1000 rad/sec	w_s = 3000 rad/sec
R_p = 2 dB	R_s = 60 dB

Solution

Using Equations (8.7) and (8.8) we substitute the given values to get

$$2 = 10\log\left(1+\varepsilon^2\right)$$

So, ε = 0.7648

$$60 = -10\log\left(\delta^2\right)$$

So, δ = 0.001. Now we find k and d as follows:

$$k = w_p / w_s = 1000/3000 = 0.333$$

$$d = \varepsilon / \sqrt{\delta^2 - 1} = 0.0007648$$

Using Equation (8.26), we get the minimum order N as

$$N = \frac{\cosh^{-1}\left(d^{-1}\right)}{\cosh^{-1}\left(k^{-1}\right)} = 4.4614$$

So the minimum filter order to achieve the above specifications is 5.

8.4.2 Inverse Chebyshev Filter (Type II Chebyshev Filters)

The Chebyshev Type II filter is monotonic in the passband. It minimizes the absolute approximation error over the entire stopband by incorporating an equiripple of R_s dB in the stopband. It is important that there is no ripple in the passband for the Type II Chebyshev filter. However, the stopband does not approach zero as quickly as the Type I filter (and does not approach zero at all for even-valued filter order N). The square-magnitude response expression is given by

$$\left|H(jw)\right|^2 = \frac{1}{1+\varepsilon^2\left[\dfrac{T_N\left(w_c/w_p\right)}{T_N\left(w_c/w\right)}\right]} = \frac{1}{1+\left[\dfrac{\delta^{-2}-1}{T_N\left(w_c/w\right)}\right]} \tag{8.27}$$

where for the Chebyshev Type II filter w_c is the frequency at which the stopband begins and the filter has magnitude response

$$\left|H(jw)\right| = 10^{-R_s/20} \tag{8.28}$$

The right-hand expression in Equation (8.27) was obtained by evaluating $\left|H(jw)\right|$ at $w = w_s$. Typical responses are as shown in Figure 8.6 for different values of N and $w_c = 1$ rad/sec. Note that a Type II Chebyshev lowpass filter can be related to a Type I filter through a transformation. Specifically, in Equation (8.18) if we replace the term $\varepsilon^2 T_N^2(w/w_p)$ by its reciprocal, and also replace the argument of $T_N^2(.)$ by its reciprocal we arrive at an equation equivalent to Equation (8.28). Actually, it is possible to design a Type I filter and then apply the above transformation to obtain the Type II Chebyshev filter. The transfer function giving rise to Equation (8.27) is

$$H_a(s) = \begin{cases} c\displaystyle\prod_{i=1}^{N}\frac{\left(s-q_i\right)}{\left(s-p_i\right)} & N \text{ is even} \\ \dfrac{c}{\left(s-p_{(N+1)/2}\right)}\displaystyle\prod_{\substack{i=1 \\ i\neq(N+1)/2}}^{N}\frac{\left(s-q_i\right)}{\left(s-p_i\right)} & N \text{ is odd} \end{cases} \tag{8.29}$$

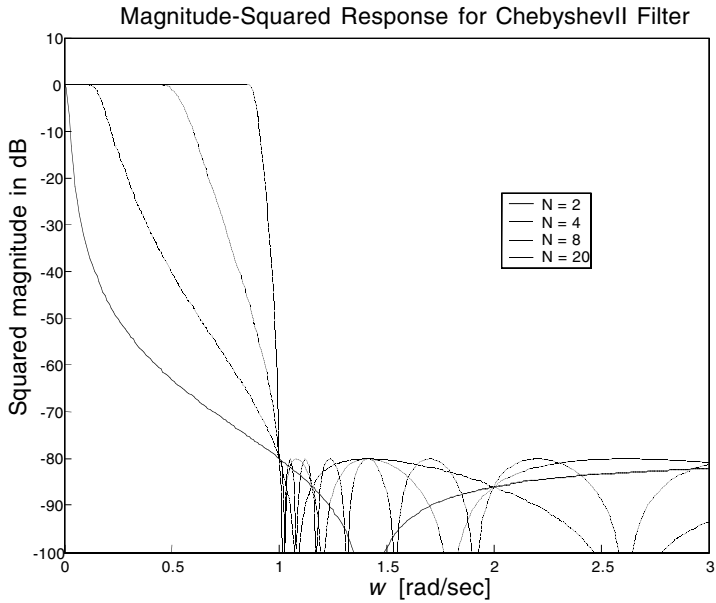

FIGURE 8.6 Magnitude-squared response of a Chebyshev Type II filter for different values of N.

where

$$p_i = \frac{w_s}{\alpha_i^2 + \beta_i^2}\left(\alpha_i - j\beta_i\right) \qquad q_i = j\frac{w_s}{\cos\left(\dfrac{2i-1}{2N}\pi\right)} \tag{8.30}$$

$$\alpha_i = -\sinh(\phi)\sin\left(\frac{2i-1}{2N}\pi\right) \tag{8.31}$$

$$\beta_i = -\cosh(\phi)\cos\left(\frac{2i-1}{2N}\pi\right) \tag{8.32}$$

$$\phi = \frac{1}{N}\cosh^{-1}\left(\delta^{-1}\right) = \frac{1}{N}\ln\left[\delta^{-1} + \left(\delta^{-2} - 1\right)^{\frac{1}{2}}\right] \tag{8.33}$$

and where

$$c = \prod_{i=1}^{N}\left(p_i / q_i\right) \tag{8.34}$$

when N is even and

$$c = -p(N+1)/2 \prod_{i=1, i\neq(N+1)/2}^{N} \left(p_i/q_i\right) \tag{8.35}$$

when N is odd. The pole locations are the inverse of the pole locations of Type I Chebyshev filter, whose poles are evenly spaced about an ellipse in the left half s-plane. The filter order N required to satisfy the filter specifications is the same as that for the Type I Chebyshev filter, which is given by Equation (8.26).

8.5 Elliptic Filter Approximation

We have noticed that by incorporating equiripple behavior in the passband or the stopband (as in the case of Chebyshev filters), the performance had improved. The improvement was in the sense of reducing the transition width ($w_s - w_p$) and minimizing the filter order required to achieve certain specifications for given values of ε, δ, and w_p.

Elliptic filters further improve the performance because they are equiripple in both the passband and stopband. Given a filter order N, passband ripple R_p in decibels and stopband ripple R_s in decibels, elliptic filters provide the steepest roll-off characteristics. This class of approximations, referred to as elliptic filters, has the form

$$\left|H_c(jw)\right|^2 = \frac{1}{1+\varepsilon^2 U_N^2(w)} \tag{8.36}$$

where $U_N(w)$ is a Jacobian elliptic function. The cut-off frequency is the frequency at which the passband ends and the filter has a magnitude response of $10^{-Rp/20}$. Figure 8.7 shows the magnitude-squared response of an elliptic filter for different values of N. The transfer function of an elliptic filter meets a given set of filter specifications, passband edge frequency w_p, stopband edge frequency w_s, passband ripple ε and minimum stopband attenuation δ, with the lowest filter order N. The theory of elliptic filter approximation is mathematically quite involved.

For most applications, the filter order meeting a given set of specifications w_s, w_p, ε and δ can be estimated using the approximate formula

$$N \cong \frac{2\log_{10}\left(4/d\right)}{\log_{10}\left(1/\rho\right)} \tag{8.37}$$

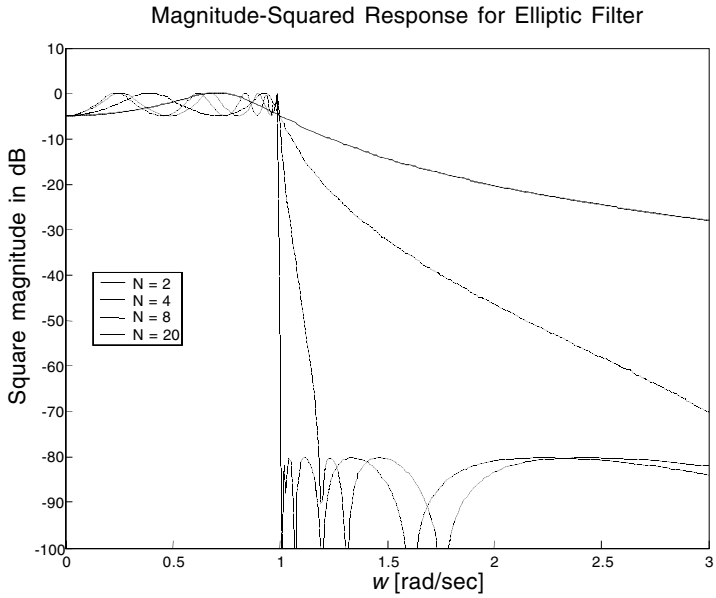

FIGURE 8.7 Magnitude-squared response of an elliptic filter for different values of N.

where d is the discrimination parameter defined in Equation (8.10) and ρ is computed as follows

$$k' = \sqrt{1-k^2} \tag{8.38}$$

$$\rho_o = \frac{1-\sqrt{k'}}{2\left(1+\sqrt{k'}\right)} \tag{8.39}$$

$$\rho = \rho_0 + 2\left(\rho_0\right)^5 + 15\left(\rho_0\right)^9 + 150\left(\rho_0\right)^{13} \tag{8.40}$$

where k is the selectivity parameter defined in Equation (8.9).

8.6 Bessel Filters

Note that the only objective in the preceding approximations is to achieve a specific magnitude response characteristic. As a result, the phase characteristic turns out to be nonlinear, i.e., the delay tends to vary with frequency, especially in the case of the elliptic approximation. It is desired to have a group delay of a filter independent of frequency. This is obtained if the phase

shift should be a linear function of frequency. One class of filters that tried to achieve this is Bessel filters which can have an all-pole transfer function of the form

$$H(s) = \frac{b_0}{\sum_{i=0}^{N} b_i s^i} = \frac{b_0}{s^N B(s)} \tag{8.41}$$

where $B(s)$ is called the Bessel polynomial and

$$b_i = \frac{(2N-i)!}{2^{N-i} i!(N-i)!} \tag{8.42}$$

The magnitude response, phase response and group delay can be found to be

$$\left|H(jw)\right|^2 = \frac{2b_0^2}{\pi w^{2N+1}\left[J_{-v}^2(w) + J_v^2(w)\right]} \tag{8.43}$$

$$\theta(w) = -w + \tan^{-1}\frac{(-1)^n J_v(w)}{J_{-v}(w)} \tag{8.44}$$

$$\tau(w) = -\frac{d\theta(w)}{dw} = 1 - \frac{(-1)^n\left(J_{-v}J_v' - J_v J_{-v}'\right)}{J_{-v}^2(w) + J_v^2(w)} \tag{8.45}$$

where $v = N+1/2$ and

$$J_v(w) = w^v \sum_{i=0}^{\infty} \frac{(-1)^i w^{2i}}{2^{2i+v} i!\,\Gamma(v+i+1)} \tag{8.46}$$

where $\Gamma(.)$ is the gamma function. Figure 8.8(a) shows the magnitude-squared response of a Bessel filter for different values of N. Note that the only parameter that defines the Bessel function is the order N, and the magnitude response is poorer than the previous approximations. The phase response here is linear, as can be seen in Figure 8.8(b).

Analogue Bessel lowpass filters have maximally flat group delay at zero frequency and retain nearly constant group delay across the entire passband. Filtered signals therefore maintain their wave shapes in the passband frequency range. Bessel filters generally require a higher filter order than other filters for satisfactory stopband attenuation. $\left|H(jw)\right| < \sqrt{1/2}$ at $w = w_c$ and decreases as filter order N increases. This is unlike the other approximations, where the magnitude response was fixed at w_c.

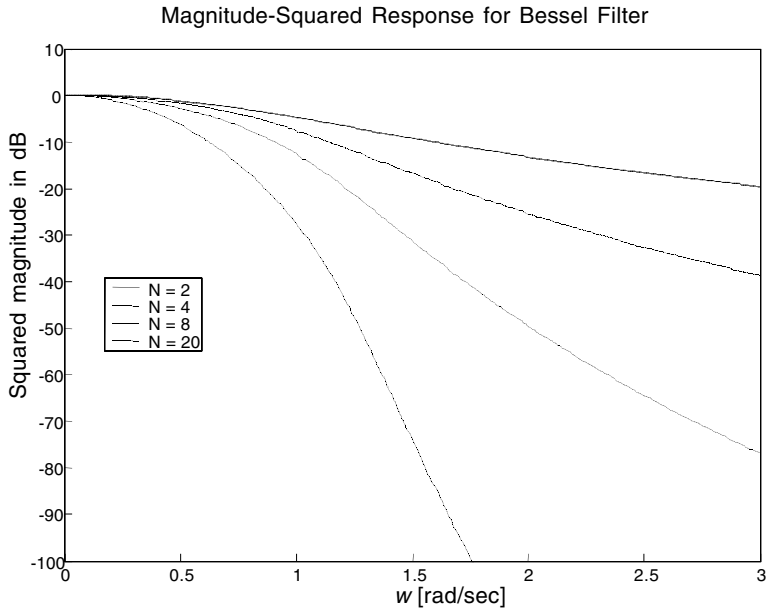

FIGURE 8.8(a) Frequency response of a Bessel filter for different values of N. (a) Magnitude-squared response; (b) phase response; (c) group delay response.

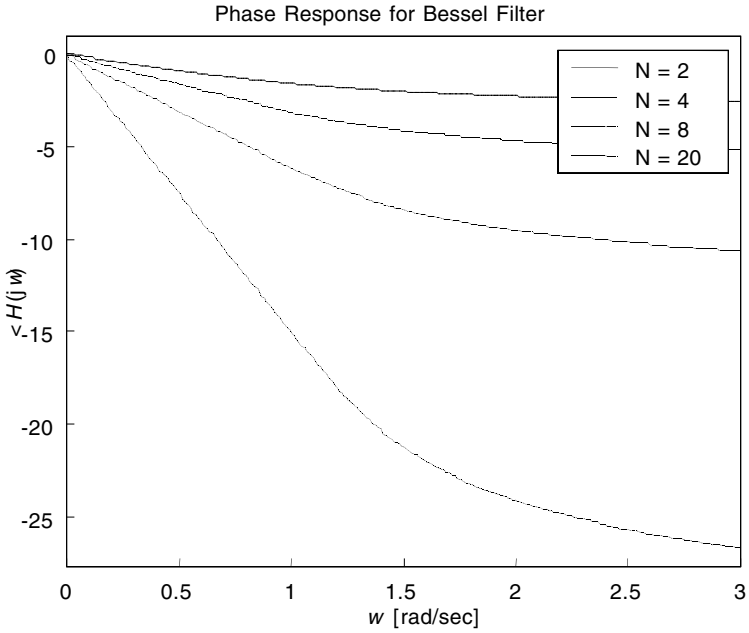

FIGURE 8.8(b)

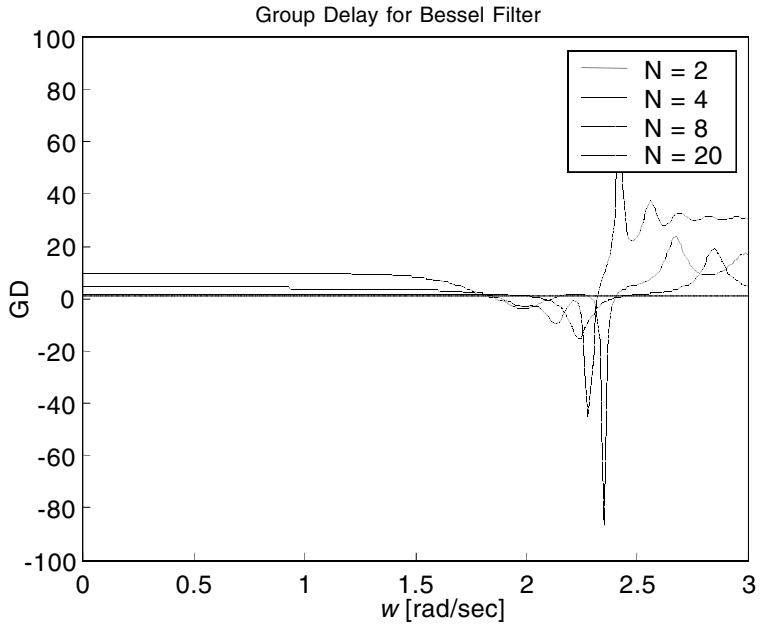

FIGURE 8.8(c)

8.7 Analogue Frequency Transformation

The preceding sections have discussed different approximations for the design of continuous-time lowpass filters. In this section we will discuss a method for transforming analogue lowpass prototypes (with cut-off frequency of 1 rad/sec) into bandpass, highpass, bandstop and lowpass filters of the desired cut-off frequency. The following is a set of frequency transformations transforming lowpass filters with system function $H(s')$ to a different configuration function $H(s)$.

$$\text{Lowpass to lowpass: } s' = \frac{s}{w_c} \tag{8.47}$$

$$\text{Lowpass to highpass: } s' = \frac{w_c}{s} \tag{8.48}$$

$$\text{Lowpass to bandpass: } s' = \frac{s^2 + w_{cl}w_{cu}}{s\left(w_{cu} - w_{cl}\right)} \tag{8.49}$$

$$\text{Lowpass to bandstop: } s' = \frac{s\left(w_{cu} - w_{cl}\right)}{s^2 + w_{cl}w_{cu}} \tag{8.50}$$

where in the above transformation w_{cl} is the lower cut-off frequency and w_{cu} is the upper cut-off frequency.

Looking at the previous transformations we can see that the transformation is not linear except for the lowpass to lowpass case. However, this nonlinearity provides no difficulty since the filter being transformed is approximately constant in the frequency band of interest. Therefore, while the frequency spacing of the ripple peaks and valleys is affected, the amount of ripple is still the same. Furthermore, in transforming the lowpass filter to bandstop or bandpass filter, the substitution is second order, so the output filter is twice the order of the input filter.

8.8 Analogue Filter Design using MATLAB

The signal processing toolbox of MATLAB has many useful functions that help in the design of analogue filters of the types described in the previous sections. These functions can be grouped into four categories:

1. Functions for the computation of minimum order required for classical IIR (infinite impulse response) filters to achieve certain specifications
2. Functions for the design of analogue lowpass prototype filters with a normalized cut-off frequency of 1 rad/sec
3. Functions for the complete design of analogue filters of any configuration; lowpass, highpass, bandpass, or bandstop, with varying w_p and w_s
4. Functions for transforming a 1 rad/sec prototype lowpass filter to other configurations

However, before we talk about these sets of functions, it is important to discuss the MATLAB function `freqs` which is used to compute the complex frequency response $|H(jw)|$ for an analogue filter. Given the numerator and denominator coefficients of the system function $H(s)$ in vectors b and a,

```
h = freqs(b,a,w)
```

returns the complex frequency response along the imaginary axis in the complex plane at the frequencies specified in real vector w. Note that there is a function called `bode` in the control toolbox and the system identification toolbox in MATLAB that computes the magnitude and phase of the frequency response of linear time-invariant (LTI) models. When invoked without

left-hand arguments, `bode` produces a Bode plot on the screen. The magnitude is plotted in decibels (dB), and the phase in degrees. The decibel calculation for the magnitude is computed as $20\log_{10}(|H(jw)|)$, where $|H(jw)|$ is the system's frequency response.

8.8.1 Order Estimation Functions

The Signal Processing Toolbox of MATLAB provides order selection functions that calculate the minimum order for an analogue filter that meets a given set of the requirements. These functions are: `buttord`, `cheb1ord`, `cheb2ord` and `ellipord` for the computation of the minimum order N required for Butterworth, Chebyshev Type I, Chebyshev Type II and elliptic filters, respectively. The following table summarizes these functions.

TABLE 8.1

Order Estimation Functions

Analogue Filter Type	Order Estimation Function
Butterworth	[n,Wn] = `buttord`(Wp,Ws,R_p,R_s ,'s')
Chebyshev Type I	[n,Wn] = `cheb1ord`(Wp, Ws, R_p, R_s, ,'s')
Chebyshev Type II	[n,Wn] = `cheb2ord`(Wp, Ws, R_p, R_s ,'s')
Elliptic	[n,Wn] = `ellipord`(Wp, Ws, R_p, R_s, ,'s')

The filter specifications are given as

Wp	Passband corner frequency; it is the cut-off frequency in rad/sec and may have a value above 1
Ws	Stopband corner frequency in rads per second
Rp	Passband ripple, in decibels; this value is the maximum permissible passband loss in decibels
Rs	Stopband attenuation, in decibels; this value is the number of decibels the stopband is down from the passband

`buttord` selects the minimum order digital or analogue Butterworth filter required to meet a set of filter design specifications.

These functions also work with the other standard band configurations, as well as for digital filters as we shall see in Chapter 10. For highpass filters, Wp is greater than Ws. For bandpass and bandstop filters, Wp and Ws are two-element vectors that specify the corner frequencies at both edges of the filter, lower frequency edge first. For the band filters, the output Wn is a two-element row vector. For the complete description of use of these and all MATLAB functions the reader is encouraged to see MATLAB's Function Reference for details.

These functions are useful in conjunction with the filter design functions in the sense that before we design a filter we need to estimate the minimum order required to satisfy a specific requirement and then use the output order N and w_n as input for the filter design functions.

8.8.2 Analogue Prototype Design Functions

There are a number of functions in MATLAB used to create lowpass analogue prototype filters with a cut-off frequency of 1 rad/sec. The table below summarizes the analogue prototype design function for each supported filter type.

TABLE 8.2

Analogue Prototype Functions

Filter Type	Analogue Prototype Function
Bessel	[z,p,k] = `besselap`(n)
Butterworth	[z,p,k] = `buttap`(n)
Chebyshev Type I	[z,p,k] = `cheb1ap`(n,Rp)
Chebyshev Type II	[z,p,k] = `cheb2ap`(n,Rs)
Elliptic	[z,p,k] = `ellipap`(n,Rp,Rs)

The above functions return the zeros, poles and gain of the assigned filter. They return the poles in the length n column vector p and the gain in scalar k. In [z,p,k] = `buttap`(n), z is an empty matrix because there are no zeros. In [z,p,k] = `cheb1ap`(n,Rp), z is an empty matrix because there are no zeros. Passband ripple Rp dB should be specified. In [z,p,k] = `cheb2ap`(n,Rs), if n is odd, z is of length n – 1 and if n is even, z is of length n. The value of Rs in dB should be specified. In [z,p,k] = `ellipap`(n,R_p,R_s), if n is odd, z is of length n – 1 and if n is even, z is of length n. The values of Rs and Rp in decibels should be specified. In [z,p,k] = `besselap`(n), z is an empty matrix, because there are no zeros.

8.8.3 Complete Classical IIR Filter Design

You can easily create an analogue filter of any order with a lowpass, highpass, bandpass, or bandstop configuration using the filter design functions shown in Table 8.3. By default, each of these functions returns a lowpass filter; you need only specify the desired cut-off frequency `Wn` in rad/sec. However, they can design lowpass, bandpass, highpass and bandstop analogue filters. They return the filter coefficients in the length `n + 1` row vectors `b` and `a`, with coefficients in descending powers of s

$$H(s) = \frac{B(s)}{A(s)} = \frac{b(1)s^n + b(2)s^{n-1} + \cdots b(n+1)}{S^n + a(2)s^{n-1} + \cdots + a(n+1)} \tag{8.51}$$

`[b,a] = besself(n,Wn)` designs an order n lowpass analogue filter with cut-off frequency `Wn`.

`[b,a] = butter(n,Wn,'s')` designs an order `n` lowpass analogue Butterworth filter with cut-off frequency `Wn`.

TABLE 8.3

Complete Design Functions

Filter Type	Design Function
Bessel	`[b,a] = besself(n,Wn)`
	`[b,a] = besself(n,Wn,'ftype')`
	`[z,p,k] = besself(n,Wn,...)`
	`[A,B,C,D] = besself(n,Wn,...)`
Butterworth	`[b,a] = butter(n,Wn,'s')`
	`[b,a] = butter(n,Wn,'ftype','s')`
	`[z,p,k] = butter(n,Wn,...)`
	`[A,B,C,D] = butter(n,Wn,...)`
Chebyshev Type I	`[b,a] = cheby1(n,Rp,Wn,'s')`
	`[b,a] = cheby1(n,Rp,Wn,'ftype','s')`
	`[z,p,k] = cheby1(n,Rp,Wn,...)`
	`[A,B,C,D] = cheby1(n,Rp,Wn,...)`
Chebyshev Type II	`[b,a] = cheby2(n,Rs,Wn,'s')`
	`[b,a] = cheby2(n,Rs,Wn,'ftype','s')`
	`[z,p,k] = cheby2(n,Rs,Wn,...)`
	`[A,B,C,D] = cheby2(n,Rs,Wn,...)`
Elliptic	`[b,a] = ellip(n,Rp,Rs,Wn,'s')`
	`[b,a] = ellip(n,Rp,Rs,Wn,'ftype','s')`
	`[z,p,k] = ellip(n,Rp,Rs,Wn,...)`
	`[A,B,C,D] = ellip(n,Rp,Rs,Wn,...)`

`[b,a] = cheby1(n,Rp,Wn,'s')` designs an order n lowpass analogue Chebyshev Type I filter with cut-off frequency `Wn` that equals w_p.

`[b,a] = cheby2(n,Rs,Wn,'s')` designs an order `n` lowpass analogue Chebyshev Type II filter with cut-off frequency `Wn` that equals w_s.

`[b,a] = ellip(n,Rp,Rs,Wn,'s')` designs an order `n` lowpass analogue elliptic filter with cut-off frequency `Wn` that equals w_s.

If `Wn` is a two-element vector, `Wn = [w1 w2]` with `w1 < w2`, then these commands return an order `2n` bandpass analogue filter with passband `w1` < w < `w2`. If the input argument `ftype` is used, then we can design a highpass or bandstop filter, where `ftype` is

1. high for a highpass analogue filter with cut-off frequency `Wn`
2. stop for an order 2n bandstop analogue filter if `Wn` is a two-element vector, `Wn = [w1 w2]`; the stopband is `w1` < ω < `w2`

With different numbers of output arguments, the complete filter design functions directly obtain other realizations of the analogue filter. We have already dealt with the case of two output arguments that return the coefficient of the numerator and denominator of $H(s)$. To obtain zero-pole-gain form, use three output arguments, e.g.,

```
[z,p,k] = butter(n,Wn,'s')
```

or

```
[z,p,k] = butter(n,Wn,'ftype','s')
```

which will return the zeros and poles in length `n` or `2n` column vectors `z` and `p` and the gain in the scalar `k`.

To obtain state-space form, use four output arguments, e.g.,

```
[A,B,C,D] = butter(n,Wn,'s')
```

or

```
[A,B,C,D] = butter(n,Wn,'ftype','s')
```

where A, B, C and D are the state-space system matrices

$$\frac{d}{dt}\mathbf{v} = \mathbf{A}\mathbf{v} + \mathbf{B}\mathbf{x}$$

$$\mathbf{y} = \mathbf{C}\mathbf{v} + \mathbf{D}\mathbf{x}$$

and **x** is the input vector, **v** is the state vector, and **y** is the output vector.

8.8.4 Analogue Frequency Transformation

The signal processing toolbox provides a set of functions to transform a 1-rad/sec cut-off frequency analogue lowpass filter to another analogue lowpass filter (with different cut off frequency) or a bandpass, highpass or bandstop filter. These are shown in Table 8.4.

As shown, all of the frequency transformation functions can accept two linear system models: transfer function and state-space form. For the bandpass and bandstop cases `w0` is the center frequency and equals $\sqrt{w_{cl}w_{cu}}$ and `Bw` is the frequency bandwidth that equals $w_{cu} - w_{cl}$, where w_{cl} is the lower cut-off frequency and w_{cu} is the upper cut-off frequency.

TABLE 8.4

Transformation Functions

Frequency Transformation	Transformation Function
Lowpass to lowpass	`[numt,dent] = lp2lp(num,den,W0)`
	`[At,Bt,Ct,Dt] = lp2lp(A,B,C,D,W0)`
Lowpass to highpass	`[numt,dent] = lp2hp(num,den,W0)`
	`[At,Bt,Ct,Dt] = lp2hp(A,B,C,D,W0)`
Lowpass to bandpass	`[numt,dent] = lp2bp(num,den,W0,Bw)`
	`[At,Bt,Ct,Dt]=lp2bp(A,B,C,D,W0,Bw)`
Lowpass to bandstop	`[numt,dent]=lp2bs(num,den,W0,Bw)`
	`[At,Bt,Ct,Dt]=lp2bs(A,B,C,D,W0,Bw)`

8.9 How Do We Find the Cut-Off Frequency Analytically?

In filter design and analysis we calculate the cut-off frequency w_c as in the following. Given the transfer function $H(jw)$, we set

$$\frac{\max\left(\left|H(jw)\right|\right)}{\sqrt{2}} = \left|H(jw)\right|_{w_c} \tag{8.52}$$

We can solve this equation by first finding the maximum value of the magnitude of $H(jw)$ and dividing that value by $\sqrt{2}$, and then setting it equal to $|H(jw)|$ evaluated at w_0, where w_0 is the frequency at which the magnitude is maximum. The maximum value for $H(jw)$ can be calculated either by using calculus (find the derivative of $H(jw)$ and setting it equal to zero) or by plotting the magnitude of $H(jw)$ vs. w and locating the maximum value.

Example 8.2

Consider the transfer function

$$\frac{Y(jw)}{X(jw)} = H(jw) = \frac{1}{jw+1}$$

Calculate w_c, the cut-off frequency.

Solution

We see that the maximum amplitude is 1 by evaluating the magnitude at $w = 0$. We now use Equation (8.52) and write

$$\frac{\max\left(\left|H(jw)\right|\right)}{\sqrt{2}} = \left|H(jw)\right|_{w_c}$$

or

$$\frac{1}{\sqrt{2}} = \left|H(jw)\right|_{w_c} = \frac{1}{\sqrt{1+w_c^2}}$$

Squaring both sides gives

$$\frac{1}{2} = \frac{1}{1+w_c^2}$$

Solving the above equation, by considering the positive side of the w-axis, gives $w_c = 1$. Using calculus, we can find the maximum magnitude of the transfer function by finding the derivative of $|H(jw)|$ with respect to w as

$$\frac{d}{dw}\left(\frac{1}{\sqrt{1+w^2}}\right) = \frac{-2w}{2\left(1+w^2\right)^{3/2}}$$

Setting the above equation equal to zero results in $w = 0$. At $w = 0$, $|H(0)| = 1$, which is the maximum magnitude of $|H(w)|$. Following the procedure above, we can solve for w_c to get the same result: $w_c = 1$.

Example 8.3

Consider the transfer function

$$\frac{Y(jw)}{X(jw)} = H(jw) = \frac{jw}{jw+1}$$

Calculate w_c, the cut-off frequency.

Solution

The maximum amplitude occurs at a very high frequency (∞) and it is unity. We now use Equation (8.52) to find the cut-off frequency and write

$$\frac{\max\left(|H(jw)|\right)}{\sqrt{2}} = |T(jw)|_{w_c}$$

$$\frac{1}{\sqrt{2}} = |H(jw)|_{w_c} = \frac{w_c}{\sqrt{1+w_c^2}}$$

Squaring both sides gives

$$\frac{1}{2} = \frac{w_c^2}{1+w_c^2}$$

Solving the above equation, while considering the positive side of the w-axis, gives $w_c = 1$. Using calculus, we can find the derivative of $|H(jw)|$ with respect to w and get

$$\frac{d}{dw}\left(\frac{w}{\sqrt{1+w^2}}\right) = \left(\frac{2}{2\left(1+w^2\right)^{1/2}}\right) \Big/ \left(1+w^2\right)$$

Setting the above equation equal to zero (the denominator cannot be zero) results in

$$\left(\frac{2}{2\left(1+w^2\right)^{1/2}}\right)=0$$

This last equation results in $w = \infty$. At $w = \infty$, $|H(j\infty)| = 1$, which is the maximum magnitude of $|H(jw)|$. Following the procedure above, we can solve for w_c to get the same result: $w_c = 1$.

Example 8.4

Consider the system described by

$$H(s)=\frac{(R/L)s}{s^2+(R/L)s+1/LC}$$

Calculate the cut-off frequency.

Solution

With $s = jw$, the magnitude of the transfer function can be calculated as

$$|H(jw)|=\frac{w\dfrac{R}{L}}{\sqrt{\left(1/LC-w^2\right)^2+\left(w\dfrac{R}{L}\right)^2}}$$

This is a bandpass filter. In case of a bandpass filter, we will look at the following parameters:

1. The cut-off frequencies w_{cl} and w_{cu}. These can be calculated using the same procedure that we used to determine the cut-off frequencies for the lowpass and the highpass filters.
2. The center frequency w_0. The center frequency is the frequency where the transfer function $H(s)$ is purely real. It is also the frequency at which the magnitude of $H(s)$ is maximum, as well as the geometric mean of w_{cl} and w_{cu} or

$$w_0 = \sqrt{w_{cl}w_{cu}} \tag{8.53}$$

3. The bandwidth β. The bandwidth is the width of the frequency band in the bandpass filter.

4. The quality factor Q. The quality factor is defined as the ratio of the center frequency to the bandwidth

$$Q = w_0/\beta \tag{8.54}$$

It is a measure of the bandwidth of the passband.

The center frequency can be calculated by making $H(s)$ purely real. We can do that by setting

$$jw_0L + \frac{1}{jw_0C} = 0$$

Solving the above equation leads to

$$w_0 = \sqrt{\frac{1}{LC}}$$

We next calculate the cut-off frequencies w_{cl} and w_{cu}. $H(jw)$ has its maximum at w_0. Therefore, the maximum magnitude of $H(jw)$ is $|H(jw_0)|$.

$$\left|H(jw_0)\right| = \frac{\sqrt{\frac{1}{LC}}\frac{R}{L}}{\sqrt{\left(1/LC - 1/LC\right)^2 + \left(\sqrt{\frac{1}{LC}}\frac{R}{L}\right)^2}} = 1$$

To find w_{cl} and w_{cu} we divide $|H(jw_0)|$ by the square root of 2 and set it equal to $|H(jw)|$ as

$$\left|H(jw)\right| = \frac{w\frac{R}{L}}{\sqrt{\left(1/LC - w^2\right)^2 + \left(w\frac{R}{L}\right)^2}} = \frac{1}{\sqrt{2}}$$

With some manipulations we get the following quadratic equation in w:

$$w^2L \pm wR - 1/C = 0$$

By solving the above equation we get the two cut-off frequencies

$$w_{cl} = -\frac{R}{2L} + \sqrt{\left(R/2L\right)^2 + \left(1/LC\right)}$$

$$w_{cu} = \frac{R}{2L} + \sqrt{\left(R/2L\right)^2 + \left(1/LC\right)}$$

For the bandwidth β we have

$$\beta = w_{cu} - w_{cl} = R/L$$

and for the quality factor

$$Q = w_0/\beta = \sqrt{\frac{L}{CR^2}}$$

8.10 Limitations

The following limitations are met in filter design using MATLAB.

1. All classical IIR lowpass filters are ill-conditioned for extremely low cut-off frequencies. Therefore, instead of designing a lowpass IIR filter with a very narrow passband, it may be better to design a wider passband and decimate the input signal.
2. For `besselap`, n must be less than or equal to 25. `besselap` finds the filter roots from a look-up table constructed using the Symbolic Math Toolbox.
3. For high-order filters, the state-space form is the most numerically accurate, followed by the zero-pole-gain form. The transfer function coefficient form is the least accurate; numerical problems can arise for filter orders as low as 15.
4. If filter specifications call for a bandpass or bandstop filter with unequal ripple in each of the passbands or stopbands, design the filter as separate lowpass and highpass sections and cascade the two filters together.

8.11 Comparison between Analogue Filter Types

Butterworth filters sacrifice roll-off steepness for smoothness due to monotonicity in the passband and the stopband. An elliptic or Chebyshev filter can generally provide steeper roll off characteristics with a lower filter order.

Unless the smoothness of the Butterworth filter is needed, use other filter types.

Chebyshev Type I filters are equiripple in the passband and monotonic in the stopband. Type I filters roll off faster than Type II filters, but at the expense of greater deviation from unity in the passband. The benefit of Type II Chebyshev filters is that they are smooth in the passband.

Elliptic filters offer steeper roll-off characteristics than Butterworth or Chebyshev filters, but are equiripple in both the passband and stopband. In general, elliptic filters meet given performance specifications with the lowest order of any filter type.

Lowpass Bessel filters have a monotonically decreasing magnitude response, as do lowpass Butterworth filters with a poorer performance in the magnitude response. Compared with the Butterworth, Chebyshev and elliptic filters, the Bessel filter has the slowest roll-off and requires the highest order to meet an attenuation specification. Nevertheless, we should not forget that Bessel filters provide us with the benefit of linear phase response.

8.12 Some Insights: Filters with High Gain vs. Filters with Low Gain and the Relation between the Time Constant and the Cut-Off Frequency for First-Order Circuits and the Series RLC Circuit

If we are interested in designing filters with a gain higher than unity we need to use circuits with active elements. Circuits with all passive elements, in most cases, give us a maximum gain of unity. If we are interested in using passive circuits with gains less than unity, we can place additional loads across the output terminals. If we seek circuits with gain higher than unity, we need to use circuits with active elements such as operational amplifiers. The time constant, τ, for first-order filters is the parameter that characterizes the shape of the transient response. For RL and RC filters, the time constants are *L/R* and *RC*, respectively. We noticed that for RL or RC filters the cut-off frequency is $w_c = 1/\tau$.

For an RLC series circuit, the overdamped, underdamped and critically damped transients are determined by the neper frequency α, and the resonant frequency w_0. The neper frequency is *R/2L* rad/sec and the resonant frequency is

$$\sqrt{\frac{1}{LC}} \text{ rad/sec}$$

α and w_0 are time domain characteristics. The bandwidth, $\beta = 2\alpha$, is a frequency characteristic. Recall that the quality factor $Q = w_c/\beta$. The time

when the response becomes underdamped is when Q = ½. As Q increases in value the peak at w_0 becomes sharper and the bandwidth smaller. This indicates an underdamped response. If Q is low and β is wide, it is an indication of overdamped transient response.

8.13 End of Chapter Examples

EOCE 8.1

A lowpass filter has a desired peak passband ripple of 0.1 dB, and a minimum stopband attenuation of 50 dB; what are the values of ε and δ?

Solution

Using Equation (8.7), we substitute $R_p = 0.1$ to get

$$0.1 = 10\log\left(1+\varepsilon^2\right)$$

$$10^{0.01} = 1+\varepsilon^2$$

$$\varepsilon = \sqrt{10^{0.01}-1} = \sqrt{0.0233} = 0.1526$$

and substituting R_s =50 dB in Equation (8.8) we have

$$50 = -10\log\delta^2$$

By solving for δ we get

$$\delta^2 = 10^{-5}$$

$$\delta = 10^{-5/2} = 0.0032$$

These conversions can be applied using MATLAB by the functions `Rp2Epsilon.m` and `Rs2Delta.m`, respectively. The first function is:

```
function Epsilon=Rp2Epsilon(Rp)
% Converts the peak passband ripple to its corresponding value
% of Epsilon
Epsilon=sqrt((10^(Rp/10)-1));
```

The other function is

```
function Delta=Rs2Delta(Rs)
% Converts the minimum stopband ripple to its corresponding
% value of Sigma
Delta=sqrt(10^-(Rs/10));
```

We can verify the results obtained above by typing

```
Epsilon=Rp2Epsilon(.1)
```

at the command line prompt and hitting the return key. Similarly we can verify the 0.0032 value by typing

```
Delta=Rs2Delta(50)
```

at the command line prompt.

EOCE 8.2

The goal of this example is to demonstrate the use of the function `freqs` and to show the different forms in which the magnitude response and phase response can be displayed in MATLAB. Find and graph the frequency response of the transfer function given by

$$H(s) = \frac{0.2s + 1}{s^2 + 0.1s + 1}$$

Solution

The magnitude and phase response can be displayed using the function `freqs` as in the following MATLAB script.

```
a = [1 0.1 1];
b = [0 0.2 1];
w = logspace(-1,1);
freqs(b,a,w)
```

The plot is shown in Figure 8.9. You can also create the plot with the script

```
h = freqs(b,a,w);
mag = abs(h);
phase = angle(h);
subplot(2,1,1), loglog(w,mag)
subplot(2,1,2), semilogx(w,phase)
```

To convert to Hertz, degrees and decibels, use

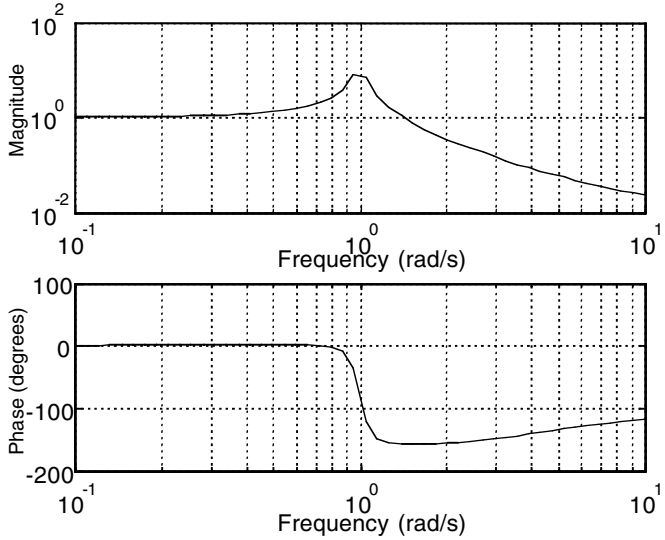

FIGURE 8.9 Frequency response computations using the `freqs` command.

```
f = w/(2*pi);
mag = 20*log10(mag);
phase = phase*180/pi;
```

The frequency response for the system (Figure 8.10) can be obtained using the `bode` function as in `Bode(b, a, w);`

EOCE 8.3

Use MATLAB to estimate the orders and cut-off frequencies for all filter types having the following specifications:

w_p = 1000 rad/sec w_s = 3000 rad/s
R_p = 2 dB R_s = 60 dB

Solution

The following MATLAB code finds the minimum order and cut-off frequency for any type of filter. Run the MATLAB file `order_estimation.m` once for every type of filter. After you type the following code and save it as `order_estimation.m`, type `order_estimation` at the command line prompt.

```
% Example8_3 : order_estimation.m
function [N,Wc] = order_estimation
% Order Estimation of analogue filters.
```

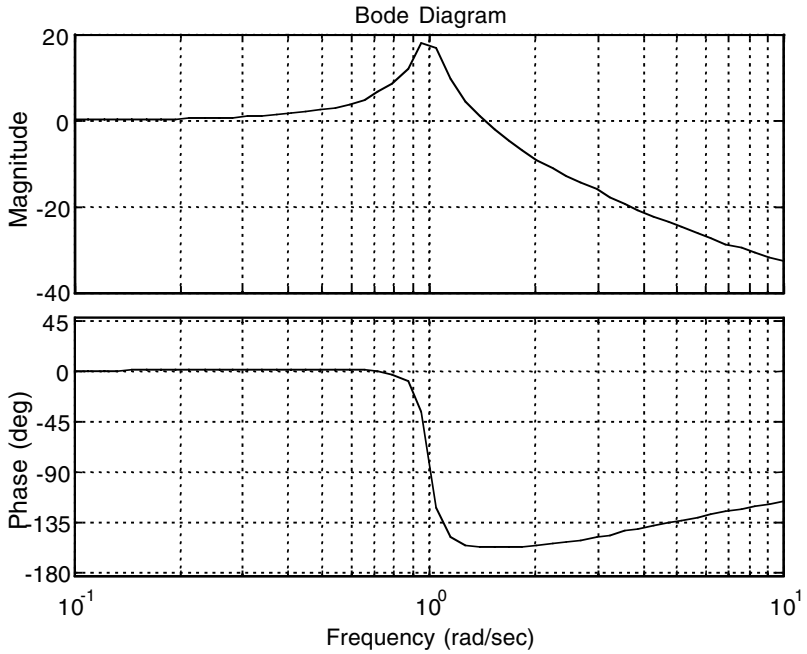

FIGURE 8.10 Frequency response computations using the `bode` command.

```
% Get filter type
FilterTypes{1}='Butterworth';
FilterTypes{2}='ChebyshevI';
FilterTypes{3}='ChebyshevII';
FilterTypes{4}='Elliptic';
for i=1:4
   disp([int2str(i) ') ' FilterTypes{i}])
end
Type = input('Analogue filter type: '); Wp = input('Pass band
 frequency (rad/sec):');
% Get specifications
Ws = input('Stop band frequency (rad/sec):');
Rp = input('Pass band Ripple (dB):');
Rs = input('Stop band Ripple (dB):');
switch Type
case 1 % Butterworth
   [N, Wc] = buttord( Wp, Ws , Rp , Rs, 's');
case 2 %Chebyshev 1
   [N, Wc] = cheb1ord( Wp, Ws , Rp , Rs, 's');
case 3 % Chebyshev 2
   [N, Wc] = cheb2ord( Wp, Ws , Rp , Rs, 's');
```

```
case 4 % Elliptic
    [N, Wc] = ellipord( Wp, Ws , Rp , Rs, 's');
end
% End of program
```

A sample run is

```
>> order_estimation
1) Butterworth
2) ChebyshevI
3) ChebyshevII
4) Elliptic
Analogue filter type: 1
Pass band frequency (rad/sec):1000
Stop band frequency (rad/sec):3000
Pass band Ripple (dB):2
Stop band Ripple (dB):60
ans = 7
```

The following table shows the obtained results:

TABLE 8.5

	Butterworth	Chebyshev 1	Chebyshev 2	Elliptic
N	7	5	5	4
w_c	1118.3	1000	2516.1	1000

Note that the least order is in the case of elliptic filter. This is due to the equiripple response in both passband and stopband regions.

EOCE 8.4

Study the effect of the filter order N on the characteristics of the Butterworth filter.

Solution

To study the dependence of the Butterworth filter characteristics on the parameter N we draw the magnitude-squared response given in Equation (8.11) for $w_c = 1$ and evaluate for $N = 2$, 4, 8 and 50. We build the MATLAB function `frequency_response.m`, which can be used to show the magnitude-squared response of analogue filters for different values of N to give an indication about the effect of filter order on the response. We specify that we want to show the magnitude-squared response of a Butterworth filter, and the curves are generated. The results are shown in Figure 8.11. As can be seen in the figure, as N increases the filter magnitude response becomes

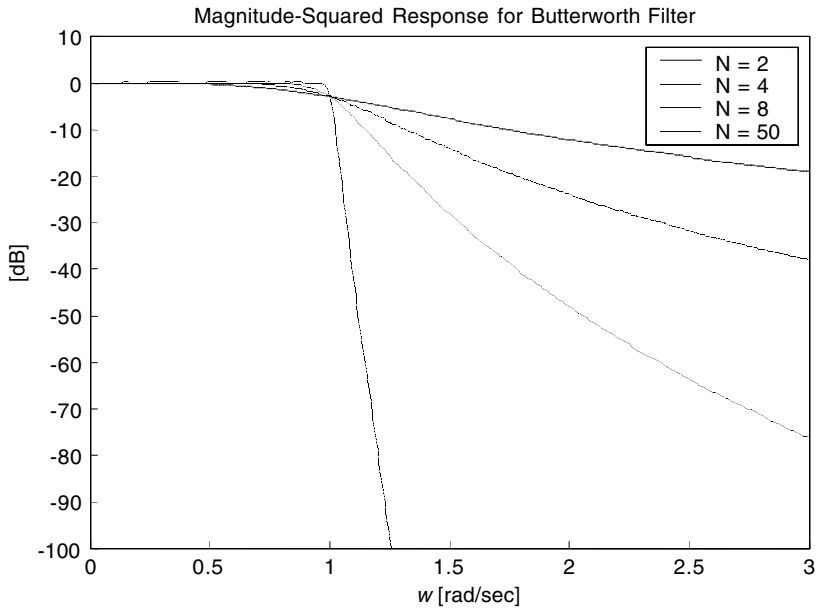

FIGURE 8. 11 Magnitude-squared response of a Butterworth filter for different N.

sharper, i.e., the transition from the passband region to the stopband region takes less time. However, for all cases, the magnitude-squared response at w_c is equal to –3 dB.

```
% frequency_response.m
% Calculates the frequency response of common analogue low
% pass filters.
% Displays the magnitude,phase response and the group delay.
% The function is able to calculate frequency responses for
% several orders at a time.
function frequency_response
% Get filter order
Order = input('Enter up to 7 filter orders, Default [2 4 8
 20]): ');
if isempty(Order)
    Order=[2 4 8 20];
end
NumOrders=length(Order);
% Get filter type
FilterTypes{1}='Butterworth';
FilterTypes{2}='ChebyshevI';
```

```
FilterTypes{3}='ChebyshevII';
FilterTypes{4}='Elliptic';
FilterTypes{5}='Bessel';
for i=1:5
     disp([int2str(i) ') ' FilterTypes{i}])
end
Type = input('Analogue filter type: ');
% Get Specifications
switch Type
case {1 ,5}
     % No specifications needed
case 2
     Rp = input('Peak Passband Ripple [in dB] : ');
case3
     Rs = input('Minimum Stopband Ripple [in dB] : ');
case 4
     Rp = input('Peak Passband Ripple [in dB] : ');
     Rs = input('Minimum Stopband Ripple [in dB] : ');
otherwise
     error('Invalid Filter Type')
end
ShowPhase = input('Plot phase?(enter number: 1-yes,2-no):');
% Values of frequency to estimate the response at
Omega = [0:0.01:3];
% Design the filter
switch Type
case 1 % Butterworth Filter
     figure;hold on;
     for i=1:NumOrders
          [z{i} p{i} k{i}] = buttap(Order(i));
     end
case 2 % Chebyshev Filter 1
     for i=1:NumOrders
          [z{i} p{i} k{i}] = cheb1ap( Order(i) , Rp);
     end
case 3 % Chebyshev ilter 2
     for i = 1 : length ( Order )
          [z{i} p{i} k{i}] = cheb2ap ( Order(i) , Rs ) ;
     end
case 4 % Elliptic Filter
     for i=1:NumOrders
          [z{i} p{i} k{i}] = ellipap(Order(i),Rp,Rs);
     end
```

```
case 5 % Bessel Filter
     for i=1:NumOrders
          [z{i} p{i} k{i}] = besselap(Order(i));
     end
end
% Display Magnitude-sqared response in dBs vs. analogue
% frequency in rad/sec
Shape = {'k-','k:','k--','k-.','k:','k.','k'};
figure
hold on;
for i=1:NumOrders
     [pz pp] = zp2tf(z{i},p{i},k{i});
     h = freqs(pz, pp, Omega);
     gain = 20 * log10(abs(h));
     plot(Omega, gain,Shape{i});
end
axis([0 3 -100 10]);
xlabel('\omega [rad/sec]')
ylabel('H(j\omega)^2| [dB]')
if NumOrders>1
     for i=1:NumOrders
          LegArray{i}=['N = ' int2str(Order(i))];
     end
     legend(LegArray);
     title(['Magnitude-Squared Response for '
 FilterTypes{Type} ' Filter'])
else
     title(['Magnitude-Squared Response for '
 FilterTypes{Type} ' Filter of order ' int2str(Order)])
     grid
end
hold off;
if ShowPhase == 1
     figure;hold on;
     for i=1:NumOrders
          [pz pp] = zp2tf(z{i},p{i},k{i});
          h = freqs(pz, pp, Omega);
          phase = unwrap(angle(h));
          plot(Omega, phase,Shape{i});
     end
```

```
        axis([0 3 min(phase)-1 max(phase)+1]);
        xlabel('\omega [rad/sec]')
        ylabel('<H(j\omega)')
        if NumOrders>1
            for i=1:NumOrders
                LegArray{i}=['N = ' int2str(Order(i))];
            end
            legend(LegArray);
            title(['Phase Response for ' FilterTypes{Type} '
       Filter'])
        else
                title(['Phase Response for ' FilterTypes{Type} '
       Filter of order ' int2str(Order)])
                grid
            end
            figure;hold on;
            for i=1:NumOrders
                [pz pp] = zp2tf(z{i},p{i},k{i});
                [Gd,W] = GRPDELAY(pz,pp,Omega)
                plot(W,Gd,Shape{i});
            end
            axis([0 W(end) -100 100]);
            xlabel('\omega [rad/sec]')
            ylabel('GD')
            if NumOrders>1
            for i=1:NumOrders
                LegArray{i}=['N = ' int2str(Order(i))];
        end
            legend(LegArray);
            title(['Group Delay for ' FilterTypes{Type} '
       Filter'])
            else
            title(['Group Delay for ' FilterTypes{Type} ' Filter
       of order ' int2str(Order)])
            grid
        end
    end
    disp('Make changes to the figure before saving if desired')
    disp('Click any key to continue ...')
    pause
    eval([' print -djpeg n_' FilterTypes{Type}])
```

A sample run of the program is given next.

```
>> frequency_response
Enter up to 7 filter orders, Default [2 4 8 20]): [2 4 8 50]
1) Butterworth
2) ChebyshevI
3) ChebyshevII
4) Elliptic
5) Bessel
Analogue filter type: 1
Plot phase?(enter number: 1-yes,2-no):2
Make changes to the figure before saving if desired
Click any key to continue ...
>>
```

EOCE 8.5

Design an analogue elliptic LP filter of order N equal to 6 and cut-off frequency equal to 1. Use MATLAB to design the filter and to generate the frequency response graphs.

Solution

Run the MATLAB program frequency_response. Enter 6 for order, 4 for Type and 2 for showing phase. The following is the sample run of the program that we just displayed in EOCE 8.4.

```
>> frequency_response
Enter up to 7 filter orders, Default [2 4 8 20]): 6
1) Butterworth
2) ChebyshevI
3) ChebyshevII
4) Elliptic
5) Bessel
Analogue filter type: 4
Peak Passband Ripple [in dB] : 5
Minimum Stopband Ripple [in dB] : 60
Plot phase?(enter number: 1-yes,2-no):2
Make changes to the figure before saving if desired
Click any key to continue ...
>>
```

The resulting frequency response plot is shown in Figure 8.12.

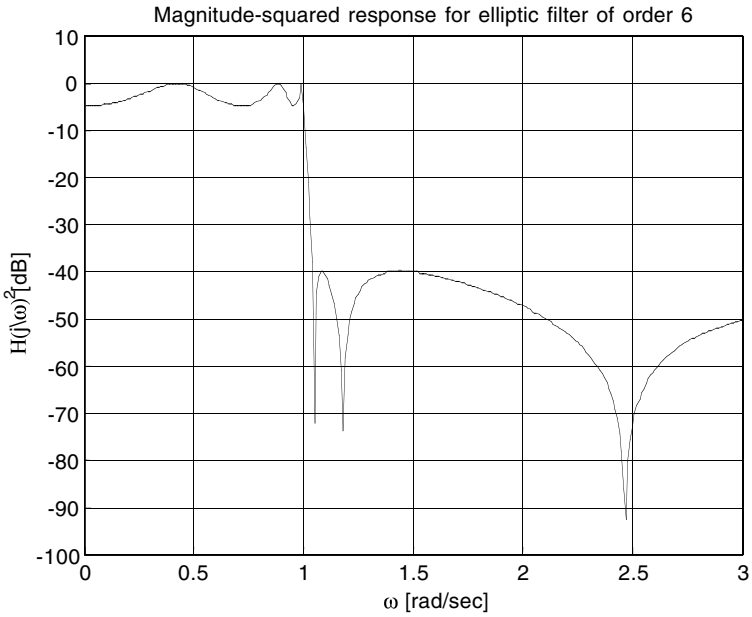

FIGURE 8.12 Magnitude-squared response of an elliptic filter of order 6 with R_p = 5 dB and R_s = 60 dB.

EOCE 8.6

Show using MATLAB that

1. If we choose w_c to equal the lower limit in Equation (8.17) then the passband specifications are met exactly whereas the stopband specifications are exceeded.
2. If we choose w_c to equal the upper limit in Equation (8.17) then the stopband specifications are met exactly while the passband specifications are exceeded.

Solution

We start by having certain analogue LPF specifications. These specifications are entered into the MATLAB script `EOCE8_6`. For example we choose the following set of values

w_p = 1000 rad/sec $\quad w_s$ = 3000 rad/s
R_p = 10dB $\quad R_s$ = 50 dB

These values are stored as the default values in the `EOCE8_6` file, so these values will be selected if you hit return when the program asks you for

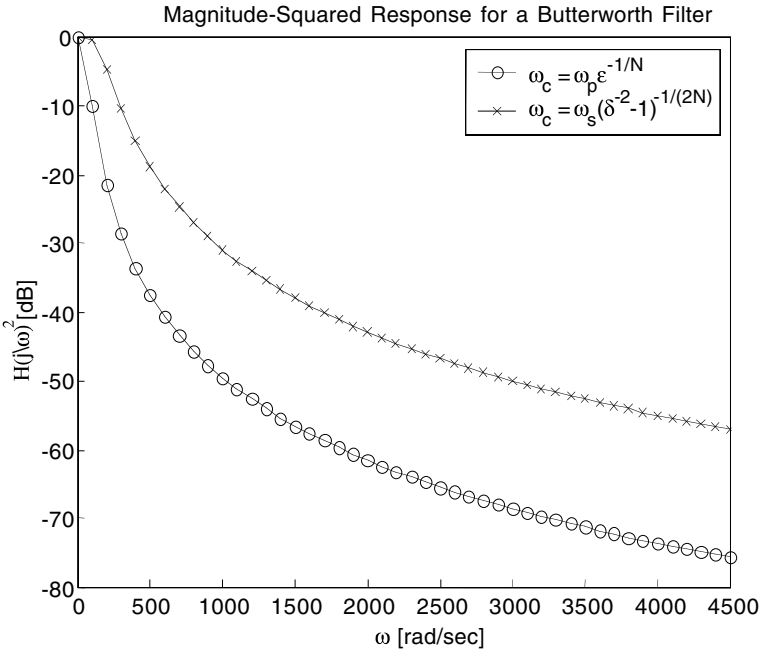

FIGURE 8.13 Plots for EOCE 8.6.

specifications. The program then computes the minimum order N needed for the Butterworth filter to satisfy the requirements. It then computes w_c according to Equation (8.17) and plots the magnitude-squared response based on it. The results are shown as the continuous curve in Figure 8.13. We can see that the passband ripple is met exactly whereas the stopband specifications are exceeded. The script `EOCE8_6` follows.

```
% EOCE8_6.m
% For a Butterworth filter, the order will be estimated and
% omega_c will be given two extreme values that ensure meeting
% the specifications of passband and stopband frequencies
% respectively.
spec = input('Enter specifications of filter ([Wp Ws Rp
 Rs]):');
if isempty(spec)
     spec = [1000 3000 10 50];
end
Wp = spec(1);Ws = spec(2); Rp = spec(3); Rs = spec(4);
Epsilon=sqrt((10^(Rp/10)-1));
Delta=sqrt(10^-(Rs/10));
[N Wn_est] = buttord(Wp,Ws,Rp,Rs,'s');
```

```
Wn_pass = Wp * Epsilon^(-1/N);
Wn_stop = Ws * (Delta^(-2) - 1)^(-1/(2*N));
Omega = [0:1.5*Ws];
[z_pass p_pass k_pass] = butter(N,Wn_pass,'s');
[z_stop p_stop k_stop] = butter(N,Wn_stop,'s');
[pz_p pp_p]=zp2tf(z_pass, p_pass, k_pass);
[pz_s pp_s]=zp2tf(z_stop, p_stop, k_stop);
h2 = freqs(pz_p,pp_p, Omega);
h3 = freqs(pz_s,pp_s, Omega);
figure
hold on;
plot(Omega(1:100:end),20*log10(abs(h2(1:100:end))),'ko-');
plot(Omega(1:100:end), 20*log10(abs(h3(1:100:end))),'kx-');
legend({'\omega_c = \omega_p\epsilon^{-1/N}',...
     '\omega_c = \omega_s(\delta^{-2}-1)^{-1/(2N)}'});
xlabel('\omega [rad/sec]')
ylabel('|H(j\omega)^2| [dB]')
title(['Magnitude-Squared Response for a Butterworth
 Filter'])
hold off;
```

A sample run is

```
>> Eoce8_6
Enter specifications of filter ([Wp Ws Rp Rs]):[100 3000 10 50]
>>
```

The plots are shown in Figure 8.13.

EOCE 8.7

Design the filters whose specifications are given in Table 8.5. Plot the magnitude responses of these filters.

Solution

The following MATLAB scripts can be run to obtain the results. For the case of Butterworth filter:

```
% Example 8.7.1
% Design of analogue LP Butterworth filter.
N = 7; Wc = 1118.3;
[b , a] = butter( N , Wc , 's');
Omega = [0:4000];
h = freqs( b , a , Omega);
```

```
plot(Omega, 20 * log10( abs( h ) ));
axis([0 4000 -80 10]);
Title(' Butterworth Filter');
```

For Chebyshev filter Type I:

```
% Example 8.7.2
% Design of analogue LP Chebyshev Type I filter
N = 5; Wc = 1000;
[b , a] = cheby1( N , 2 , Wc , 's');
Omega = [0:4000];
h = freqs( b , a , Omega);
plot(Omega, 20 * log10( abs( h ) ));
axis([0 4000 -80 10]);
Title(' Chebyshev 1 Filter');
```

For Chebyshev filter Type II:

```
% Example 8.7.3
% Design of analogue LP Chebyshev Type II filter
N = 5; Wc = 2516.1;
[b , a] = cheby2( N , 60 , Wc , 's');
Omega = [0:4000];
h = freqs( b , a , Omega);
plot(Omega, 20 * log10( abs( h ) ));
axis([0 4000 -80 10]);
Title(' Chebyshev 2 Filter');
```

For elliptic filter:

```
% Eample 8.7.4
% Design of analogue LP Elliptic filter
N = 4; Wc = 1000;
[b , a] = ellip( N , 2 , 60 , Wc , 's');
Omega = [0:4000];
h = freqs( b , a , Omega);
plot(Omega, 20 * log10( abs( h ) ));
axis([0 4000 -80 10]);
Title(' Elliptic Filter');
```

The plots are shown in Figures 8.14, 8.15, 8.16, and 8.17.

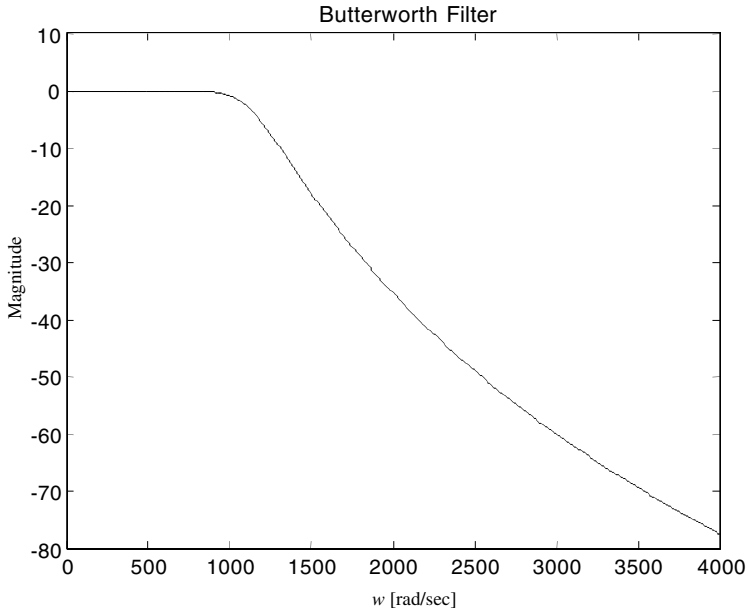

FIGURE 8.14 Magnitude response of Butterworth filter of order 7 and w_n = 1118.3 rad/sec.

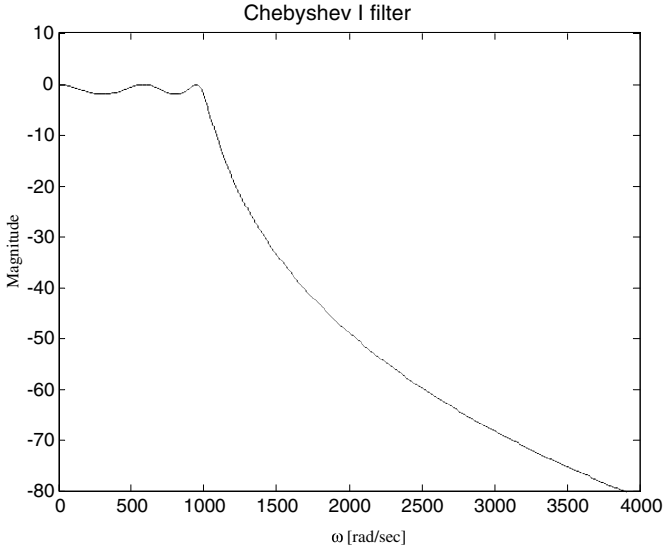

FIGURE 8.15 Magnitude response of Chebyshev Type I filter of order 5 and w_n = 1000 rad/sec.

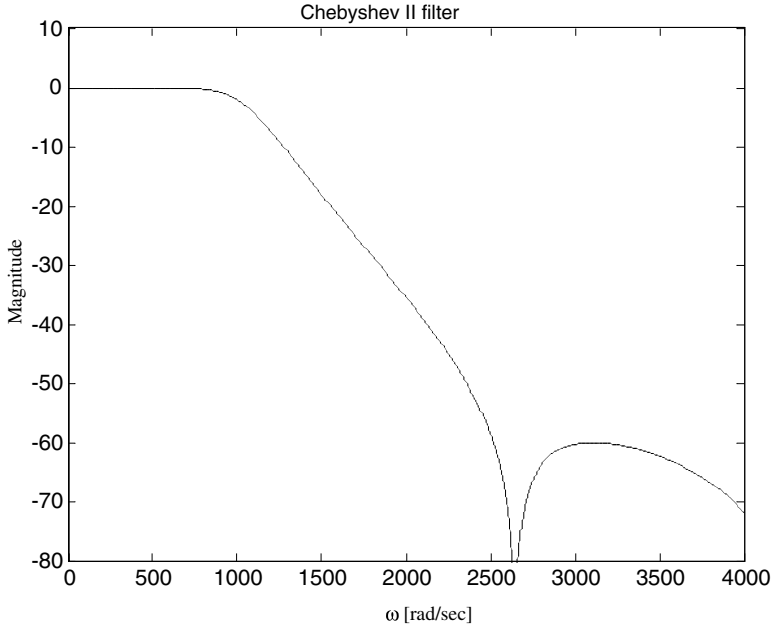

FIGURE 8.16 Magnitude response of Chebyshev Type II filter of order 5 and $w_n = 2516.1$ rad/sec.

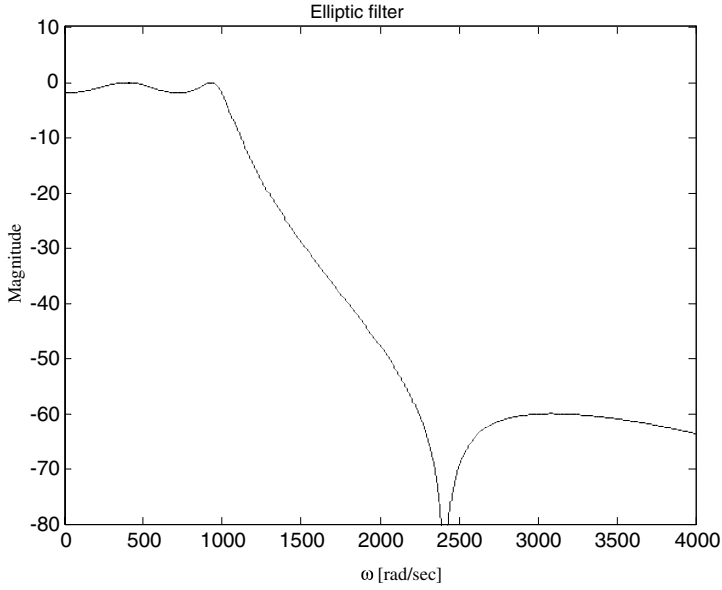

FIGURE 8.17 Magnitude response of elliptic filter of order 4 and $w_n = 1000$ rad/sec.

EOCE 8.8

Plot the pole-zero locations on the s-plane for the filters plotted in EOCE 8.7.

Solution

Use the MATLAB function zplane to plot the zeros and poles after getting them as shown below. The first script (Figure 8.18) is

```
N = 7; Wc = 1118.3;
[z , p , k] = butter( N , Wc , 's');
zplane( z , p);
```

The second script (Figure 8.19) is

```
N = 5; Wc = 1000;
[z , p , k] = cheby1( N , 2 , Wc , 's');
zplane( z , p);
```

The third script (Figure 8.20) is

```
N = 5; Wc = 2516.1;
[z , p , k] = cheby2( N , 60 , Wc , 's');
zplane(z , p);
```

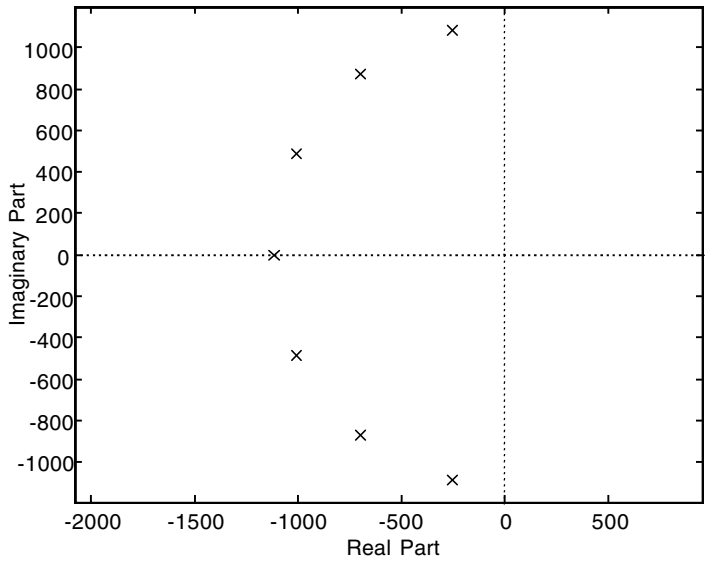

FIGURE 8.18 Pole Zero plot of Butterworth filter, $N = 7$, $w_n = 1118.3$ rad/sec.

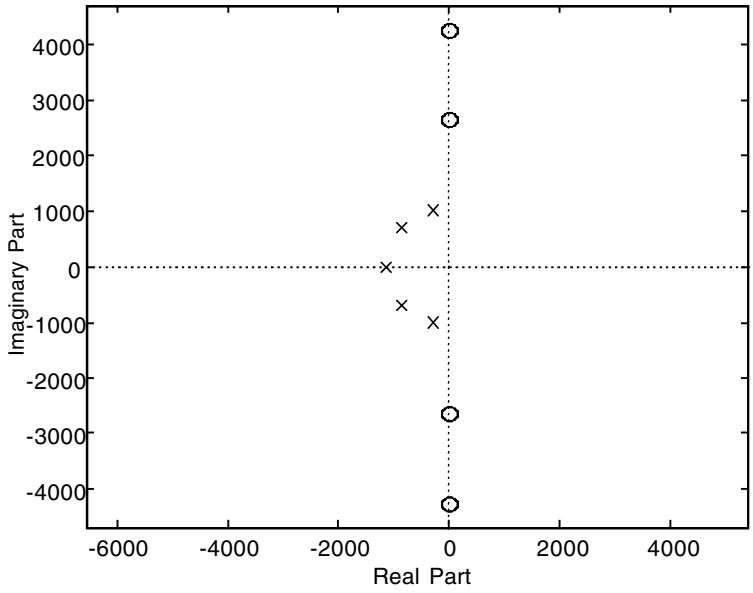

FIGURE 8.19 Pole Zero plot of Chebyshev Type I filter, $N = 5$, $w_n = 1000$ rad/sec.

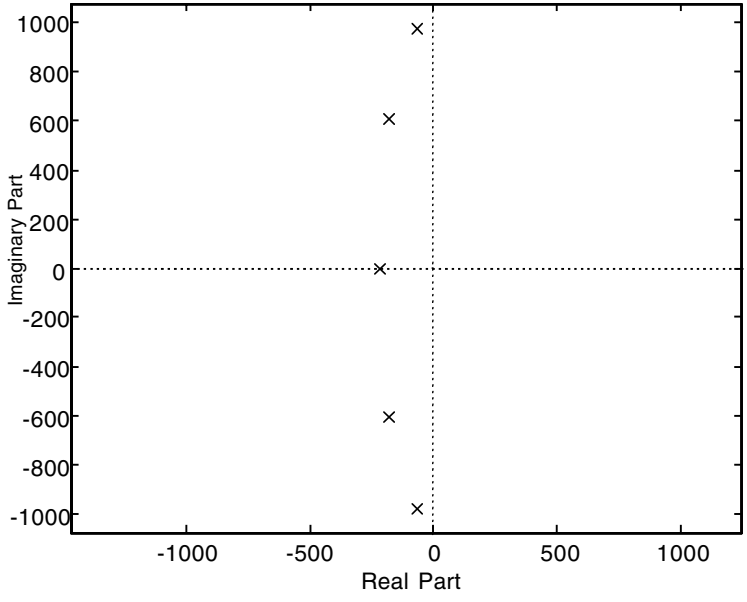

FIGURE 8.20 Pole Zero plot of Chebyshev Type II filter, $N = 5$, $w_n = 2516.1$ rad/sec.

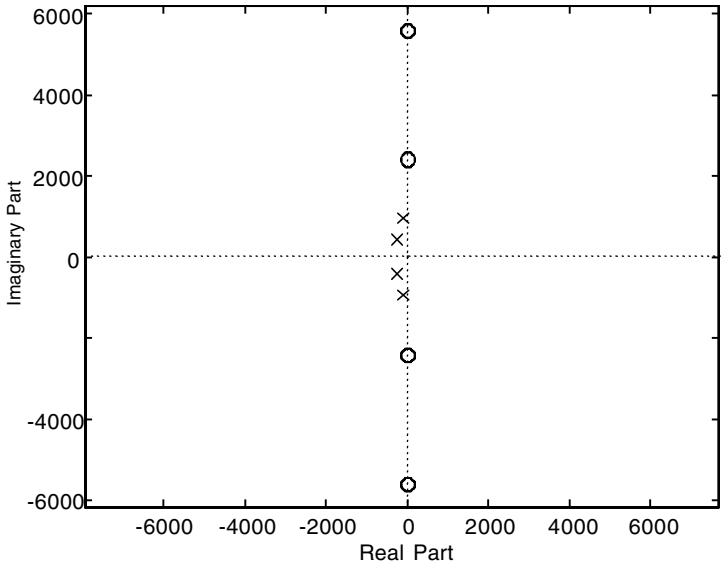

FIGURE 8.21 Pole Zero plot of elliptic filter, $N = 4$, $w_n = 1000$ rad/sec.

The fourth script (Figure 8.21) is

```
N = 4; Wc = 1000;
[z , p , k] = ellip( N , 2 , 60 , Wc , 's');
zplane( z , p );
```

Notice that poles are located on a circle whose radius is equal to w_n in the case of the Butterworth filter, whereas the poles are located on an ellipse in Chebyshev Types I and II and elliptic filters. Zeros are always located on the imaginary axis. Moreover, notice that the Butterworth and Chebyshev Type II filters don't have zeros; relate this to the fact that they are monotonic in the passband.

EOCE 8.9

Given the following signal

$$x(t) = 1 + \sin(t) + \sin(6t)$$

design an analogue filter that eliminates the component $\sin(6t)$. Plot the filter magnitude response. Assume 60-dB allowable ripple in the stopband.

Solution

We wish to design a filter with a maximally flat response in the passband and the minimum possible order. Therefore, we choose Chebyshev Type II filter. To eliminate $\sin(6t)$ and keep the other components within tolerable

attenuation, we choose w_p greater than 1 rad/sec and w_s less than 6 rad/sec. Let $w_p = 1.5$ rad/sec and $w_s = 4$ rad/sec. Now assume a passband attenuation of 3 dB is acceptable. We can estimate the order and cut-off frequency and draw the magnitude response as in the following script.

```
%Example8_9.m
Wp = 1.5; Ws = 4; Rp = 3 ; Rs = 60;
[N,Wc]=cheb2ord(Wp,Ws,
 Rp , Rs , 's');
[ b , a ] = cheby2( N , Rs , Wc , 's');
Omega = [0:0.01:7];
h = freqs( b , a , Omega);
gain = 20 * log10( abs( h ) );
plot( Omega , gain);
```

The plot is shown in Figure 8.22. Note that according to the magnitude response shown, frequencies higher than 4 rad/sec will be highly attenuated while frequencies less than 1.5 rad/sec will be attenuated slightly.

EOCE 8.10

Design a fifth-order bandpass filter having a maximally flat response in the passband and equiripple response in the stopband. Figure 8.23 shows the ideal bandpass filter. Then plot the magnitude and phase responses of the filter using Bode plots.

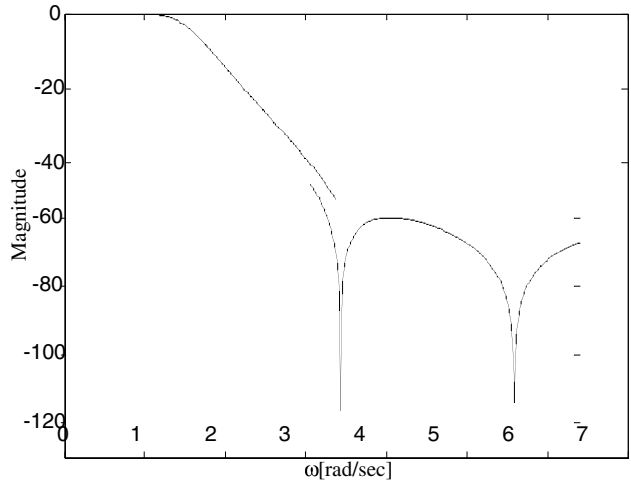

FIGURE 8.22 Magnitude response of the designed filter.

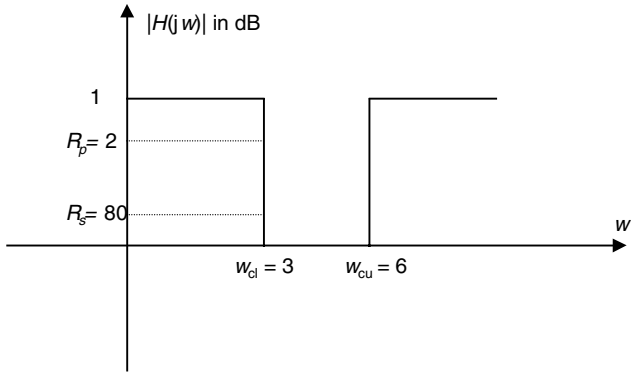

FIGURE 8.23 Specifications of the required bandpass filter shown on an ideal bandpass.

Solution

Chebyshev Type II approximation has a maximally flat response in the passband and an equiripple response in the stopband. We will use the `cheby2` MATLAB function as in the following script.

```
%Example8_10.m
N = 5; Rs = 80 ; Wc1 = 3 ; Wc2 = 6;
[B , A] = cheby2( N , Rs , [Wc1 Wc2], 's');
bode( B , A , [0:0.1:100]);
grid;
```

We used the function `bode` from the control toolbox of MATLAB to plot magnitude and phase responses. `bode` plots the magnitude in dB and phase in degrees vs. radian frequency w in log scale. The plots are shown in Figure 8.24.

EOCE 8.11

Design an analogue elliptic LP filter of the fifth order. Then use frequency transformations to obtain a highpass filter with cut-off frequency equal to 1000 rad/sec, and a bandstop filter with lower and upper cut-off frequencies equal to 1000 and 2000 rad/sec, respectively.

Solution

Using the MATLAB function `ellipap` we get the required analogue filter using the MATLAB command

```
[z, p, k] = ellipap(5,3,20);
```

where 3 is the 3-dB allowed attenuation in the passband and 20 is the allowed 20-dB attenuation in the stopband.

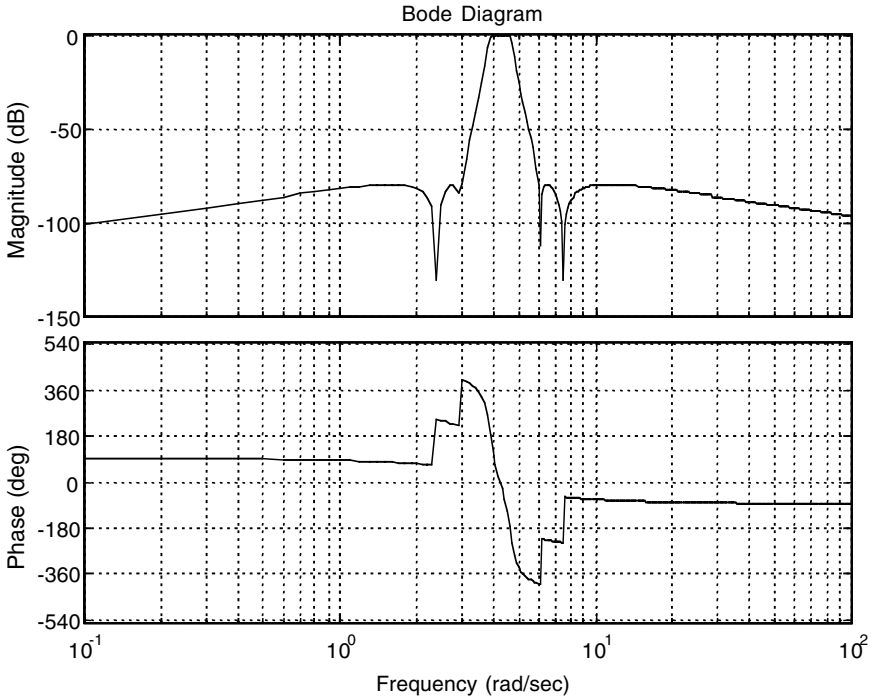

FIGURE 8.24 Bode plot showing the magnitude and phase responses of the designed bandpass filter.

Next we get the transfer function representation of the filter using MATLAB using the command

```
[B A] = zp2tf(z, p, k);
```

We then plot the response for the lowpass filter.

First let us find the highpass filter transfer function. The MATLAB function `lp2hp` is used for this purpose as in the following

```
Wn = 1000;
[Bhp Ahp] = lp2hp(B, A, Wn);
```

Finally we find the bandstop transfer function. The MATLAB function `lp2bs` is used for this purpose. The following script contains all these commands and plots the lowpass and the bandstop filter responses as shown in Figure 8.25.

```
%Example 8.11
% Using the MATLAB function ellipap we get the required
 analogue filter:
[z, p, k] = ellipap(5,3,20);
```

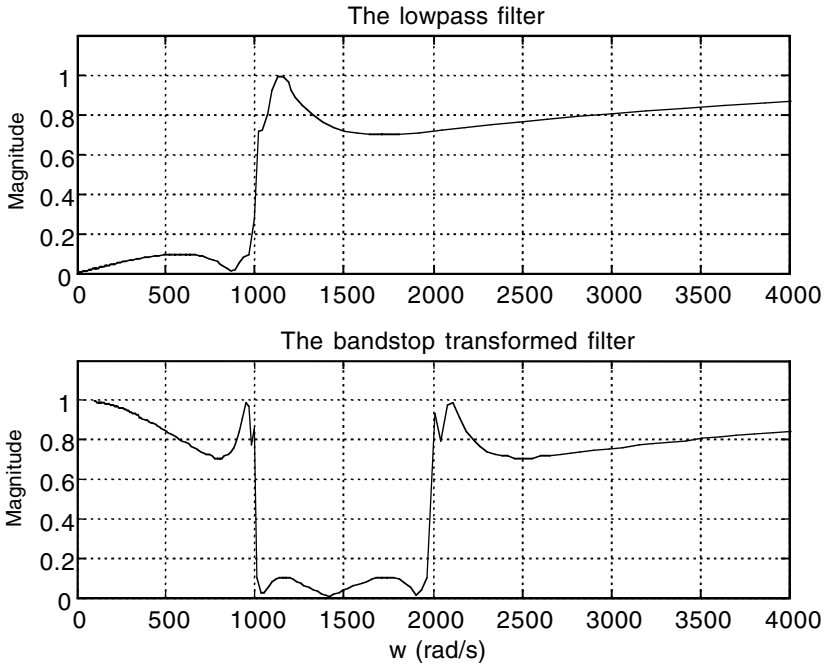

FIGURE 8.25 Magnitude response of the bandstop filter.

```
% Now we get the transfer function representation of the
% filter as follows:
[B A] = zp2tf(z, p, k);
% First lets find the highpass filter transfer function.
% The MATLAB function lp2hp is used for this purpose:
Wn = 1000;
[Bhp Ahp] = lp2hp(B, A, Wn);
% Now we find the bandstop transfer function.
% The MATLAB function lp2bs is used for this purpose:
Wnl = 1000;
Wnu = 2000;
W0 = sqrt(Wnl*Wnu);
Bw = Wnu - Wnl
[Bbs Abs] = lp2bs(B, A, W0, Bw);
% Here we generate the magnitude response plots:
%For highpass filter
[Hhp,Whp] = freqs(Bhp, Ahp);
subplot(2,1,1);
plot(Whp, abs(Hhp));
axis([0 4000 0 1.2]);
```

```
title('The lowpass filter');
grid;
ylabel('Magnitude');
%For bandstop filter
[Hbs,Wbs] = freqs(Bbs, Abs);
subplot(2,1,2);plot(Wbs, abs(Hbs));
axis([0 4000 0 1.2]);
grid;
xlabel('w (rad/s)');
ylabel('Magnitude');
title('The bandstop transformed filter');
```

Notice that the frequency transformation preserved the filter specifications.

EOCE 8.12

Consider the circuit in Figure 8.26. Find values for R and L so that the output $y(t)$ will contain frequencies below 20π rad/sec only.

Solution

We will start by investigating what kind of filter is given. We will put the circuit in the Laplace domain and find the transfer function relating the input $x(t)$ to the output $y(t)$. The transfer function is obtained by the voltage divider method as

$$Y(jw) = \frac{X(jw)(R)}{R + jwL}$$

or

$$H(jw) = \frac{R}{R + jwL} = \frac{(R/L)}{R/L + jw}$$

We can see that this circuit is a lowpass filter with a maximum gain of unity and a cut-off frequency of R/L (prove that). It is desired that we pass frequencies below 20π rad/sec, which means that the desired cut-off frequency is 20π rad/sec. Let L = 1000 mH, then $R = 20\pi\ \Omega$.

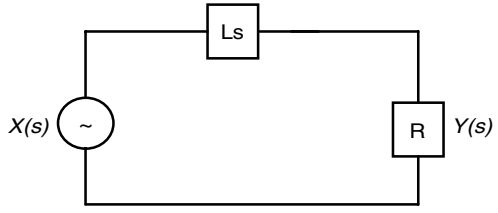

FIGURE 8.26 Circuit for EOCE 8.12.

EOCE 8.13

The maximum gain that can be achieved with almost all passive circuits is unity. Suppose we reconsider EOCE 8.12 and, in addition, we need to attenuate the input signal by 50%. Can we modify the existing circuit as described in EOCE 8.12 to accomplish this?

Solution

Let us add another resistor in parallel with R across the output $y(t)$. The transfer function in this case is

$$Y(jw) = \frac{X(jw)(R_{eq})}{R_{eq} + jwL}$$

$$H(jw) = \frac{R_{eq}}{R_{eq} + jwL} = \frac{\left(R_{eq}/L\right)}{R_{eq}/L + jw}$$

where

$$R_{eq} = \frac{RR_L}{R + R_L}$$

and R_L is the resistor added in parallel with R. It is clear that adding a load resistor will not change the maximum magnitude of the transfer function; it is still unity and the attempt fails.

Let us instead add another resistor in series with the input source, $x(t)$. The transfer function in this case is

$$H(jw) = \frac{R/L}{jw + R_s/L + R/L}$$

with cut-off frequency at $\frac{R_s + R}{L}$ and maximum gain of $\frac{R}{R_s + R}$. We desire a magnitude of 0.5. This can be accomplished by setting $R = R_s$. The desired cut-off frequency is 20π rad/sec. So we write

$$\frac{R_s + R}{L} = 20\pi = \frac{2R}{L}$$

With $R = 20\pi$, we select a new L of 2 H.

EOCE 8.14

Consider the series RLC circuit in Figure 8.27, with $y(t)$ as the output and $x(t)$ as the input. Select values for R, L and C so that the output will receive only the range of frequencies between 10 and 20 Hz.

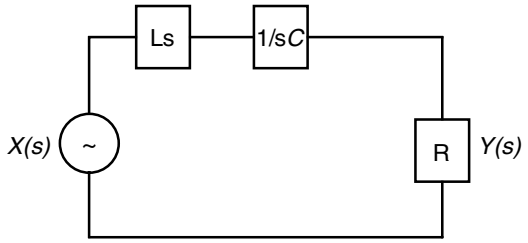

FIGURE 8.27 Circuit for EOCE 8.14.

Solution

At low frequencies the output voltage is zero. At high frequencies, the output voltage is zero as well. This is a bandpass filter. We start the design by calculating the center frequency as $w_0 = 2\pi f_0$ and $f_0 = \sqrt{f_{cl} f_{cu}} = 14.14$ Hz. $(w_0)^2 = (2\pi f_0)^2 = 8186.2$. Let $C = 1 \times 10^{-6}$ F, then $L = 1/Cw_0^2 = 122.156$ Hz and the quality factor is $Q = \dfrac{f_0}{f_{cu} - f_{cl}} = 1.414$. For R we use the equation

$$R = \sqrt{L/\left(CQ^2\right)} = 7816.4\ \Omega.$$

Having values for R, L and C, the design is complete. Notice that the center frequency is independent on R. Let us vary R and give it values of 10, 100, 1000 and the designed value of 7816.4 Ω. Let us fix L and C as in the design. We will write the next MATLAB script to demonstrate the effect of varying R and observing the bandwidth and the magnitude of each transfer function that corresponds to each R.

```
clf
R=[10 100 1000 7816.4]
L=122.156;
C=0.000001;
w=0:1:200;
for i=1:4
T1=((j*w)*(R(i)/L))./(-w.^2+j*w*(R(i)/L)+1/(L*C));
Mag_T1=abs(T1);
plot(w,Mag_T1);
hold on
end
title('Second order RLC series bandpass filter')
xlabel('w in rad');
ylabel('Magnitude of the transfer functions');
```

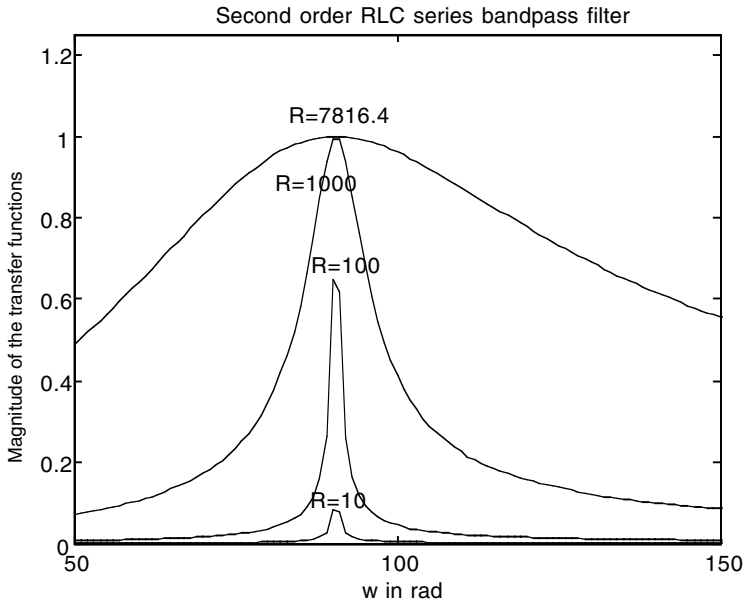

FIGURE 8.28 Plots for EOCE 8.14.

```
axis([50 150 0 1.25])
gtext('R=10');gtext('R=100');
gtext('R=1000');gtext('R=7816.4');
```

The plots are shown in Figure 8.28. We can see that the center frequency is the same for each R used, but for $R = 10\ \Omega$, the maximum magnitude is very small. This magnitude increases as the value of R increases.

EOCE 8.15

Design a bandstop filter that has a center frequency of 50 Hz and a bandwidth of 100 Hz. Use passive circuit elements. The circuit is shown in Figure 8.29.

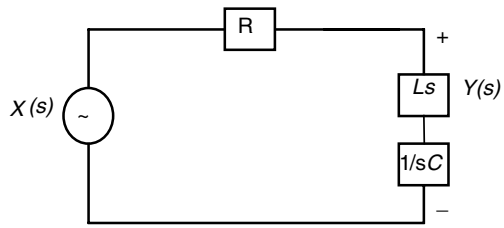

FIGURE 8.29 Circuit for EOCE 8.15.

Solution

The transfer function relating the input to the output is

$$H(s) = \frac{s^2 + \dfrac{1}{LC}}{s^2 + (R/L)s + 1/LC}$$

The quality factor is

$$Q = \frac{f_0}{f_{cu} - f_{cl}} = 0.5$$

and the inductance is

$$L = \frac{1}{Cw_c^2}$$

With $C = 1 \times 10^{-6}$ F, L is

$$L = \frac{1}{Cw_0^2} = \frac{1}{(2\pi(50))^2(10^{-6})} = 10.13 \text{ H}$$

For R we use the equation

$$R = \beta L = 2\pi\,(100)(10.13) = 6364.9\ \Omega.$$

Having values for R, L and C, the design is complete.

The following MATLAB script generates multiple plots for the design and also varies the value of R which has no effect on the center frequency but controls the bandwidth of the rejection band.

```
clf
R=[10 100 1000 6364.9]
L=10.1321;
C=0.000001;;
w=0:1:200*pi;
for i=1:4
T1=(-w.^2+1/(L*C))./(-w.^2+j*w*(R(i)/L)+1/(L*C));
Mag_T1=abs(T1);
plot(w,Mag_T1);
hold on
```

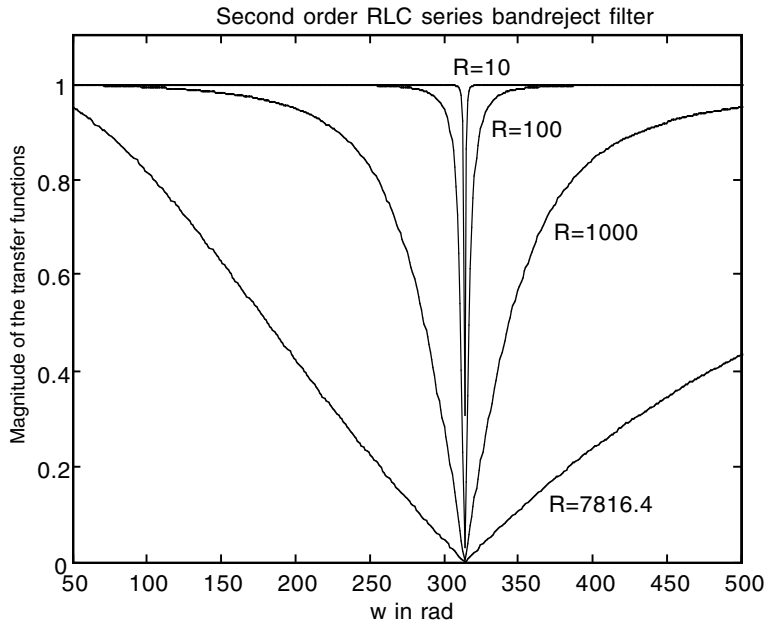

FIGURE 8.30 Plots for EOCE 8.15.

```
end
title('Second order RLC series bandreject filter')
xlabel('w in rad');
ylabel('Magnitude of the transfer functions');
axis([50 500 0 1.1])
gtext('R=10');gtext('R=100');
gtext('R=1000');gtext('R=7816.4');
```

The plots are in Figure 8.30. Pay close attention to the way the resistance R is shaping the magnitude of the transfer function. The smaller the R, the narrower the rejection, or the more selective the filter becomes in terms of rejecting a specific band of frequencies.

EOCE 8.16

Consider the two systems

$$H_1(s) = \frac{1}{s+10} \quad \text{and} \quad H_2(s) = \frac{s}{s+1}$$

What range of frequencies in the input $x(t)$ that the combination of the two systems, if placed in series, will allow to pass? What is the resulting filter?

Solution

We will have $H_1(s)H_2(s)$ as the new system. Note that the first system is lowpass and the second is highpass. When we cascade the two systems together, we expect the overall system to be bandpass with the passing range as [1 10] rad/sec. We will use MATLAB again and write the following script to see what type of filter this combination is.

```
Clf
num1=[1];
den1=[1 10];
num2=[1 0];
den2=[1 1];
num=conv(num1,num2); % multiply num1 with num2
den=conv(den1,den2);
bode(num,den);
title('The filter representing H1(s)H2(s)')
```

The plots are in Figure 8.31. This is a bandpass filter as it emphasizes frequencies in the mid range. Why don't you try the other combinations?

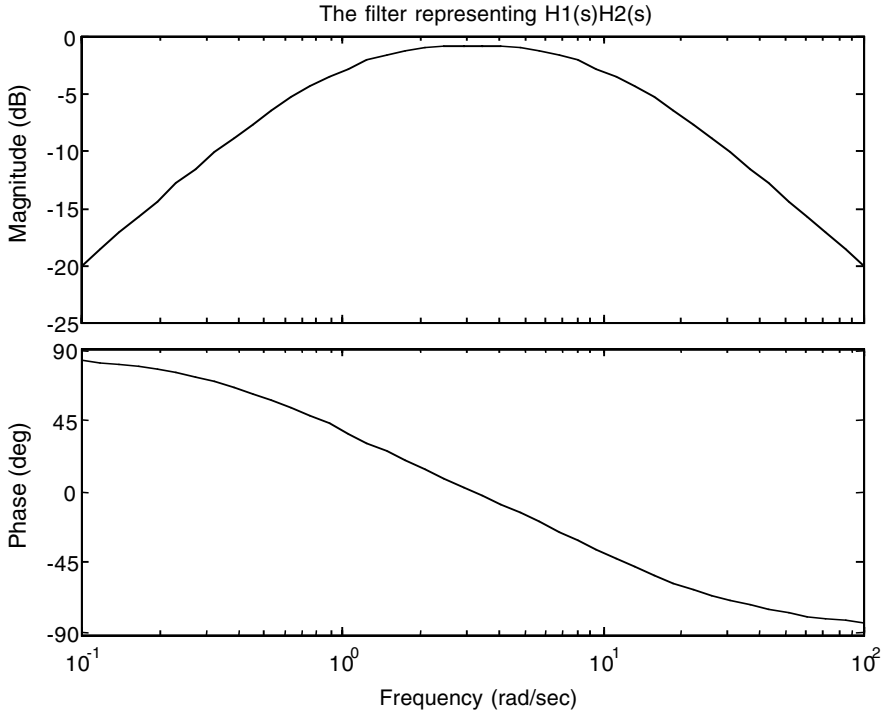

FIGURE 8.31 Plots for EOCE 8.16.

8.14 End of Chapter Problems

EOCP 8.1

Draw the poles of

$$H(s) = \frac{1}{1 + \left(s/jw_s\right)^{2N}}$$

for a third-order lowpass Butterworth filter after replacing s by jw.

EOCP 8.2

Consider a series RLC circuit where the output voltage $y(t)$ is taken across the resistor and the input is the voltage $x(t)$.

1. Find the transfer function of this filter.
2. What is this type of filter?

EOCP 8.3

An elliptic bandstop filter is to be designed. It should fulfill the following specifications:

$1 >= |H(\mathrm{j}w)|^2 >= 0.95 \quad w <= 1200$ and $w >= 1800$

$|H(\mathrm{j}w)|^2 <= 0.02 \quad 800 >= w >= 2200$

1. Estimate the order and the cut-off frequencies.
2. Find the transfer function of the filter.

EOCP 8.4

Plot the magnitude and phase responses of Butterworth, Chebyshev Type I, Chebyshev Type II, elliptic, and Bessel LP filters. Let the order be 5 and the cut-off frequency be 1. Assume R_p = 3 dB and R_s = 60 dB. Comment on the phase response of Bessel filter compared with others.

EOCP 8.5

Design a bandstop filter that has a bandwidth β of 1000 rad/sec that can reject the component sin(1414t) from the following signal

$$x(t) = \sin(500t) + \sin(1414t) + \cos(2500t)$$

EOCP 8.6

Consider the following transfer function

$$H(s) = \frac{as^2 + bs + c}{s^2 + ds + e}$$

Design by finding the values of a, b, c, d and e the filters in the table below. (Hint: The transfer function above is for a Butterworth filter.)

Type	w_n	a	b	c	d	e
LP	1000					
HP	1000					
BP	1000, 2000					
BS	1000, 2000					

EOCP 8.7

A lowpass filter has a desired peak passband ripple of 0.2 dB, and a minimum stopband attenuation of 40 dB; what are the values of ε and δ?

EOCP 8.8

Find and graph the frequency response of the transfer functions given by

$$H(s) = \frac{0.2s^2 + 1}{s^2 + 0.1s + 1}$$

$$H(s) = \frac{s^2}{s^2 + 0.1s + 1}$$

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

$$H(s) = \frac{s}{s^2 + 0.1s + 1}$$

What type of filters are these systems?

EOCP 8.9

Use MATLAB to estimate the orders and cut-off frequencies for all filter types having the following specifications:

$w_p = 2000$ rad/sec $\quad w_s = 3000$ rad/s
$R_p = .1$ dB $\quad R_s = 50$ dB

EOCP 8.10

Study the effect of the filter order N on the characteristics of the elliptic filter. Show by drawing graphs, each with different order N.

EOCP 8.11

Design an analogue elliptic LP filter of order N equal to 6 and cut-off frequency equal to 100. Use MATLAB to design the filter and to generate the frequency response graphs.

EOCP 8.12

Given the following signal

$$x(t) = 1 + \sin(t) + \sin(6t)$$

design an analogue filter that eliminates the component $\sin(t)$. Plot the filter magnitude response. Assume 60-dB allowable ripple in the stopband.

EOCP 8.13

Design a sixth-order bandstop filter having a maximally flat response in the passband and equiripple response in the stopband. Let $R_s = 50$ dB.

EOCP 8.14

Consider the system in Figure 8.32.

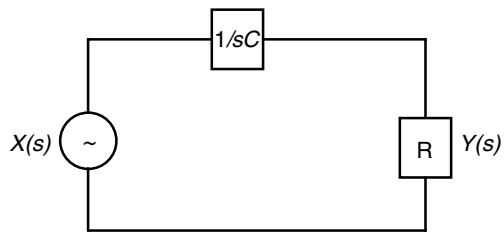

FIGURE 8.32 Circuit for EOCP 8.14.

1. Taking $y(t)$ as the output voltage across the resistor, what is the transfer function relating $Y(s)$ to $X(s)$?
2. Taking $y(t)$ as the voltage across the capacitor, what is the transfer function relating $Y(s)$ to $X(s)$?
3. What are the eigenvalues in parts 1 and 2?
4. What types of filters are represented in parts 1 and 2?
5. What are the effects of the zeros in parts 1 and 2 on the nature of the filter?
6. What is the cut-off frequency in parts 1 and 2?
7. If R = 1k and C = 1μF, what is an approximation to the output $y(t)$ in the filter represented in part 1 if the input $x(t) = 10\sin(10t) - \sin(1000t) + 12$?
8. If R = 1k and C = 1μF, what is an approximation to the output $y(t)$ in the filter represented in part 2 if the input $x(t) = 10\sin(10t) - \sin(1000t) + 12$?

EOCP 8.15

Consider the circuit in Figure 8.33.

1. Taking $y(t)$ as the voltage across the capacitor, what is the transfer function relating $Y(s)$ to $X(s)$.
2. What type of filter is represented in Figure 8.33?
3. What is the cut-off frequency if it exists?

EOCP 8.16

Consider the system in Figure 8.34.

1. Taking $y(t)$ as seen in Figure 8.34, what is the transfer function relating $Y(s)$ to $X(s)$?
2. What type of filter is represented in Figure 8.34?
3. What is the cut-off frequency if it exists?

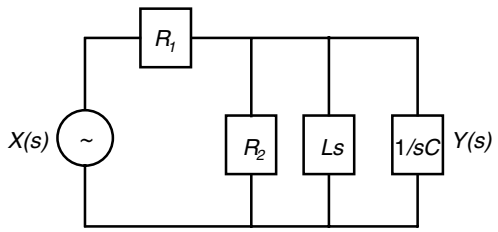

FIGURE 8.33 Circuit for EOCP 8.15.

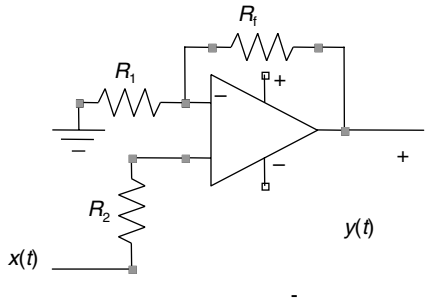

FIGURE 8.34 Circuit for EOCP 8.16.

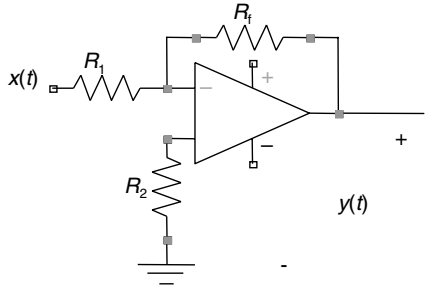

FIGURE 8.35 Circuit for EOCP 8.16.

EOCP 8.17

Consider the system in Figure 8.35.

1. Taking $y(t)$ as seen in Figure 8.35, what is the transfer function relating $Y(s)$ to $X(s)$?
2. What type of filter is represented in Figure 8.35?
3. What is the cut-off frequency if it exists?

EOCP 8.18

Consider the system in Figure 8.36.

1. Taking $y(t)$ as seen in Figure 8.36, what is the transfer function relating $Y(s)$ to $X(s)$?
2. What type of filter is represented in Figure 8.36?
3. What is the cut-off frequency if it exists?
4. What is the output $y(t)$ if $x(t) = 10\sin(10t) - \sin(1000t) + 12$?

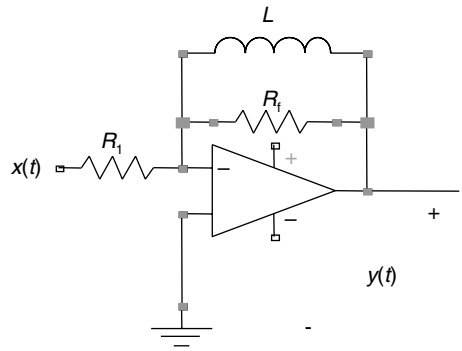

FIGURE 8.36 Circuit for EOCP 8.17.

EOCP 8.19

Consider the following systems represented by the transfer functions relating the output to the input of some dynamic systems.

$$\text{(a)}\ \frac{1}{s^2+5s+6}$$

$$\text{(b)}\ \frac{s}{s^2+4s+4}$$

$$\text{(c)}\ \frac{s^2}{s^2+7s+10}$$

$$\text{(d)}\ \frac{s^3+1}{s^2+7s+10}$$

$$\text{(e)}\ \frac{s^3+s^2+1}{s^2+7s+10}$$

1. Plot the magnitude of each system vs. frequency. You may use MATLAB to obtain the bode plots.
2. Remember that the cut-off frequency is found by locating the maximum magnitude on the bode plot and going 3 db down the magnitude scale. Find the cut-off frequencies for the given systems?
3. Characterize the filters given in this problem.

EOCP 8.20

Consider the following systems

$$H_1(s)\ \frac{1}{7s+a}$$

$$H_2(s)\ \frac{1}{s^2+as+b}$$

1. For the first system, design to have a maximum DC gain of 10 and a cut-off frequency of 10 k. Find the a value and the additional gain desired.
2. For the second system, a DC gain of 5 and a cut-off frequency of 10 k is desired. We want the design to resemble a lowpass filter with positive a and b values. Find a and b.

9

Transformations between Continuous and Discrete Representations

9.1 The Need for Converting a Continuous Signal to a Discrete Signal

We convert a continuous signal to a discrete signal so that we can use the computer to analyze and perform many complex computations that are otherwise very difficult to be carried out analytically. The area of digital signal processing is based on the discretized analogue signal.

When a speech signal is discretized, it becomes very easy to store the spoken words electronically and to transmit these words digitally. This is done using a *digital processor*. Some digital processors are used to enhance recorded speech. Others are used to generate waveforms that are similar to our spoken words.

In a radar station, a continuous signal is sent into space. When this signal hits a target, it reflects back and is picked up by the radar antenna to be analyzed. Discretizing the reflected signal at this point becomes very important and the computer can perform all kinds of analyses and make very important calculations regarding the distance, speed and direction of the target as well as how high it is. Without the discrete signal, this crucial information is very difficult to obtain.

In medical imaging, a very complex mathematical operation is needed in the CAT scan operation. This mathematical operation is very difficult to implement using analogue circuits; however, it is easily carried on using a digital processor of a computer.

In real life situations, most signals that we need to process are analogue signals. These signals must be discretized so that we can study them and perform some computations using the computer. We may need to reconstruct the discrete signal that the computer outputs and generate an analogue signal to drive a radar antenna, for example. To use a computer in studying signals we need to discretize the input signal so that the computer can work on it to produce the discrete output. This output is then converted to analogue signal again. Thus, there is a process needed to be performed before

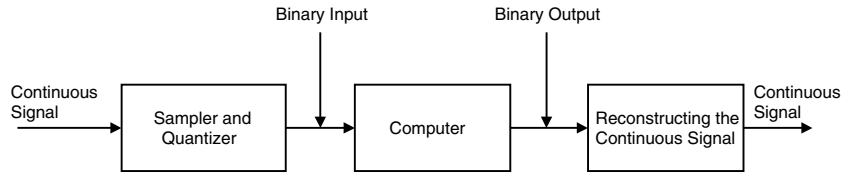

FIGURE 9.1 Mixed system.

and after the signal is acted upon by the digital processor. However, the advantages of using a digital processor are many.

If you use a calculator to multiply 5 by 5 you will get a precise result, 25. A digital signal processing is not affected by temperature or any other disturbances like the analogue processor. Very complex tasks such as looping and conditional statements are difficult to build using analogue circuits. Yet these are readily built using digital processors.

In control systems, we often need to have an observer to estimate some of the parameters of the controlled system. We can build this observer using analogue circuits. But if we need to do some modification to the observer, the complete design has to be revised. However, if we use a digital observer, we need only to program it to make the changes. This is an issue of flexibility. An observer made from analogue circuits may be heavy and large; a digital processor is small and lightweight.

Infinite impulse response digital filters are very difficult to design in the z-domain. We transform the continuous analogue filter with feedback (designing analogue filters in the s-domain is straightforward) to a digital IIR filter in a very easy way.

The process of discretizing a continuous signal and then converting the output of the digital processor to a continuous signal is shown in Figure 9.1. The continuous signal is digitized, that is, it is discretized and then each sample is quantified to a finite binary number. The computer will process this binary input and produce the binary output. This binary output is then converted to a continuous signal. Sampling and quantization is often known as analogue-to-digital conversion (A/D) and reconstruction is often known as digital-to-analogue (D/A).

9.2 From the Continuous Signal to Its Binary Code Representation

Before it gets to the computer input, the continuous signal is first sampled at regular intervals. An example is shown in Figure 9.2. After sampling, we will have only values of the continuous signal $x(t)$ at nT_s where T_s is the sampling interval and n is an integer. Every time the sampler picks a sample, the sample is being held to be quantified and given a binary number representation. Then

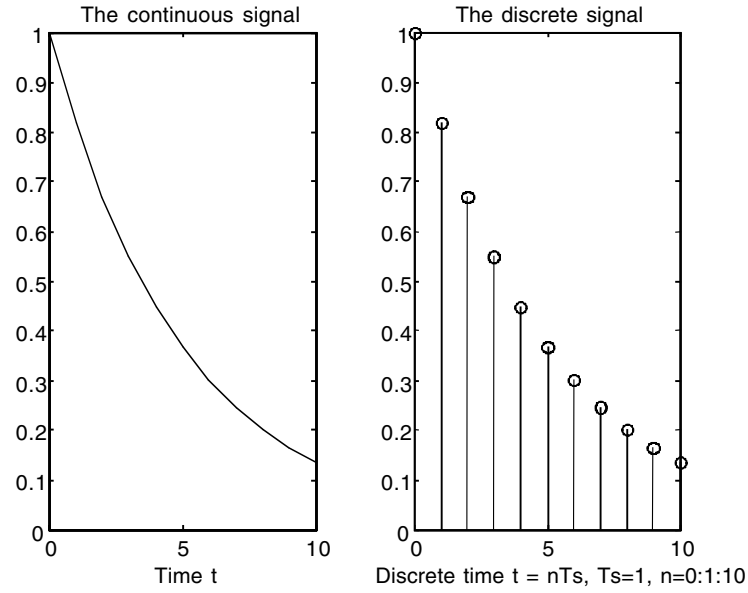

FIGURE 9.2 Continuous and discrete signals.

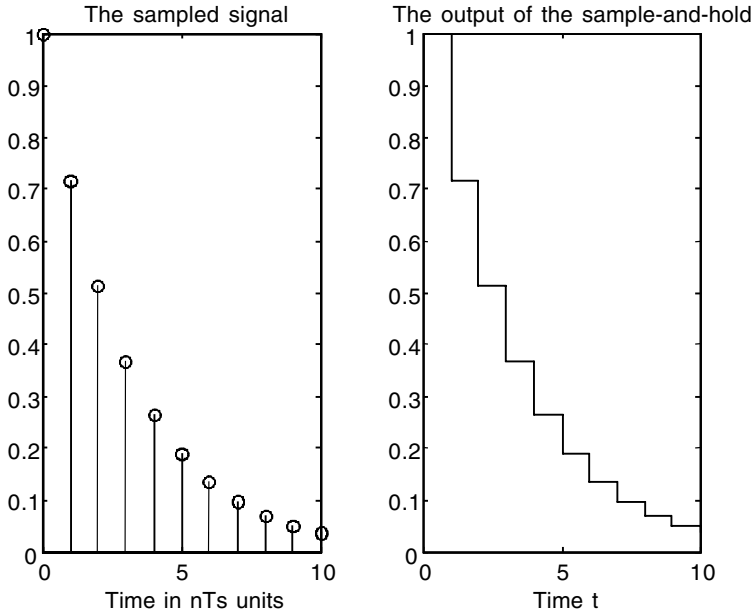

FIGURE 9.3 The sample-and-hold process.

the sampler picks another sample and the process of holding and binary representation continues. The holding is needed so that the value of the sample taken is given a binary code. This process is shown in Figure 9.3.

The sample-and-hold is a device that will sample $x(t)$ and hold the value of the signal $x(n)$ so that a binary code is attached to it. Then another $x(n)$ value in taken from $x(t)$. This value will be held and given a binary code. The process continues in this fashion.

9.3 From the Binary Code to the Continuous Signal

The digital-to-analogue device at the output of the computer will pick the value of the first binary code and hold it until the next value is made ready and the process continues. The output will be similar to the output of the sample-and-hold discussed earlier. The final analogue or continuous signal is obtained by smoothing this staircase-like signal by passing it through a lowpass filter.

9.4 The Sampling Operation

We will look at the sampling operation in real-time and frequency domains.

9.4.1 Ambiguity in Real-Time Domain

Consider the following real-time signal

$$x(t) = \sin(wt) = \sin(2\pi ft) \tag{9.1}$$

Let us sample $x(t)$ at the rate of f_s samples per second. Let us also assume that the sampling is at regular intervals of T_s duration where $T_s = 1/f$ sec. By starting the sampling at $t = 0$, the next sample will be at T_s, the one after is at $2T_s$, and so on. Then we can say that we are sampling at $T = nT_s$ with $n = 0, 1, 2,\ldots$

The sampled signal is then

$$x(nT_s) = \sin(2\pi fnT_s) = \sin\left(2\pi \frac{f}{f_s} n\right) \tag{9.2}$$

where $x(nT_s)$ is the next sample of $x(t)$. Let us write $x(nT_s)$ as $x(n)$ where we understand that we are sampling at nT_s.

We know that the sinusoidal signal repeats every $2\pi m$ radians. Thus we can write

$$x(n) = \sin\left(2\pi \frac{f}{f_s} n\right) = \sin\left(2\pi \frac{f}{f_s} n + 2\pi m\right) \tag{9.3}$$

or

$$x(n) = \sin\left(2\pi f n T_s\right) = \sin\left(2\pi f n T_s + 2\pi m\right) = \sin\left(2\pi\left(f + \frac{m}{nT_s}\right) n T_s\right) \tag{9.4}$$

If we let $m/n = k$, then we have

$$x(n) = \sin\left(2\pi f n T_s\right) = \sin\left(2\pi\left(f + \frac{k}{T_s}\right) n T_s\right) \tag{9.5}$$

If we equate the arguments in Equation (9.5) we will get

$$f = f + \frac{k}{T_s} = f + k f_s \tag{9.6}$$

The significance of the last equality is that if we have a periodic signal with frequency f that is sampled at f_s, and another periodic signal with frequencies $f + kf_s$ that are sampled also at f_s, the samples of all these signals will be the same. This implies that we can have the same $x(n)$ discrete signals that could be the result of sampling, for example, the following signals:

$x_1(t) = \sin(2\pi(1)t)$ with $f = 1$Hz
$x_2(t) = \sin(2\pi(3)t)$ with $f = 3$Hz
$x_3(t) = \sin(2\pi(6)t)$ with $f = 6$Hz

at the sampling frequency $f_s = 2$ samples per second. We will take three samples from each signal.

For the first signal

$$x_1(n) = \sin\left(2\pi n T_s\right) = \sin\left(2\pi n/2\right) = \sin(n\pi)$$

$$x_1(0) = \sin(0) = 0,\ x_1(1) = \sin(\pi) = 0,\ x_1(2) = \sin(2\pi) = 0$$

The second signal is

$$x_2(n) = \sin\left(2\pi(3) n T_s\right) = \sin\left(\frac{6\pi n}{2}\right) = \sin(3\pi n)$$

$$x_2(0) = \sin(0) = 0,\ x_2(1) = \sin(3\pi) = 0,\ x_2(2) = \sin(6\pi) = 0$$

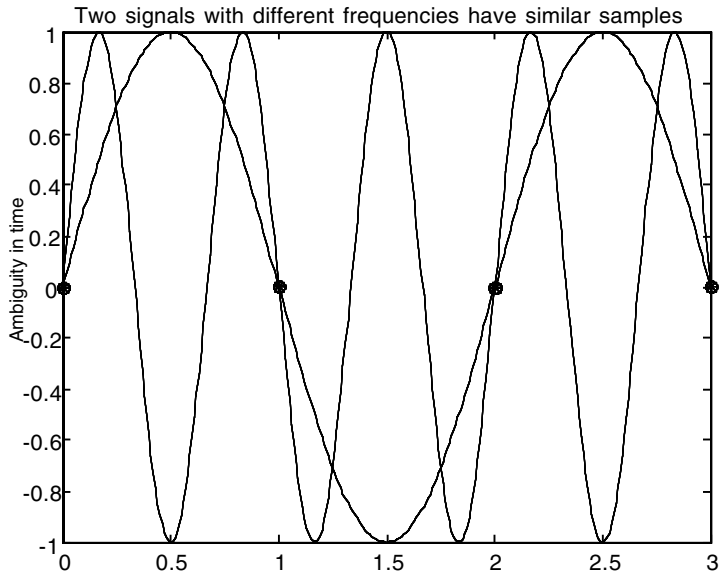

FIGURE 9.4 Ambiguity in time.

For the third signal

$$x_3(n) = \sin\left(2\pi(6)n\frac{1}{2}\right) = \sin(6\pi n)$$

$$x_3(0) = \sin(0) = 0,\ x_3(1) = \sin(6\pi) = 0,\ x_3(2) = \sin(12\pi) = 0$$

You can see that in the 3-samples sequence (and it could be the 4 samples, the 50 samples, the 1000 samples) $x(n)$ is the same signal that we get if we sample the three signals at the same frequency $f_s = 2$ Hz. It is known that, given the signal $x(n)$, it would be impossible to decide which time signal $x(n)$ represents without any extra information. This is why ambiguity arises here. The plot to illustrate this ambiguity is shown in Figure 9.4.

To conclude, we can say that if we sample a sinusoidal wave of frequency f at a sampling frequency f_s and then sample any other sinusoidal signal of frequency $(f + kf_s)$, at the same frequency f_s with k as a positive or negative integer, we will not be able to distinguish among the sampled values of these signals. The name aliasing arises from these observations. One $x(n)$ discrete signal can contain the sample values at the sampling frequency f_s of $x(2\pi t)$ and $x(2\pi(3)t)$. Thus the 1 Hz-signal is an alias to the 3-Hz signal.

9.4.2 Ambiguity in the Frequency Domain

Consider the band-limited signal $x(t)$ with its spectrum shown in Figure 9.5. The signal is said to be band-limited since there are no frequency components

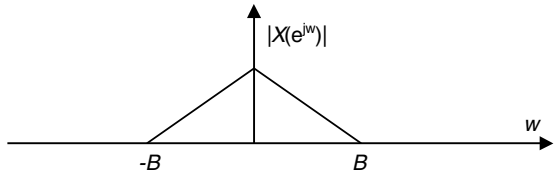

FIGURE 9.5 The band-limited signal $x(t)$.

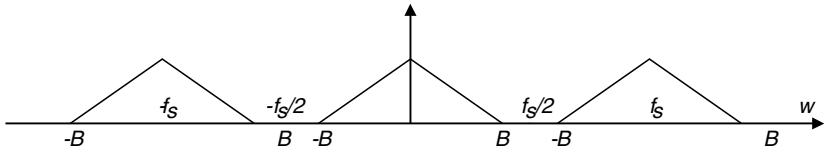

FIGURE 9.6 Sampling without aliasing.

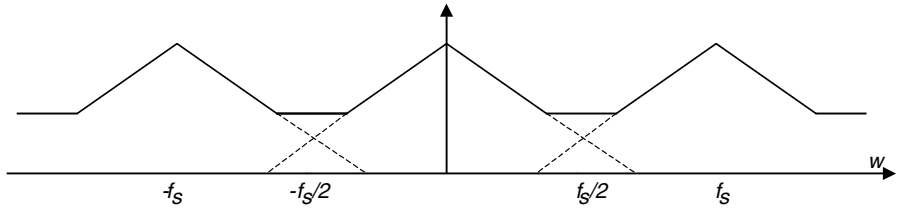

FIGURE 9.7 Sampling with aliasing.

below or above B Hz. The discrete spectrum, $|X(e^{j\theta})|$, is periodic. If we sample $x(t)$ at $f_s > 2B$, we will have the case as shown in Figure 9.6. From this figure you can see that the original signal can be filtered out and uniquely determined as the output of a lowpass filter since there is no overlapping among the repeated spectrum. However, if we sample at $f_s < 2B$, we will have the case as shown in Figure 9.7. You can also see from Figure 9.7 that overlapping and aliasing occur as a result at the edges of the repeated spectra. In this case, we cannot filter our original signal because it is distorted.

9.4.3 The Sampling Theorem

In Sections 9.4.1 and 9.4.2 we observed that if we are given a continuous-time signal $x(t)$ that is to be sampled at the sampling frequency f_s to produce the discrete signal $x(n)$, the original signal $x(t)$ will be recovered completely from its sampled signal $x(n)$ if $x(n)$ is the result of sampling $x(t)$ at the sampling frequency f_s that is greater than two times the highest frequency in $x(t)$. The condition $f_s > 2f_m$ where f_m is the highest frequency in $x(t)$ is known as the sampling condition or the sampling theorem.

9.4.4 Filtering before Sampling

If we are trying to sample the continuous signal $x(t)$ that is of interest to us at the sampling frequency f_s, the whole spectral content of the signal $x(t)$ will reside in the interval $[-f_s/2 \quad f_s/2]$. If there is a noise signal $n(t)$ that has frequency components outside the range of the signal of interest $x(t)$, these frequency components will appear in the spectral interval $[-f_s/2 \quad f_s/2]$. To avoid this we will filter out the noise signal $n(t)$ before we sample the signal

$$s(t) = x(t) + n(t)$$

by passing $s(t)$ through a lowpass filter that has a cut-off frequency of $f_c = B$ Hz where B is the positive maximum frequency in the signal $x(t)$. In practice we always have some noise associated with the signal of interest $x(t)$ and we always filter the input signal.

Example 9.1

Consider the continuous signals

$$x_1(t) = \sin(2\pi(5)t)$$

$$x_2(t) = \sin(2\pi(15)t)$$

1. Sample $x_1(t)$ at $f_s = 10$ Hz and find $x_1(n)$.
2. Sample $x_2(t)$ at $f_s = 10$ Hz and find $x_2(n)$.
3. Compare $x_1(n)$ and $x_2(n)$ and comment on the result.

Solution

1. If we sample $x_1(t)$ at $t = nT_s = n(.1)$, we get $x_1(n) = \sin(2\pi(5)n(.1))$ or $x_1(n) = \sin(n\pi)$.
2. If we sample $x_2(t)$ at $t = nT_s = n(.1)$, we get $x_2(n) = \sin(2\pi(15)(n)(.1))$ or $x_2(n) = sin(3n\pi)$.
3. $x_1(n) = \sin(n\pi)$ and $x_2(n) = \sin(3n\pi) = \sin(3n\pi - 2n\pi) = \sin(n\pi)$.

This indicates that $x_1(n)$ and $x_2(n)$ are the same. Remember that $x_1(t)$ has the 5-Hz frequency and $x_2(t)$ has the 15-Hz frequency. This is the ambiguity that we talked about in time-domain.

Example 9.2

Consider the signals we had in Example 9.1, namely,

$$x_1(t) = \sin(2\pi(5)t)$$

$$x_2(t) = \sin(2\pi(15)t)$$

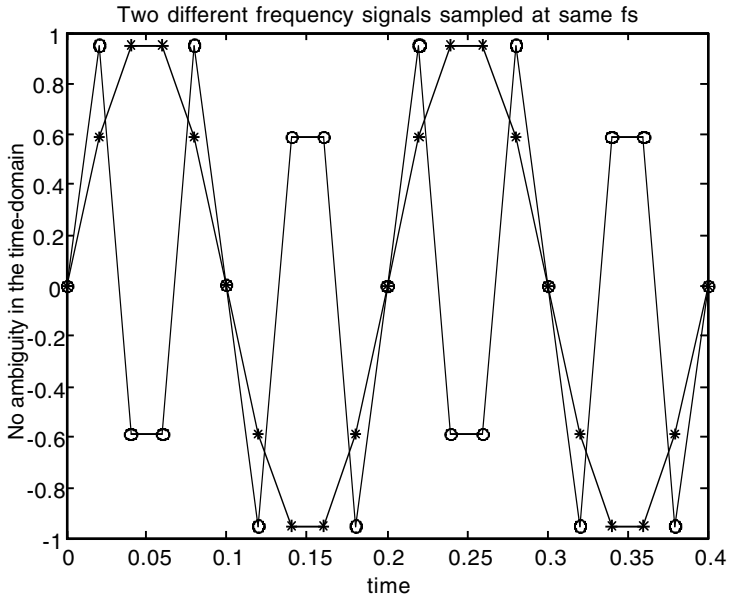

FIGURE 9.8 No ambiguity in time-domain.

1. Sample $x_1(t)$ and $x_2(t)$ at f_s = 50Hz and find $x_1(n)$ and $x_2(n)$.
2. Compare $x_1(n)$ and $x_2(n)$ and comment on the results.
3. Plot the continuous and the sampled signals.

Solution

When we sample $x_1(t)$ at f_s = 50 Hz we get

$$x_1(n) = \sin\big(2\pi(5)(n)(.02)\big) = \sin(.2\pi n)$$

By sampling $x_2(t)$ at f_s = 50 Hz we get

$$x_2(n) = \sin\big(2\pi(15)(n)(.02)\big) = \sin(.6\pi n)$$

But $x_1(n) \neq x_2(n)$ for any n. This happens because $f_s = 50 > 2(5)$, the frequency of $x_1(t)$ and $f_s > 2(15)$, the frequency of $x_2(t)$. In this we have satisfied the sampling theorem and hence $x_1(n) \neq x_2(n)$. The plots are shown in Figure 9.8 using the MATLAB script

```
n = 0:.02:.4;
x1 = sin(10*pi*n);
x2 = sin(30*pi*n); plot(n, x1); hold on; plot(n,x2);
plot(n, x1,'*'); plot(n, x2,'o');
```

```
Title('Two different frequency signals sampled at same fs');
ylabel('No ambiguity in the time-domain');
xlabel('time');
```

9.4.5 Sampling and Recovery of the Continuous Signal

Let $x(t)$ be a continuous-time signal that is sampled at the frequency f_s Hz where $f_s = 1/T_s$ and $t = nT_s$. We then have with T_s as the sampling period

$$x(n) = x(nT_s) \qquad -\infty < n < +\infty$$

In the frequency domain the continuous signal is

$$X(jw) = \int_{-\infty}^{+\infty} x(t)e^{-jwt}dt \tag{9.7}$$

and the discrete signal $x(n)$ is

$$X(e^{j\theta}) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\theta n} \tag{9.8}$$

where θ is the digital frequency.

The sampling of $x(t)$ can be viewed as multiplication of $x(t)$ by the train of impulses separated by T_s sec. We will write this train of impulses as

$$i(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_s) \tag{9.9}$$

The multiplication of $x(t)$ by the train of impulses can be written mathematically as

$$x_s(t) = \sum_{n=-\infty}^{+\infty} x(nT_s)\delta(t - nT_s) \tag{9.10}$$

where

$$i(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_s) \tag{9.11}$$

is the train of impulses. The frequency domain representation of $x_s(t)$ is

$$X_s(jw) = \sum_{n=-\infty}^{+\infty} x(nT_s) e^{-jnwT_s} \tag{9.12}$$

The train of impulses $i(t)$ can be represented using the Fourier series as

$$i(t) = \frac{1}{T_s} \sum_{n=-\infty}^{+\infty} e^{j\left(\frac{2\pi}{T_s}\right)kt} = \frac{1}{T_s} \sum_{n=-\infty}^{+\infty} e^{j2\pi f_s kt} \tag{9.13}$$

Now the product of $i(t)$ with $x(t)$ can be seen in a different expression using the Fourier series as

$$x_s(t) = x(t)\left[\frac{1}{T_s} \sum_{k=-\infty}^{+\infty} e^{j2\pi f_s kt}\right] \tag{9.14}$$

The frequency domain representation of Equation (9.14) for $x_s(t)$ is

$$X_s(jw) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X(jw - jk2\pi f_s)$$

or

$$X_s(jw) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X(jw - jkw_s) \tag{9.15}$$

where $w_s = 2\pi f_s$ is the sampling frequency.

From Equation (9.15) we can see that the continuous signal multiplied by the train of impulses in the frequency domain is a scaled and shifted version of the continuous signal $x(t)$ in the frequency domain. The scaling factor is $1/T_s$ and the shifting is $k(2\pi f_s) = kw_s$. We used this result in Section 9.4.2 when we explained ambiguity in the frequency domain. Again, if $f_s > 2f$ or $w_s > 2w$, where w_s is the sampling frequency and w is the highest radian frequency in the signal $x(t)$, we will have no aliasing and should be able to reconstruct the original signal $x(t)$. Further, we observe that

$$X(e^{j\theta}) = X_s(jw)\Big|_{w=\theta/T_s} \tag{9.16}$$

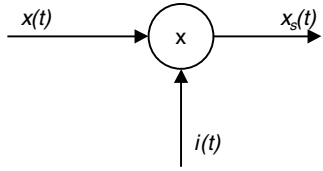

FIGURE 9.9 Ideal sampling.

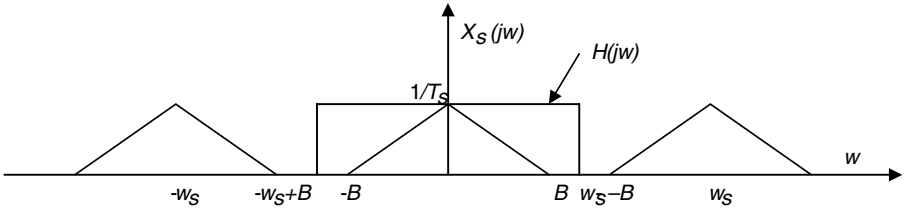

FIGURE 9.10 Recovering $x(t)$.

and by the same way

$$X_s(jw) = X\left(e^{j\theta}\right)\Big|_{\theta=wt} \tag{9.17}$$

This means that the discrete Fourier transform is obtained from the Fourier transform of the output of the ideal sampler shown in Figure 9.9 by simply substituting θ/T_s for w and is given as

$$X_s\left(e^{j\theta}\right) = \frac{1}{T_s}\sum_{k=-\infty}^{+\infty} X\left(j\frac{\theta}{T_s} - jk\frac{2\pi}{T_s}\right) \tag{9.18}$$

We call it ideal sampler because the duration of the impulse is ideally zero.

We can see from this mathematical development that if we have a signal $x(t)$ with f being the highest frequency in it, and if we sample $x(t)$ by multiplying it by the train of impulses where the period between the impulses $T_s = 1/f$ is the sampling period, then we can pass the product $i(t)x(t)$ through a lowpass filter with the cut-off frequency

$$B < w_c < w_s - B$$

where B is the highest nonzero radian frequency in $x(t)$ and w_s is the sampling frequency as illustrated in Figure 9.10. This ideal lowpass filter has the following characteristics:

$$H(jw) = \begin{cases} T_s & |w| \le w_c \\ 0 & |w| > w_c \end{cases} \tag{9.19}$$

To get an approximation to $x(t)$ after sampling, let us find the impulse response $h(t)$ for $H(jw)$. We have

$$h(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} H(jw)e^{jwt}dw = \frac{T_s}{2\pi}\int_{-w_c}^{w_c} e^{jwt}dw$$

or by carrying the integration we get

$$h(t) = \frac{\sin(w_c t)}{w_s(t/2)} \tag{9.20}$$

So $h(t)$ is the impulse response for the lowpass filter used to filter the original signal, or realistically, to approximate $x(t)$.

Now we can use the methods of Chapter 2 and use convolution to get the approximated $x(t)$. The sampled time signal is

$$x_s(t) = \sum_{n=-\infty}^{+\infty} x(nT_s)\delta(t - nT_s) \tag{9.21}$$

This $x_s(t)$ signal is now applied to the lowpass filter with $h(t)$ as its impulse response. Therefore, the output of the filter is now an approximation for $x(t)$ and is

$$x_a(t) = \sum_{n=-\infty}^{+\infty} x(nT_s)h(t - nT_s) \tag{9.22}$$

With

$$h(t) = \frac{\sin(w_c t)}{w_s(t/2)}$$

the shifted $h(t - nT_s)$ becomes

$$h(t - nT_s) = \frac{\sin(w_c(t - nT_s))}{w_s(t - nT_s)/2}$$

To simplify things let

$$w_c = w_s/2 = \frac{2\pi f_s}{2} = \pi/T_s$$

Then we get

$$h(t-nT_s)=\frac{\sin\left(\frac{\pi}{T_s}(t-nT_s)\right)}{\frac{\pi}{T_s}(t-nT_s)} \tag{9.23}$$

Finally the approximation to $x(t)$ becomes

$$x_a(t)=\sum_{n=-\infty}^{+\infty}x(nT_s)\frac{\sin\left(\frac{\pi}{T_s}(t-nT_s)\right)}{\frac{\pi}{T_s}(t-nT_s)} \tag{9.24}$$

You can see from the above expression that the approximation to $x(t)$ can be obtained by shifting the impulse response by nT_s and scaling by a factor of $x(nT_s)$ for $-\infty \le n \le \infty$.

9.5 How Do We Discretize the Derivative Operation?

The derivative $\frac{d}{dt}$ of the time signal $x(t)$ is $\frac{d}{dt}x(t)$ and at the point $t = nT_s$ is

$$\frac{d}{dt}x(t)\Big|_{t=nT_s}\approx\frac{x(nT_s+T_s)-x(nT_s)}{T_s} \tag{9.25}$$

Example 9.3

Consider the system described by the differential equation

$$\frac{d}{dt}y(t)+ay(t)=bx(t)$$

where $y(t)$ is the output and $x(t)$ is the input.

1. With no input find the output $y(t)$ for $t \ge 0$.
2. Use the discrete approximation for differentiation to find $y(t)$ for $t \ge 0$.
3. Compare the results of 1 and 2.

Solution

1. With $x(t)$ identically zero, the solution to

$$\frac{dy(t)}{dt} + ay(t) = bx(t)$$

is

$$y(t) = e^{-at}y(0) \qquad \text{for } t \geq 0$$

where $y(0)$ is the initial condition for the output $y(t)$.

2. With

$$\frac{d}{dt}x(t)\Big|_{t=nT_s} \approx \frac{x(nT_s + T_s) - x(nT_s)}{T_s}$$

the equation describing the system becomes

$$\frac{y(nT_s + T_s) - y(nT_s)}{T_s} + ay(nT_s) = 0$$

Writing $y(nT_s)$ as $y(n)$ and keeping in mind that we are sampling uniformly at T_s units of time, we arrive at

$$\frac{y(n+1) - y(n)}{T_s} + ay(n) = 0$$

This means that the distance between n and $n + 1$ is T_s units of time. With $n - 1$ replacing n we get

$$\frac{y(n) - y(n-1)}{T_s} + ay(n-1) = 0$$

Finally, the difference equation is written as

$$y(n) = (1 - aT_s)y(n-1)$$

The solution $y(n)$ in discrete form is

$$y(n) = (1 - aT_s)^n y(0) \qquad \text{for } n \geq 0$$

3. The solution in the continuous case was calculated as

$$y(t) = e^{-at}y(0)$$

With $t = nT_s$ and writing $y(nT_s)$ as $y(n)$, we get

$$y(n) = e^{-anT_s}\,y(0) = \left[1 - aT_s + \frac{a^2T_s^{\,2}}{2} - \ldots\right]^n y(0)$$

with

$$e^{-aT_s} = \left[1 - aT_s + \frac{a^2T_s^{\,2}}{2} - \ldots\right] \tag{9.26}$$

Notice that if aT_s is much smaller than unity in magnitude, then $a^2T_s^2$ and similarly $a^3T_s^3$ and $a^4T_s^4$ will have a negligible magnitude. So for $|aT_s|$ much smaller than unity, we can replace

$$y(n) = \left[1 - aT_s + \frac{a^2T_s^{\,2}}{2} - \ldots\right]^n y(0)$$

with

$$y(n) = \left(1 - aT_s\right)^n y(0) \qquad n \geq 0$$

which is the same solution we obtained using the discretization formula.

Example 9.4

Consider the second-order system

$$\frac{d^2}{dt}y(t) + a\frac{d}{dt}y(t) + by(t) = cx(t)$$

with the initial conditions $y(0)$ and $\frac{d}{dt}y(0)$. Write the equivalent difference equation that will approximate the above system.

Solution

With

$$\frac{d}{dt}x(t)\Big|_{t=nT_s} \approx \frac{x\left(nT_s + T_s\right) - x\left(nT_s\right)}{T_s}$$

we can write the second derivative as

$$\frac{d^2}{dt}y(t)\Big|_{t=nT_s} = \frac{\frac{d}{dt}y(t)\Big|_{t=nT_s+T_s} - \frac{d}{dt}y(t)\Big|_{t=nT_s}}{T_s} \tag{9.27}$$

By substituting for the approximation of $\frac{d}{dt}y(t)$ in Equation (9.27) we get

$$\frac{d^2}{dt}y(t)\Big|_{t=nT_s} = \frac{y(nT_s+2T_s) - 2y(nT_s+T_s) + y(nT_s)}{T_s^2} \tag{9.28}$$

Now we can substitute in the given differential equation describing the system with $y(nT_s)$ and $x(nT_s)$ written as $y(n)$ and $x(n)$ to get

$$\frac{y(n+2) - 2y(n+1) + y(n)}{T_s^2} + a\frac{y(n-1) - y(n)}{T_s} + by(n) = cx(n)$$

If we replace n with $n - 2$, we will get

$$\frac{y(n) - 2y(n-1) + y(n-2)}{T_s^2} + a\left(\frac{y(n-1) - y(n-2)}{T_s}\right) + by(n-2) = cx(n-2)$$

Multiplying the last equation by T_s^2 we get

$$y(n) + (aT_s - 2)y(n-1) + \left(1 - aT_s + bT_s^2\right)y(n-2) = cT_s^2x(n-2)$$

The solution of the above approximation to the given differential equation will start at $n = 2$, and thus we require discrete values for $y(0)$ and $y(1)$. The discrete value $y(0)$ is the continuous value $y(0)$, and the discrete value $y(1)$ is the continuous value $y(T_s)$. To get $y(T_s)$ we can use the approximation

$$\frac{d}{dt}y(0) = \frac{y(T_s) - y(0)}{T_s} \tag{9.29}$$

with $\frac{d}{dt}y(0)$ as given. Therefore, $y(T_s)$ is solved as

$$y(T_s) = T_s\frac{d}{dt}y(0) + y(0) = y(1)$$

9.6 Discretization of the State-Space Representation

Consider the multi-input multi-output system in state-space in the time-domain

$$\begin{aligned} \frac{d}{dt}\mathbf{v}(t) &= \mathbf{A}\mathbf{v}(t) + \mathbf{B}\mathbf{x}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{v}(t) + \mathbf{D}\mathbf{x}(t) \end{aligned} \tag{9.30}$$

where $\mathbf{v}(t)$ is the $n \times 1$ state vector, $\mathbf{x}(t)$ is the $r \times 1$ input vector, $\mathbf{y}(t)$ is the $p \times 1$ output vector, $\boldsymbol{A}$ is the $n \times n$ system matrix, $\mathbf{B}$ is the $n \times r$ input matrix, $\mathbf{C}$ is the $p \times n$ output matrix and $\mathbf{D}$ is the $p \times r$ transmission matrix. With the initial condition vector $\mathbf{v}(0)$ the solution vector $\mathbf{v}(t)$ is

$$\mathbf{v}(t) = e^{\mathbf{A}t}\mathbf{v}(0) + \int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{x}(\tau)d\tau \qquad t > 0 \tag{9.31}$$

If the starting time is set to t_0, then the solution to the state vector becomes

$$\mathbf{v}(t) = e^{\mathbf{A}(t-t_0)}\mathbf{v}(t_0) + \int_{t_0}^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{x}(\tau)d\tau \quad t > t_0$$

If we set $t_0 = nT_s$ and $t = nT_s + T_s$ in the above equation, we will get

$$\mathbf{v}(nT_s + T_s) = e^{\mathbf{A}T_s}\mathbf{v}(nT_s) + \int_{nT_s}^{nT_s+T_s} e^{\mathbf{A}(nT_s+T_s-\tau)}\mathbf{B}\mathbf{x}(\tau)d\tau$$

In the sampling process it is fair to assume that the input $x(t)$ is constant in the interval $nT_s \le t \le nT_s + T_s$. The solution $\mathbf{v}(t)$ in the discrete form is now

$$\mathbf{v}(nT_s + T_s) = e^{\mathbf{A}T_s}\mathbf{v}(nT_s) + \int_{nT_s}^{nT_s+T_s} e^{\mathbf{A}(nT_s+T_s-\tau)}\mathbf{B}\mathbf{d}\tau\,\mathbf{x}(nT_s) \tag{9.32}$$

If we now let

$\int_{nT_s}^{nT_s+T_s} e^{\mathbf{A}(nT_s+T_s-\tau)}\mathbf{B}d\tau = \Psi$, $e^{\mathbf{A}T_s} = \Phi$, and $nT_s + T_s - \tau = \lambda$, then we have

$$\Psi = -\int_{T_s}^{0} e^{\mathbf{A}\lambda}\mathbf{B}d\lambda = \int_{0}^{T_s} e^{\mathbf{A}\lambda}\mathbf{B}d\lambda \tag{9.33}$$

Therefore, the solution to the state vector becomes

$$\mathbf{v}\left(nT_s + T_s\right) = \Phi\mathbf{v}\left(nT_s\right) + \Psi\,\mathbf{x}\left(nT_s\right) \tag{9.34}$$

The output is obtained by substituting nT_s for t to get

$$\mathbf{y}\left(nT_s\right) = \mathbf{C}\mathbf{v}\left(nT_s\right) + \mathbf{D}\mathbf{x}\left(nT_s\right) \tag{9.35}$$

If we now write $\mathbf{x}(nT_s)$ as $\mathbf{x}(n)$ and $\mathbf{v}(nT_s)$ as $\mathbf{v}(n)$ we finally have

$$\mathbf{v}(n+1) = \Phi\mathbf{v}(n) + \Psi\,\mathbf{x}(n) \tag{9.36}$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{v}(n) + \mathbf{D}\mathbf{x}(n) \tag{9.37}$$

Example 9.5

Consider the state equations

$$\frac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} 0 & 1 \\ 0 & -1 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(t)$$

$$\mathbf{y}(t) = (1\ \ 0)\mathbf{v}(t)$$

What is the discrete state-space approximation?

Solution

To find Ψ and Φ we need to find $e^{\mathbf{A}t}$ first. $e^{\mathbf{A}t}$ is the inverse transform of $(s\mathbf{I} - \mathbf{A})^{-1}$ and

$$(s\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} s & -1 \\ 0 & s+1 \end{pmatrix}^{-1} = \begin{pmatrix} \dfrac{1}{s} & \dfrac{1}{s(s+1)} \\ 0 & \dfrac{1}{s+1} \end{pmatrix}$$

Taking the inverse transform on the above matrix we get the transition matrix

$$e^{\mathbf{A}t} = \begin{pmatrix} t & 1 - e^{-t} \\ 0 & e^{-t} \end{pmatrix}$$

The discrete matrix Ψ is given as

$$\Psi = \int_0^{T_s} \begin{pmatrix} \lambda & 1-e^{-\lambda} \\ 0 & e^{-\lambda} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} d\lambda$$

With $T_s = 1$,

$$\Psi = \begin{pmatrix} e^{-1} \\ 1-e^{-1} \end{pmatrix}$$

The transition matrix at this sampling time is

$$\Phi = \begin{pmatrix} 1 & 1-e^{-1} \\ 0 & e^{-1} \end{pmatrix}$$

The discrete approximation to the continuous state-space vector representation is

$$\begin{pmatrix} v_1(n+1) \\ v_2(n+2) \end{pmatrix} = \begin{pmatrix} 1 & 1-e^{-1} \\ 0 & e^{-1} \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix} + \begin{pmatrix} e^{-1} \\ 1-e^{-1} \end{pmatrix} \mathbf{x}(n)$$

and the output approximation is given as

$$\mathbf{y}(n) = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} v_1(n) \\ v_2(n) \end{pmatrix}$$

9.7 The Bilinear Transformation and the Relationship between the Laplace-Domain and the *z*-Domain Representations

A continuous time system has the input $x(t)$, the transfer function $H(s)$, and the output $y(t)$. A discrete system has the input $x(n)$, the transfer function $H(z)$, and the output $y(n)$. The discrete system has an output that matches the output of the continuous system at discrete points nT_s where n is an integer and T_s is the sampling or the discretization interval. We are assuming here that the discrete system is the continuous system discretized. In frequency, let $Y(e^{j\theta})$ denote the discrete Fourier transform of the output $y(n)$ so that

$$Y\left(e^{j\theta}\right) = \sum_{n=-\infty}^{+\infty} y\left(nT_s\right)e^{-j\theta n} \tag{9.38}$$

where $y(nT_s)$ is $y(t)$ taken at nT_s discrete points. We mentioned earlier in this chapter that we sample the signal $y(t)$ by multiplying it by the train of impulses $i(t)$ to get $y_s(t)$. Let $Y_s(w)$ be the Fourier transform of the sampled signal $y_s(t)$ and write

$$Y_s(w) = \sum_{n=-\infty}^{+\infty} y\left(nT_s\right)e^{-jwnT_s} \tag{9.39}$$

We have learned before that the digital frequency is

$$\theta = wT_s$$

where w is the analogue frequency. Then we can write the Fourier transform as

$$Y_s\left(\frac{\theta}{T_s}\right) = \sum_{n=-\infty}^{\infty} y\left(nT_s\right)e^{-j\theta n} \tag{9.40}$$

From Equations (9.38) and (9.39) we can deduce that

$$Y\left(e^{j\theta}\right) = Y_s(w)\Big|_{w=\frac{\theta}{T_s}} = Y_s\left(\frac{\theta}{T_s}\right)$$

Again, $Y(e^{j\theta})$ is the discrete Fourier transform of $y(n)$ and $Y_s(w)$ is the continuous Fourier transform of $y_s(t)$ which is the product of the signal $y(t)$ by the impulse train $i(t)$.

In many cases in practice we will be using a digital filter represented by the transfer function $H(z)$ to filter an analogue or continuous signal. In many cases, too, we will be designing the analogue filter $H(s)$, then we can transform it to the digital filter $H(z)$, where we have to make sure that

$$Y\left(e^{j\theta}\right) = Y_s(w)\Big|_{w=\frac{\theta}{T_s}} \tag{9.41}$$

in the range $-\pi \le \theta \le \pi$ since $Y(e^{j\theta})$ is periodic with period 2π.

For the continuous system we have the input-output relationship

$$Y(w) = H(w)\ X(w)$$

where $H(w) = H(s)\,|_{s=jw}$. In the design of $H(z)$, the digital filter, using $H(w)$, the analogue filter, we will assume that the magnitude of $H(w)$ is approximately zero for $w > w_s/2$, where $w_s = 2\pi/T_s$. Also it is assumed that the magnitude of the input $X(w)$ is approximately zero for $w > w_s/2$. With $Y_s(w)$ and $X_s(w)$ as the Fourier transforms of the sampled output and input $y_s(t)$ and $x_s(t)$, we have

$$Y_s(w) = H(w)\ X_s(w) \quad \text{for } -\frac{w_s}{2} \le w \le \frac{w_s}{2} \tag{9.42}$$

With Equation (9.41) and (9.42), we can write

$$Y\left(e^{j\theta}\right) = H(w)X_s(w)\Big|_{w=\frac{\theta}{T_s}} \quad \text{for } -\pi \le \theta \le \pi \tag{9.43}$$

We also know that

$$X_s(w) = \sum_{n=-\infty}^{+\infty} x\left(nT_s\right)e^{-j\frac{\theta}{T_s}nT_s} \tag{9.44}$$

If we set $w = \theta/T_s$ in Equation (9.44), we will get

$$X_s\left(\frac{\theta}{T_s}\right) = \sum_{n=-\infty}^{+\infty} x\left(nT_s\right)e^{-j\frac{\theta}{T_s}nT_s} = \sum_{n=-\infty}^{+\infty} x\left(nT_s\right)e^{-j\theta n} = X\left(e^{j\theta}\right) \tag{9.45}$$

which is the discrete Fourier transform of $x(nT_s)$.

Now we can write the input-output relation

$$Y\left(e^{j\theta}\right) = H(w)X_s(w)\Big|_{w=\frac{\theta}{T_s}} = H\left(\frac{\theta}{T_s}\right)X_s\left(\frac{\theta}{T_s}\right) = H\left(\frac{\theta}{T_s}\right)X\left(e^{j\theta}\right) \tag{9.46}$$

with the discrete input-output relationship written as

$$Y\left(e^{j\theta}\right) = H\left(e^{j\theta}\right)X\left(e^{j\theta}\right) \tag{9.47}$$

We see from Equations (9.46) and (9.47) for $Y(e^{j\theta})$ that

$$H\left(\frac{\theta}{T_s}\right) = H\left(e^{j\theta}\right) \tag{9.48}$$

where it is clear that the digital transfer function $H(e^{j\theta})$ is obtained from the continuous $H(w)$ by the substitution $w = \theta/T_s$. Note that if $z = e^{j\theta}$ and if $s = jw = 1/T_s \ln(z)$ then

$$H\left(e^{j\theta}\right) = H(z) = H(s)\Big|_{s=\frac{1}{T_s}\ln z} \tag{9.49}$$

To verify Equation (9.49) set $z = e^{j\theta}$ and simplify to get

$$H\left(e^{j\theta}\right) = H(s)\Big|_{s=\frac{1}{T_s}\ln e^{j\theta}} = H(s)\Big|_{s=\frac{1}{T_s}j\theta} = H\left(\frac{j\theta}{T_s}\right) = H(jw) = H(s)$$

The approximation to $\ln(z)$ is given by

$$\ln(z) = 2\frac{z-1}{z+1}$$

Therefore, if we design $H(s)$ for the continuous filter, we can obtain $H(z)$ for the digital filter by performing the substitution

$$s = \frac{2}{T_s}\frac{z-1}{z+1} \tag{9.50}$$

With this transformation we can have the digital filter transfer function as

$$H(z) = H(s)\Big|_{s=\frac{2}{T_s}\frac{z-1}{z+1}} \tag{9.51}$$

This is the bilinear transformation equation that is the most popular nowadays. Other transformations that are used to obtain $H(z)$ from $H(s)$ will be listed in Section 9.8 without further details of their derivations. The bilinear transform in Equation (9.50) is superior.

If the cut-off frequency of the continuous filter is w_c then from $w = \theta/T_s$ we have

$$\theta_c = w_c T_s \tag{9.52}$$

where θ_c is the corresponding approximation to w_c due to the approximation in the transformation. We know that points on the unit circle in the z-domain will be mapped on the jw axis in the Laplace domain. With $z = e^{j\theta}$ we can write

$$H(z)\big|_{z=e^{j\theta}} = H\left(\frac{2}{T_s}\frac{e^{j\theta}-1}{e^{j\theta}+1}\right) \tag{9.53}$$

The point $\frac{2}{T_s}\frac{e^{j\theta}-1}{e^{j\theta}+1}$ in the z-plane must be mapped into the point jw in the Laplace domain. We then write

$$jw = \frac{2}{T_s}\frac{e^{j\theta}-1}{e^{j\theta}+1} \tag{9.54}$$

Solving for θ in Equation (9.54), we get

$$\theta_c = 2\tan^{-1}\frac{w_c T_s}{2} \tag{9.55}$$

Example 9.6

Consider the continuous filter described by the transfer function

$$H(s) = \frac{1}{s+3}$$

What is $H(z)$?

Solution

With $s = \frac{2}{T_s}\frac{z-1}{z+1}$ and assuming $T_s = 0.1$, we get

$$H(z) = \frac{1}{\left(\frac{2}{.1}\frac{z-1}{z+1}\right)+3}$$

After simplification we arrive at

$$H(z) = \frac{\frac{1}{23}(z+1)}{z-17/23}$$

We will use MATLAB to plot $|H(z)|$ and $|H(s)|$ as in the following script:

```
w=-pi:pi/10:pi;
theta=w*.1;
nz=[1/23 1/23]; dz=[1 -17/23];
```

```
Hz=freqz(nz,dz,theta);
ns=[1]; ds=[1 3];
plot(w,abs(Hz),'o'); hold on;
Hs=freqs(ns,ds,w);
plot(w,abs(Hs),'*');
legend('Discrete transfer function','Continuous transfer
 function',0);
title('Magnitude frequency response for H(s) and its
 approximate H(z)')
xlabel('Radian frequency');
```

The plots are shown in Figure 9.11. Notice in the MATLAB script above that we called the function `freqz` and passed to it the digital frequency `theta=w*.1`. This is according to the relation that must be satisfied: $\theta = wT_s$. θ is the digital frequency and w is the analogue frequency. The cut-off frequency of the continuous filter is $w_c = 3$ and the cut-off frequency of the digital filter is approximately

$$\theta_c = 2\tan^{-1}\left(\frac{3(.1)}{2}\right) = 2\tan^{-1}\left(\frac{3}{20}\right) = 0.279$$

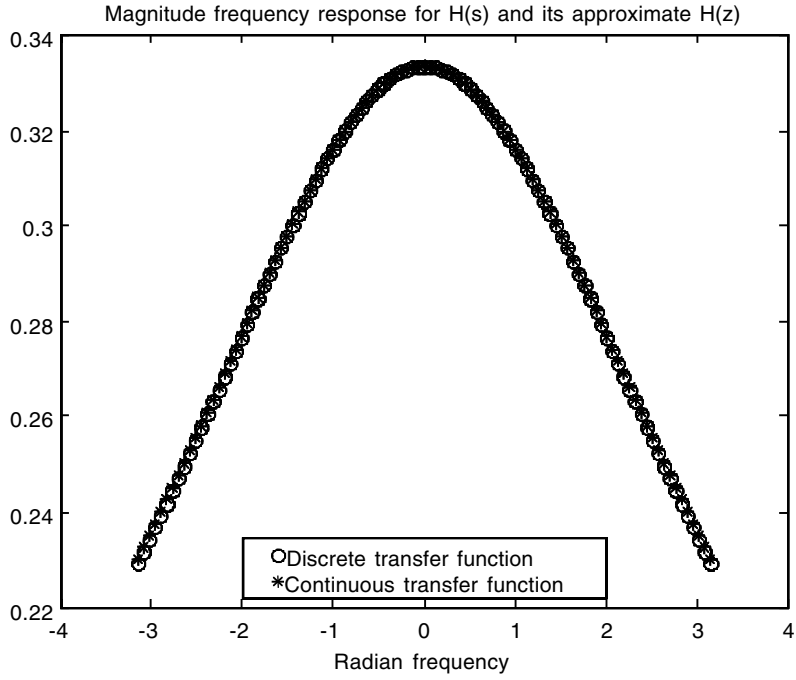

FIGURE 9.11 Plots for Example 9.6.

9.8 Other Transformation Methods

There are many methods that are used to transform a transfer function $H(s)$ to a transfer function $H(z)$. Next we will present some of them without further details of how these methods were derived. We are interested in the application of these methods when we are discretizing continuous systems.

9.8.1 Impulse Invariance Method

If $H(s)$ is strictly proper (the degree of the numerator is less than the degree of the denominator), then the digital filter $H(z)$ is the sampling period T_s multiplied by the z-transform of $H(s)$.

9.8.2 The Step Invariance Method

Given the continuous transfer function $H(s)$, the discrete transfer function $H(z)$ is $(1 - z^{-1})$ times the z-transform of $H(s)/s$.

9.8.3 The Forward Difference Method

The transformation here is carried by the substitution

$$s = \frac{z-1}{T_s}$$

9.8.4 The Backward Difference Method

To go from $H(s)$ to $H(z)$ we use the transformation

$$s = \frac{z-1}{T_s z}$$

9.8.5 The Bilinear Transformation

This method was discussed earlier in this chapter and is carried by the transformation

$$s = \frac{2}{T_s}\frac{z-1}{z+1}$$

Example 9.7

Consider now the filter $H(s)$ as

$$H(s) = \frac{1}{s+3}$$

and let us find $H(z)$ using the five methods described above with $T_s = .1$

Solution

Using the impulse invariance method

With $H(s) = \dfrac{1}{s+3}$, the impulse response is

$$h(t) = e^{-3t}$$

The sampled $h(t)$ is then

$$h\left(nT_s\right) = e^{-3nT_s} = \left(e^{-3T_s}\right)^n$$

The z-transform of $(e^{-3T_w})^n$ is $\dfrac{z}{z - e^{-3T_s}}$. Therefore,

$$\mathrm{H(z)} = T_s\left[\frac{z}{z - e^{-3T_s}}\right] = \frac{.1z}{z - e^{-.3}}$$

Using the step invariance method

In this case,

$$\frac{H(s)}{s} = \frac{1}{s(s+3)} = \frac{1/3}{s} + \frac{-1/3}{s+3}$$

and the inverse-transform of $\dfrac{H(s)}{s}$ is $\dfrac{1}{3}\left(1 - e^{-3t}\right)$. If we evaluate this time signal at $t = nT_s$ we get

$$\frac{1}{3} - \frac{1}{3}e^{-3nT_s} = \frac{1}{3} - \frac{1}{3}\left(e^{-3T_s}\right)^n$$

with the z-transform as

$$\frac{1/3z}{z-1} - \frac{1}{3}\frac{z}{z - e^{-3T_s}} = \frac{1/3z}{z-1} - \frac{1}{3}\frac{z}{z - e^{-.3}}$$

Therefore, the transfer function $H(z)$ is

$$H(z)=\frac{z-1}{z}\left[\frac{1/3z}{z-1}-\frac{1}{3}\frac{z}{z-e^{-.3}}\right]=\frac{1}{3}-\frac{1}{3}\frac{z-1}{z-e^{-.3}}=\frac{1}{3}\left[\frac{z-e^{-.3}-z+1}{z-e^{-.3}}\right]=\frac{1}{3}\left[\frac{1-e^{-.3}}{z-e^{-.3}}\right]$$

Using the forward difference method

With $s=\frac{z-1}{T_s}$ the discrete transfer function is

$$H(z)=\frac{1}{\frac{z-1}{T_s}+3}=\frac{T_s}{z-1+3T_s}=\frac{.1}{z-.7}$$

Using the backward difference method

With $s=\frac{z-1}{zT_s}$,

$$H(z)=\frac{1}{\frac{z-1}{zT_s}+3}=\frac{zT_s}{z-1+3zT_s}=\frac{.1z}{1.3z-1}=\frac{\frac{1}{13}z}{z-\frac{10}{13}}$$

Using the bilinear transformation method

With $s=\frac{2}{T_s}\frac{z-1}{z+1}=\frac{2}{.1}\frac{z-1}{z+1}=20\frac{z-1}{z+1}$,

$$H(z)=\frac{1}{20\frac{z-1}{z+1}+3}=\frac{(z+1)}{20z-20+3z+3}=\frac{\frac{1}{23}(z+1)}{z-\frac{17}{23}}$$

To compare the quality of the transformations we will plot the magnitude of $H(s)$ and all of the five transfer functions $H(z)$. We will do that using the following MATLAB script:

```
w=-pi:pi/50:pi;
theta=w*.1;
nc=[1]; dc=[1 3];
nii=[.1 0]; dii=[1 -exp(-.3)];
nsi=[0 (1/3)*(1-exp(-.3))]; dsi=[1 -1*exp(-.3)];
nfd=[0 .1]; dfd=[1 -.7];
nbd=[1/13 0]; dbd=[1 -10/13];
nbt=[1/23 1/23]; dbt=[1 -17/23];
Hs=freqs(nc,dc,w);
```

```
Hzii=freqz(nii,dii,theta);
Hzsi=freqz(nsi,dsi,theta);
Hzfd=freqz(nfd,dfd,theta);
Hzbd=freqz(nbd,dbd,theta);
Hzbt=freqz(nbt,dbt,theta);
plot(w,abs(Hs)); hold on;
plot(w,abs(Hzii),'*');
plot(w,abs(Hzsi),'o');
plot(w,abs(Hzfd),'s');
plot(w,abs(Hzbd),'d');
plot(w,abs(Hzbt),'.');
legend('CTF','DTF II','DTF SI','DTF FD','DTF BD','DTF BT',0);
title('Magnitude frequency response for H(s) and its
 approximate H(z)')
xlabel('Radian frequency');
```

The plots are shown in Figure 9.12. Note that every transformation is different and produces different approximation to $H(s)$. This is to say that every transformation should be selected carefully depending on the problem at hand. The Impulse Invariance method was the worst in Example 9.7. The step invariant and the bilinear transformations were almost exact in approximating the continuous transfer function. If you decrease the sampling time T_s from 0.1 to 0.01, the Impulse Invariant method will be a good approximate to the continuous transfer function.

9.9 Some Insights

9.9.1 The Choice of the Sampling Interval T_s

As long as the sampling theorem is satisfied we will not have any aliasing. The sampling theorem is satisfied by choosing f_s such that $f_s > 2f_m$ where f_m is the maximum frequency in the signal to be sampled. The smaller the T_s the better the approximation as long as we are satisfying the sampling theorem. In practice, we do not need tosample at very small T_s, since that will require huge amounts of computations that may not actually be needed.

9.9.2 The Effect of Choosing T_s on the Dynamics of the System

If the transfer function of the system described by $H(s)$ has complex poles, then discretizing this continuous transfer function can result in the loss of some dynamics (change of location of poles) if the sampling period T_s is according to the relation

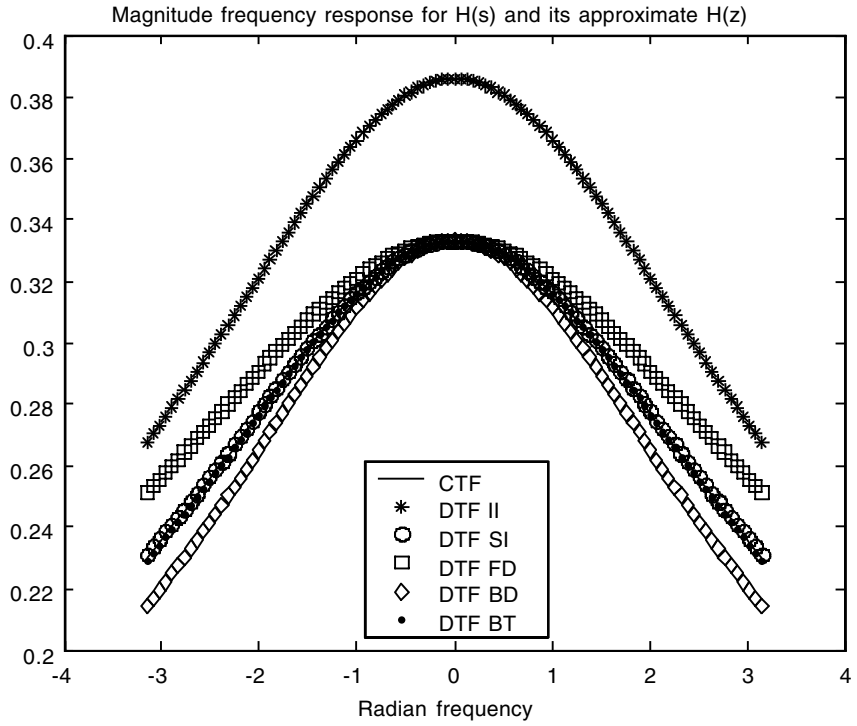

FIGURE 9.12 Plots for Example 9.7.

$$\frac{\pi}{T_s} = \frac{1}{2}|b|$$

where b is the imaginary part of the complex pole $a \pm jb$. The reason is that the mapping from the s-plane to the z-plane is not one-to-one.

9.9.3 Does Sampling Introduce Additional Zeros for the Transfer Function $H(z)$?

If the difference between the number poles and zeros in $H(s)$ is zero then $H(z)$ will have zero difference between the number of its poles and zeros due to sampling with any sampling interval T_s. If the difference between the number of poles and zeros in $H(s)$ is nonzero, then when we discretize $H(s)$ to get $H(z)$, new zeros will be introduced in $H(z)$. The number of the new zeros in $H(z)$ is equal to the difference between the number of poles and zeros in $H(s)$ minus 1.

9.10 End of Chapter Examples

EOCE 9.1

Consider the continuous transfer function

$$H(s) = \frac{1}{s^2 - 4}$$

1. Find the magnitude of the complex poles.
2. Choose T_s such that $\pi/T_s = 1/2$ times the magnitude of the imaginary part of the complex pole.
3. Discretize $H(s)$ with T_s as obtained in part 2.
4. Comment on the results

Solution

1. We will use MATLAB to find the complex poles and their magnitude.

```
den=[1 0 4];
eigenvalues=(roots(den))
```

to get

```
eigenvalues =
0.0000 2.0000i
0.0000 –2.0000i
```

2. With $\frac{\pi}{T_s} = \frac{1}{2}(2)$, the sampling time is $T_s = \pi$.
3. We will use MATLAB to get $H(z)$ and write the MATLAB script

```
num=[1]; den=[ 1 0 4];
Ts=pi;
[A, B, C, D]=tf2ss(num,den);
%next we will dicritize A and B with Ts =.879
[disA, disB]=c2d(A,B,Ts);
%now back to transfer function representation
[disnum,disden]=ss2tf(disA, disB, C, D)
```

to get

disnum = 1.0e – 015 * [0 0.2220 –0.2220]
disden = 1.0000 –2.0000 1.0000

This means that the discrete transfer function is given by

$$H(z) = \frac{(.2220z - .2220) * 10^{-15}}{z^2 - 2z + 1}$$

4. Notice that the dynamics of $H(s)$ has changed. An additional zero was introduced in the discrete transfer function. The system was marginally stable, and after discretization it became unstable.

EOCE 9.2

Consider the two systems

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

$$H(s) = \frac{s}{s + 1}$$

Show that for any T_s we will always get (2 – 1 = 1) extra zero using the first transfer function but no extra zeros in using the second.

Solution

To do that we will use the MATLAB script

```
%For the first transfer function
num=[1]; den=[ 1 0 1];
Ts=.1;
[A, B, C, D]=tf2ss(num,den);
%next we will discretize A and B with Ts =.879
[disA, disB]=c2d(A,B,Ts);
%now back to transfer function representation
[disnum,disden]=ss2tf(disA, disB, C, D)
```

to get

$$H(z) = \frac{.0050 + .0050z}{z^2 - 1.9900z + 1}$$

with one additional zero. If we now change T_s from .1 to 10 we will get

$$H(z) = \frac{1.8391 + 1.8391z}{z^2 + 1.6781z + 1}$$

with only one additional zero as expected. You can choose any value for T_s in the above script and you will always get one additional zero.

For the second transfer function we can use the same script as above with little modification. For T_s = .1, we get the discrete transfer function

$$H(z) = \frac{z - 1}{(z - .9048)}$$

with no additional zeros as expected, because the number of poles is equal to the number of zeros in $H(s)$. Notice that the zero at $s = 0$ for $H(s)$ is mapped into the unit circle in $H(z)$ as expected. If we change T_s to 10 we will not get any additional zeros. We will get

$$H(z) = \frac{z - 1}{z}$$

Notice that the s-plane zero at zero still maps into the z-plane at $z = 1$. However, since we have chosen a sampling interval of 10, the location of the s-plane pole is changed. Hence the dynamics will change as well.

EOCE 9.3

Consider the continuous system

$$\frac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} -2 & 0 \\ 0 & -1 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(t)$$

$$\mathbf{y}(t) = (0 \quad 1)\mathbf{v}(t)$$

1. Find $y(t)$ due to $x(t) = 0$ and $\mathbf{v}(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
2. Find $y(n)$ as in part 1 by discretizing the state equations.
3. Comment on the plots for $y(t)$ and $y(n)$.

Solution

1. Using the Laplace transform method we have

$$\mathbf{V}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{v}(0)$$

The transition matrix is the inverse transform of $(s\mathbf{I} - \mathbf{A})^{-1}$. We have

$$(s\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{1}{s+2} & 0 \\ 0 & \dfrac{1}{s+1} \end{pmatrix}$$

and the transition matrix is then

$$e^{\mathbf{A}t} = \begin{pmatrix} e^{-2t} & 0 \\ 0 & e^{-t} \end{pmatrix}$$

and the state vector becomes

$$\mathbf{V}(s) = \begin{pmatrix} \dfrac{1}{s+2} & 0 \\ 0 & \dfrac{1}{s+1} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \dfrac{1}{s+1} \end{pmatrix}$$

with

$$\mathbf{v}(t) = \begin{pmatrix} 0 \\ e^{-t} \end{pmatrix} \text{ and } \mathbf{y}(t) = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ e^{-t} \end{pmatrix} = e^{-t} \quad t > 0$$

2. To find $y(n)$ we calculate $e^{\mathbf{A}T_s}$ using $e^{\mathbf{A}t}$. In doing that we get

$$\phi = \begin{pmatrix} e^{-2T_s} & 0 \\ 0 & e^{-T_s} \end{pmatrix}$$

To find $y(n)$ we still need

$$\Psi = \int_0^{T_s} \begin{pmatrix} e^{-2t} & 0 \\ 0 & e^{-t} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} dt = \int_0^{T_s} \begin{pmatrix} 0 \\ e^{-t} \end{pmatrix} dt = \begin{pmatrix} 0 \\ -e^{-T_s} + 1 \end{pmatrix}$$

Finally the discrete state and output equations are

$$v(nT_s + T_s) = \begin{pmatrix} e^{-2T_s} & 0 \\ 0 & e^{-T_s} \end{pmatrix} \mathbf{v}(nT_s) + \begin{pmatrix} 0 \\ -e^{-T_s} + 1 \end{pmatrix} \mathbf{x}(nT_s)$$

$$\mathbf{y}(nT_s) = \begin{pmatrix} 0 & 1 \end{pmatrix} \mathbf{v}(nT_s) + (0)\mathbf{x}(nT_s)$$

To plot $y(n)$ we need the initial conditions in the discrete domain. We have

$$v_1(nT_s) = v_1(t)\Big|_{t=nT_s}$$

$$v_2(nT_s) = v_2(t)\Big|_{t=nT_s}$$

Therefore, the initial conditions are the same and we can use the following MATLAB script to plot $y(t)$ and $y(n)$, the approximation to $y(t)$. Two methods of discretization will be used in the following MATLAB script. The first is the method described in this chapter. The second is the method that MATLAB uses, the `c2d` (continuous to discrete) function.

```
%The continuous solution
A=[ -2 0; 0 -1]; B=[0;1]; C=[ 0 1]; D=[0];
vin=[0;1]; t=0:.1:5; x=zeros(size(t)); %zero input
[yt, vt]=lsim(A ,B ,C ,D ,x,t, vin);
%The discrete solution
%pick up a sampling interval
Ts=.1; n=0:50; x=zeros(size(n));
vin=[0;1];% same initial conditions
Ad=[exp(-2*Ts) 0; 0 exp(-Ts)];
Bd=[ 0; -1*exp(-Ts)+1];
Cd=[0 1];
Dd=[0];
[yd, vd]=dlsim(Ad, Bd, Cd, Dd, x,vin);
%Dicritize A and B with Ts =.1 using the c2d MATLAB function
[disA, disB]=c2d(A,B,Ts);
[ydm vdm]=dlsim(disA, disB, C,D,x,vin);
plot(t,yt); xlabel('Time(sec)');
title('The continuous and the discrete solutions. Ts
 =0.1');
hold on
plot(n*Ts,yd,'*'); xlabel('Time(sec)');
plot(n*Ts,ydm,'o');
legend('Cont sol','Disc sol: book method','Disc sol MATLAB
 method',0);
```

and the plot is shown in Figure 9.13.

4. The approximation can vary by changing the sampling interval T_s.

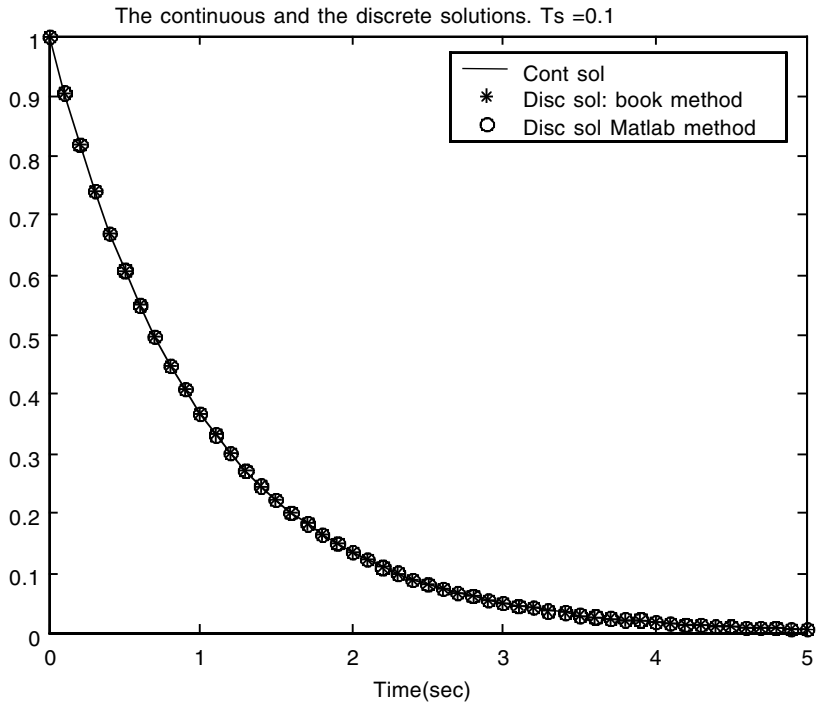

FIGURE 9.13 Plots for EOCE 9.3.

EOCE 9.4

Consider the continuous system described by the differential equation

$$\frac{d^2}{dt}y(t) + y(t) = x(t)$$

with $y(0) = 0$, $\frac{d}{dt}y(0) = 0$, and $x(t) = u(t)$. Find $y(t)$ and its approximate solution $y(n)$ by discretizing the differential equation and plot the results.

Solution

As in Example 9.3 and with $a = 0$, $b = 1$ and $c = 1$, the approximate difference equation is

$$y(n) - 2y(n-1) + \left(1 + T_s^2\right)y(n-2) = T_s^2 x(n-2)$$

We will use MATLAB to plot $y(t)$ and $y(n)$ for the value of $T_s = .1$ sec as in the following script:

```
% for the continuous system;
nc = [1]; dc =[1 0 1];
t = 0: .1: 10; % simulation for 5 seconds
xc = ones(size(t)); % input unit step signal
yc = lsim(nc, dc, xc, t);
plot(t,yc); hold on;
title('The continuous and the discrete outputs: Ts =.1');
xlabel('Time(sec)');
% for the discretize system;
Ts = .1; % sampling period
nd = [0 0 Ts*Ts]; dd=[ 1 -2 1+Ts*Ts]
n = 0:1:100 ; %simulations for 5 seconds(.1*50)
xd = ones(size(n)); yd = dlsim(nd, dd, xd); plot (n*Ts, yd,'*');
legend('Continuous solution','Discrete solution',0);
```

The plots are shown in Figure 9.14. You can see that the approximation is not very accurate. The original continuous system is oscillatory; its poles are on the imaginary axis. Since we are discretizing this system we should be careful with the sampling interval we choose. As seen in Figures 9.14 and 9.15, the approximation is much better with small T_s. The simulations for T_s = .5 and .01 are shown in Figure 9.15.

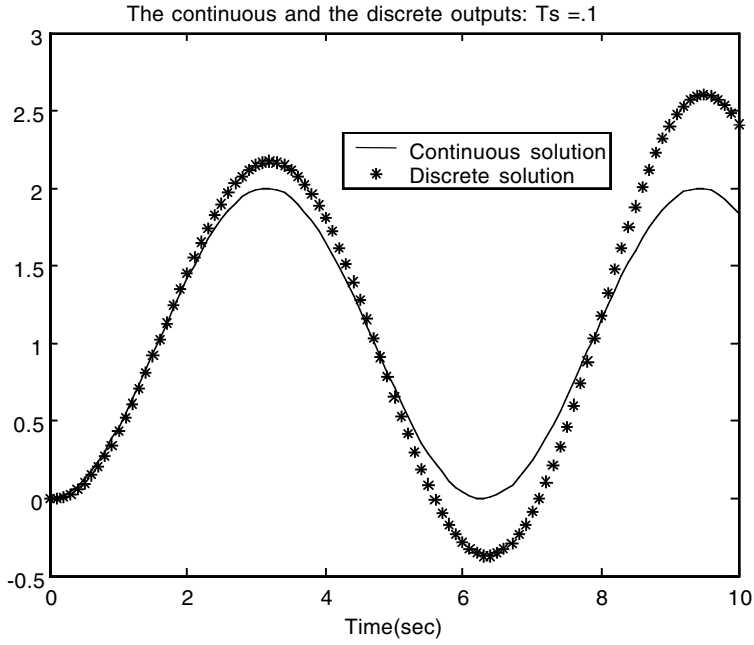

FIGURE 9.14 Plots for EOCE 9.4.

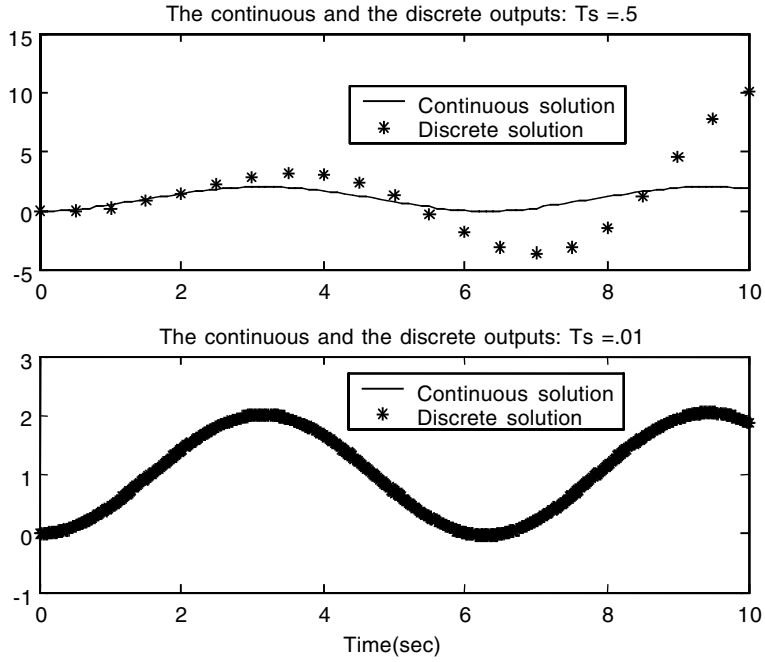

FIGURE 9.15 Plots for EOCE 9.4.

EOCE 9.5

Consider the state-space continuous systems

1. $\dfrac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 0 \\ 1 \end{pmatrix}\mathbf{x}(t)$
2. $\dfrac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -2 & -3 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\mathbf{x}(t)$

Find the approximate discrete representation for the two systems.

Solution

For the first system we have

$$\mathbf{v}\left(nT_s + T_s\right) = \Phi\mathbf{v}\left(nT_s\right) + \Psi\,\mathbf{x}\left(nT_s\right)$$

with

$$e^{\mathbf{A}T_s} = \Phi \text{ and}$$

$$\Psi = \int_0^{T_s} e^{\mathbf{A}\lambda}\mathbf{B}d\lambda$$

For Φ we calculate first $e^{\mathbf{A}t}$ as the inverse Laplace transform of $(s\mathbf{I} - \mathbf{A})^{-1}$.

$$(s\mathbf{I} - \mathbf{A}) = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix} = \begin{pmatrix} s & -1 \\ 2 & s+3 \end{pmatrix}$$

and

$$(s\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} s & -1 \\ 0 & s+1 \end{pmatrix}^{-1} = \begin{pmatrix} \dfrac{s+3}{(s+1)(s+2)} & \dfrac{1}{(s+1)(s+2)} \\ \dfrac{-2}{(s+1)(s+2)} & \dfrac{s}{(s+1)(s+2)} \end{pmatrix}$$

Using partial fraction expansion we get

$$(s\mathbf{I} - \mathbf{A})^{-1} = \begin{pmatrix} \dfrac{-1}{(s+2)} + \dfrac{2}{(s+1)} & \dfrac{-1}{(s+1)} + \dfrac{1}{(s+1)} \\ \dfrac{2}{(s+2)} - \dfrac{-2}{(s+1)} & \dfrac{2}{(s+2)} - \dfrac{1}{(s+1)} \end{pmatrix}$$

and the transition matrix is

$$e^{\mathbf{A}t} = \begin{pmatrix} -e^{-2t} + 2e^{-t} & -e^{-2t} + e^{-t} \\ 2e^{-2t} - 2e^{-t} & 2e^{-2t} - e^{-t} \end{pmatrix}$$

The discrete transition matrix is

$$e^{\mathbf{A}T_s} = \Phi(T_s) = \begin{pmatrix} -e^{-2T_s} + 2e^{-T_s} & -e^{-2T_s} + e^{-T_s} \\ 2e^{-2T_s} - 2e^{-T_s} & 2e^{-2T_s} - e^{-T_s} \end{pmatrix}$$

The matrix $\Psi(T_s)$ is given by

$$\Psi = \int_0^{T_s} \begin{pmatrix} -e^{-2t} + e^{-t} \\ 2e^{-2t} - e^{-t} \end{pmatrix} dt = \begin{pmatrix} e^{-2T_s}/2 - e^{-T_s} + 1/2 \\ -e^{-2T_s} + e^{-T_s} \end{pmatrix}$$

For $T_s = 1$, we have

$$\Phi(1) = \begin{pmatrix} 0.6004 & .2325 \\ -.4651 & -.0972 \end{pmatrix}$$

and

$$\Psi(1) = \begin{pmatrix} .1998 \\ .2325 \end{pmatrix}$$

We can use MATLAB to find $\Phi(1)$ and $\Psi(1)$ as in the following script:

```
A=[0 1; -2 -3];
B=[0;1];
Ts=1;
[dA,dB]=c2d(A,B,1)%Ts=1
```

to get

```
dA =
    0.6004   2325
    –.4651   –.0972

dB =
    .1998
    .2325
```

For the second system we can use MATLAB to find $\Phi(1)$ and $\Psi(1)$ as in the following script:

```
A=[0 1 0; 0 0 1;-1 -2 -3];
B=[0;0;1];
Ts=1;
[dA,dB]=c2d(A,B,1)%Ts=1
```

to get

```
dA =
    0.9166  0.8092  0.1966
    –0.1966  0.5234  0.2194
    –0.2194  –0.6355  –0.1349

dB =
    0.0834
    0.1966
    0.2194
```

EOCE 9.6

Consider the continuous system

$$H(s) = \frac{4s + 11}{s^2 + 7s + 10}$$

Find two transfer functions, $H(z)$, to represent the continuous system.

Solution

We will use the Impulse Invariance method first. From

$$H(s) = \frac{4s + 11}{s^2 + 7s + 10} = \frac{1}{s + 2} + \frac{3}{s + 5}$$

we have the impulse response $h(t)$ as

$$h(t) = e^{-2t} + 3e^{-5t}$$

and

$$h(nT_s) = e^{-2nT_s} + 3e^{-5nT_s} = \left(e^{-2T_s}\right)^n + 3\left(e^{-5T_s}\right)^n$$

If we take the z-transform or $h(nT_s)$ we will get the first z-domain representation as

$$H(z) = \frac{T_s z}{z - e^{-2T_s}} + \frac{3T_s z}{z - e^{-5T_s}}$$

By using the Step Invariant method we have

$$H(s)/s = \frac{4s + 11}{s\left(s^2 + 7s + 10\right)} = \frac{-.6}{s + 5} - \frac{.5}{s + 5} + \frac{1.1}{s}$$

with the time signal as

$$h(t) = -.6e^{-5t} - .5e^{-2t} + 1.1$$

and

$$h(nT_s) = -.6e^{-5nT_s} - .5e^{-2nT_s} + 1.1 = -.6\left(e^{-5T_s}\right)^n - .5\left(e^{-2T_s}\right)^n + 1.1$$

The z-transform of $h(nT_s)$ is

$$H(z) = \frac{-.6z}{z - e^{-5T_s}} - \frac{.5z}{z - e^{-2T_s}} + \frac{1.1z}{z - 1}$$

Thus the second z-domain representation is

$$H(z) = \frac{z-1}{z}\left(\frac{-.6z}{z - e^{-5T_s}} - \frac{.5z}{z - e^{-2T_s}} + \frac{1.1z}{z - 1}\right)$$

An appropriate value for T_s in the two z-domain representations will approximate the continuous filter $H(s)$.

EOCE 9.7

A continuous signal $x(t)$ is given. Its continuous Fourier transform is given by

$$X(jw) = \int_{-\infty}^{\infty} x(t)e^{-jwt}dt$$

The inverse continuous transform is also given by

$$x(t) = \frac{1}{2\pi} = \int_{-\infty}^{\infty} X(jw)e^{-jwt}dw$$

If we sample $x(t)$ at the sampling interval T_s sec, we will obtain the discrete signal

$$x(n) = x(t)\Big|_{t=nT_s}$$

Keep in mind that as long as $f_s > 2f_m$, where f_m is the highest frequency in $x(t)$, we can reconstruct $x(t)$ from $x(n)$.

Solution

In this example we will use MATLAB to study the effect of sampling in the frequency domain. To do that we need to represent $x(t)$ and $X(jw)$ in MATLAB. We will sample $x(t)$ on a finite gird with t_g being the grid interval where $t_g \ll T_s$. In this case, the MATLAB simulated continuous signal $x(t)$ is

$$x_g(k) = x\left(kt_g\right)$$

In the same way, and using the relation

$$X(jw) = \int_{-\infty}^{\infty} x(t)e^{-jwt}dt$$

the MATLAB-simulated Fourier transform can be written as

$$X_g(jw) = t_g \sum_k x_g(k)e^{-jwkt_g}$$

To see this let us consider the continuous signal

$$x(t) = e^{-1000t}$$

with its Fourier transform calculated as

$$X(jw) = \frac{1}{jw + 1000}$$

For $w \geq 2\pi(2000)$ the magnitude of $X(jw)$ is approximately zero. Also, for $t = .005$, e^{-1000t} is approximately zero and we can approximate $x(t)$ in the interval from 0 to .005 seconds. Let us choose the grid interval, t_g, to be 0.00005 which is less than 1/2(2000). In the following MATLAB script we will plot the simulated continuous signal $x(t)$ and its simulated continuous Fourier transform $X(jw)$.

```
tg = 0.00005;
t = 0: tg: .005;
xg =(exp(-1000*t));
wmax = 2*pi*2000;
M = 400; m = 0: 1: M;
w = m*wmax/M;% frequency for 0 to 2*pi*2000
Xg =tg*(exp(-j).^(w'*t))*xg'
Xg = abs(Xg); subplot(2, 1, 1);
plot(t, xg); xlabel('Time(sec)');ylabel('x(t)');
title('x(t) and its Continuous Fourier transform X(jw): an
 approximation to both');
subplot(2, 1, 2); plot(w/(2*pi*1000), Xg);
xlabel('Frequency in KHz'); ylabel('Magnitude: continuous
 X(jw)');
```

The plots are shown in Figure 9.16. To study the effect of sampling in the frequency domain, let us sample $x(t)$ at f_s = 8000 Hz and 200 Hz. Notice that the Nyquist rate for $x(t)$ is 2(2000) = 4000 Hz. In the first case, we are satisfying the Nyquist rate but we are not in the second case. We hope to see the aliasing effects on the magnitude plots for the discrete Fourier transforms due to this type of sampling. We will use the following MATLAB script to do that. We will start with f_s = 8000 Hz, the no-aliasing case.

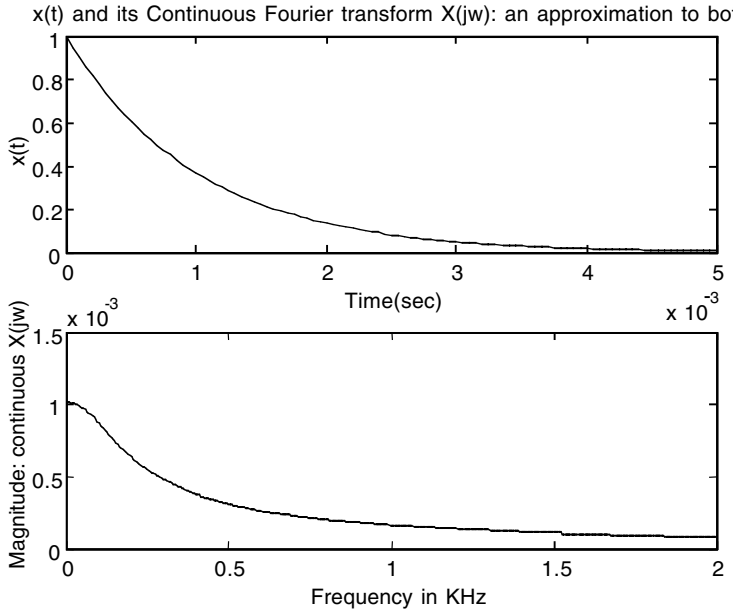

FIGURE 9.16 Plots for EOCE 9.7.

```
tg = 0.00005;
t = 0: tg: .005;
xg =(exp(-1000*t));
%the above is for the continuous signal x(t)
fs=8000; Ts=1/fs;
n=0:1:40; %for time duration of .005 seconds
xd=exp(-1000*n*Ts);
%the above is for x(t) sampled at fs=8000 Hz
M = 400; m = 0: 1: M;
w = m*pi/M;% frequency for 0 to pi
Xd =xd*exp(-j*n'*w);% the discrete Fourier transform
Xd = abs(Xd); subplot(2, 1, 1);
stem(n*Ts, xd); xlabel('Time(sec)');ylabel('x(n): fs= 8000
 Hz');
title('x(n) and its Discrete Fourier transform Xd(jw): an
 approximation');
subplot(2, 1, 2); plot(w, Xd);
xlabel('Frequency: Radians/second'); ylabel('Magnitude of
 Xd(jw)');
axis([0 3 0 10]);
```

The plots are shown in Figure 9.17. You can see from the plots in Figure 9.17 that there is a scaling factor to the magnitude plot for $Xd(jw)$ due to sampling. The scaling factor of 8000 Hz is present. The maximum magnitude is around

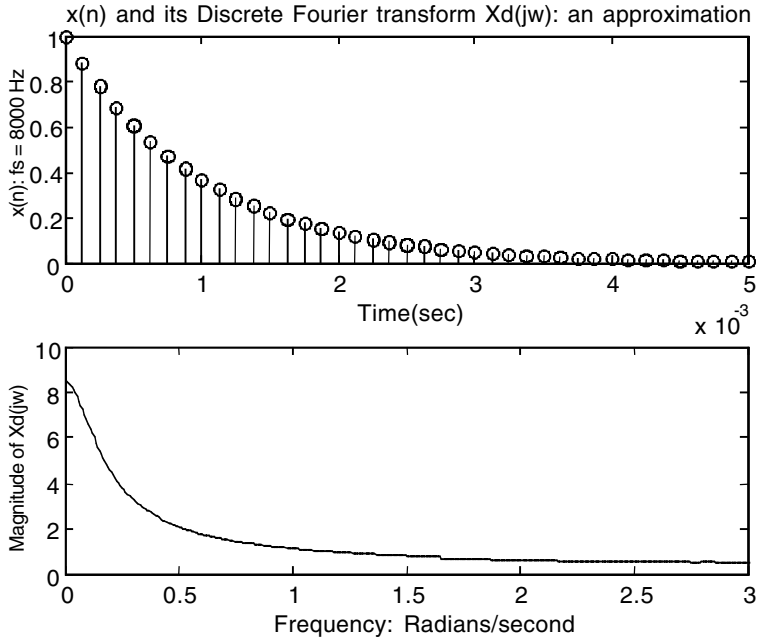

FIGURE 9.17 Plots for EOCE 9.7 with no aliasing.

8 compared with 0.001 for the continuous case. We can also see in the figure that for f_s = 8000 Hz, there is no aliasing present since the plot for the continuous Fourier transform in Figure 9.16 is similar to the one in Figure 9.17.

For the sampling frequency of 200 Hz (the sampling frequency criteria is not satisfied), the magnitude plot for the discrete Fourier transform of $x(n)$ is scaled by 200 Hz. We can also see in Figure 9.18 that for f_s = 200 Hz, there is aliasing present since the plot for the continuous Fourier transform in Figure 9.16 is not similar to the one in Figure 9.18.

EOCE 9.8

In this example we will study the effects of sampling in the time domain. We have derived the formula for the reconstructed signal as

$$x_a(t) = \sum_{n=-\infty}^{\infty} x(nT_s) \frac{\sin\left(\frac{\pi}{T_s}(t - nT_s)\right)}{\frac{\pi}{T_s}(t - nT_s)}$$

where $x_a(t)$ is the approximation to the signal $x_a(t)$ obtained by the reconstruction operation from the discrete signal $x_a(n)$. A closer look at the above approximation shows that it is an interpolation of infinite order.

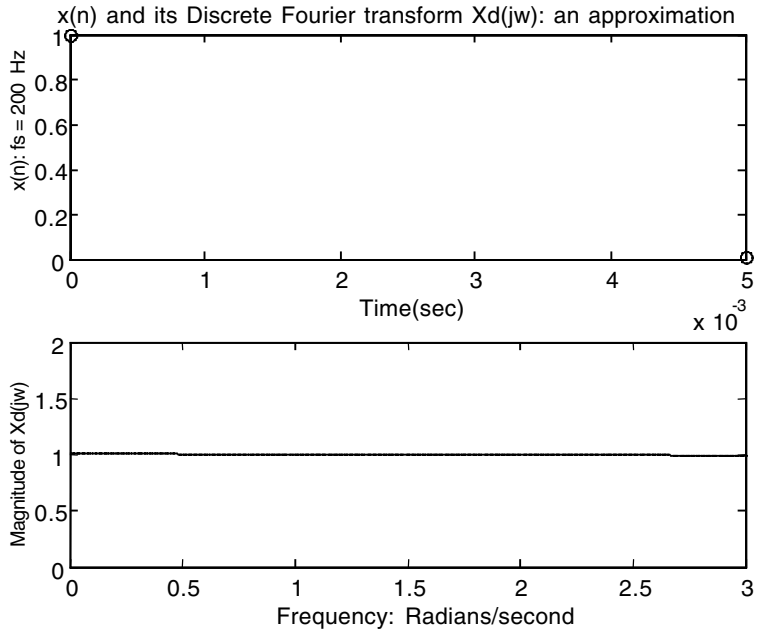

FIGURE 9.18 Plots for EOCE 9.7 with aliasing.

Solution

In practice, the zero-order hold method is employed and is implemented in MATLAB by the function `stairs`. Using the zero-order-hold method, the signal $x(n)$ is converted into a weighted impulse train and then passed through the filter

$$h(t) = \begin{cases} 1 & 0 \le t \le T_s \\ 0 & \text{otherwise} \end{cases}$$

The second practical approach is the first-order-hold interpolation that is implemented in MATLAB by the function `plot`. In this case $x(n)$ again is converted into a weighted impulse train and then passed through the filter

$$h(t) = \begin{cases} 1 + \dfrac{t}{T_s} & 0 \le t \le T_s \\ 1 - \dfrac{t}{T_s} & T_s \le t \le 2T_s \\ 0 & \text{otherwise} \end{cases}$$

The third practical approach is to use the cubic-spline interpolation which does not require that we pass the converted $x(n)$ through a filter like the previous two reconstruction methods. This method is implemented in MATLAB using the function `spline` that takes as inputs to it the samples $x(n)$, the sampling intervals nT_s and the time grid for which the values for $x_a(t)$ are desired.

We will use the same signal $x(t)$ that we used in the last EOCE to demonstrate the aliasing in time. The continuous signal was given as

$$x(t) = e^{-1000t}$$

with its Fourier transform calculated as

$$X(jw) = \frac{1}{jw + 1000}$$

For $t = .005$, e^{-1000t} is approximately zero and we can approximate $x(t)$ in the interval from 0 to .005 seconds. Let us choose the grid interval, t_g, to be 0.00005 which is less than 1/2(2000). In the following MATLAB script we will use the sampling frequency of $f_s = 8000$ Hz and the three MATLAB implementations for the reconstruction of $x(t)$.

```
fs=8000; Ts=1/fs;
n=0:1:40; %for time duration of .005 seconds
xn=exp(-1000*n*Ts);
subplot(1, 3, 1);stairs(n*Ts, xn);
xlabel('Time(sec)');
ylabel('Reconstruction with zero-order-hold');
hold on; stem(n*Ts, xn);hold off;
subplot(1, 3, 2); plot(n*Ts, xn);
xlabel('Time(sec)')
ylabel('Reconstruction using first-order-hold');
hold on; stem(n*Ts,xn);hold off
tg=0.00005;%the grid for the spline
t=0:tg:0.005; subplot(1,3,3);
 xd=spline(n*Ts,xn,t);plot(t,xd);
ylabel('Reconstruction using spline'); hold on;
 stem(n*Ts,xn);
xlabel('Time(sec)')
```

The plots are shown in Figure 9.19.

If we use $f_s = 200$ Hz and the three MATLAB reconstruction implementations, we get the plots in Figure 9.20. You can clearly see that for $f_s = 200$ Hz, a frequency below the Nyquist rate, aliasing is obvious since the plots do not match with the fine signal $x(t)$, even with small error.

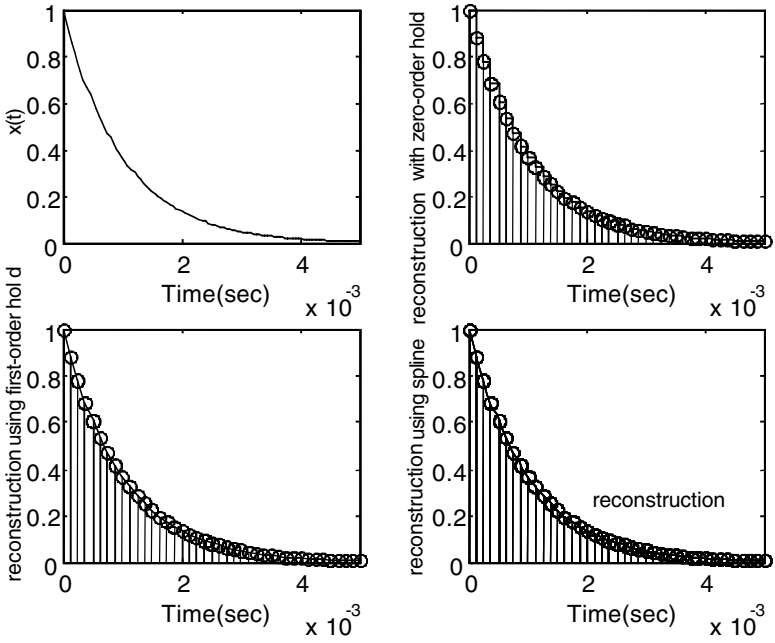

FIGURE 9.19 Plots for EOCE 9.8 without aliasing.

9.11 End of Chapter Problems

EOCP 9.1

Consider the two signals

$$x_1(t) = \sin\left(2\pi 10^3 t\right)$$

$$x_2(t) = \sin\left(4\pi 10^3 t\right)$$

1. If you sample the above two signals at the sampling frequency of 1 kHz, would you be able to distinguish between the two signals?
2. Repeat, using $f_s = 10^3/3$ kHz.
3. Use MATLAB to plot $x_1(t)$, $x_1(n)$, $x_2(t)$ and $x_2(n)$.

EOCP 9.2

Give an example of five continuous signals that when sampled at $f_s = 10^4$ Hz,

1. Would produce five discrete signals each of which is unique
2. Would produce five discrete signals where

$$x_1(n) = x_2(n) = x_3(n) = x_4(n) = x_5(n)$$

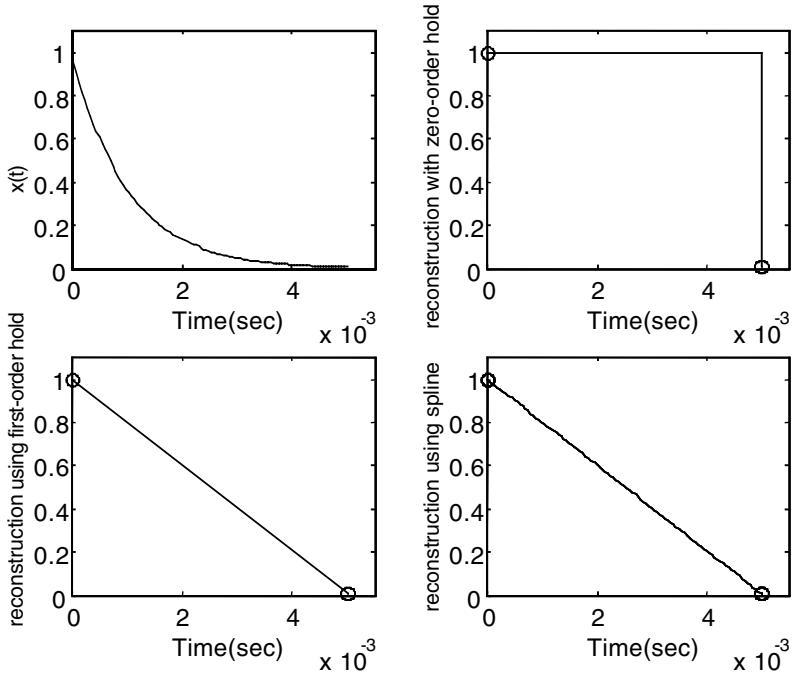

FIGURE 9.20 Plots for EOCE 9.8 with aliasing.

EOCP 9.3

Consider the following continuous time signals

$$x(t) = 1e^{-2000|t|} \quad -\infty < t < \infty$$

$$x(t) = 100e^{-500t} t \geq 0$$

Follow a procedure similar to that in EOCE 9.8 and EOCE 9.7 to

1. Find the approximate maximum frequencies for both signals where the magnitude spectra approach zero magnitude.
2. Find appropriate time intervals where the two signals approach a value of zero outside these intervals.
3. Sample both signals at $f_s > 2f_m$ and find $x(n)$ for both where f_{m} is the approximate maximum frequency after which the magnitude spectra is approximately zero.
4. Sample both signals at $f_s < f_m$.
5. Look at the effect of sampling in the frequency domain for both signals.

6. Look at the effect of sampling in the time domain.
7. Draw a conclusion regarding observations on this problem.

EOCP 9.4

Repeat EOCP 9.3 for

$$x(t) = 100e^{-|t|} \quad -\infty < t < \infty$$

$$x(t) = \frac{e^{-t}}{100} \quad t \geq 0$$

EOCP 9.5

Find the approximate difference equations and the corresponding discrete initial conditions for the following first-order differential equations:

1. $\frac{d}{dt}y(t) + 2y(t) = x(t)$ with $y(0) = 2$
2. $2\frac{d}{dt}y(t) - y(t) = \frac{x(t)}{3}$, with $y(0) = \frac{1}{3}$
3. $\frac{d}{dt}y(t) - 3y(t) = 0$, with $y(0) = 10$
4. $\frac{d}{dt}y(t) + 10y(t) = \frac{1}{2}x(t)$, with y(0) = 0

EOCP 9.6

Discretize the following continuous systems along with their initial conditions.

1. $\frac{d^2}{dt}y(t) + \frac{d}{dt}y(t) + y(t) = x(t)$

   $y(0) = \frac{d}{dt}y(0) = 0$

2. $\frac{d^2}{d}y(t) + y(t) = x(t)$

   $y(0) = 0, \frac{d}{dt}y(0) = 1$

3. $\frac{d^2}{dt}y(t) + 2\frac{d}{dt}y(t) = x(t)$

   $y(0) = \frac{d}{dt}y(0) = 0$

4. $$\frac{d^2}{dt}y(t) - y(t) = 10x(t)$$
$$y(0) = \frac{d}{dt}y(0) = 0$$

5. $$\frac{d^2}{dt} + 5\frac{d}{dt}y(t) + 6y(t) = 0$$
$$y(0) = 10\frac{d}{dt}y(0) = 0$$

EOCP 9.7

Consider the following time domain state-space systems:

1. $$\frac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} 0 & 1 \\ -2 & -5 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 0 \\ 10 \end{pmatrix}\mathbf{x}(t)$$
$$\mathbf{y}(t) = \begin{pmatrix} 1 & 0 \end{pmatrix}\mathbf{v}(t) + 0\mathbf{x}(t)$$

2. $$\frac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} -3 & 0 \\ 0 & -4 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 1 \\ 1 \end{pmatrix}\mathbf{x}(t)$$
$$\mathbf{y}(t) = \begin{pmatrix} 0 & 1 \end{pmatrix}\mathbf{v}(t) + \mathbf{x}(t)$$

3. $$\frac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -6 & -10 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\mathbf{x}(t)$$
$$y(t) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}\mathbf{v}(t) + 0\mathbf{x}(t)$$

4. $$\frac{d}{dt}\mathbf{v}(t) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -3 \end{pmatrix}\mathbf{v}(t) + \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}\mathbf{x}(t)$$
$$\mathbf{y}(t) = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}\mathbf{v}(t) + \mathbf{x}(t)$$

a) Analytically, discretize the first two systems with zero initial conditions.
b) Use MATLAB to plot $y(t)$ and $y(n)$ for values of T_s of your choice for the first two systems if $x(t) = t\sin(t)$.
c) Use MATLAB to put the last two systems in a discretized form. Use zero initial condition with $x(t) = u(t)$ to plot $y(t)$ and $y(n)$ for both systems with appropriate T_s.

EOCP 9.8

Consider the systems in the *s*-domain.

$$H(s) = \frac{1}{s+1}$$

$$H(s) = \frac{100}{s+10}$$

1. What is the cut-off frequency w_c for both systems?
2. What is the digital cut-off frequency θ_c for both systems?
3. Use the transformation

$$s = \frac{2}{T_s}\frac{z-1}{z+1}$$

   to find the approximate $H(z)$ for both systems
4. Use MATLAB to plot the frequency response for the two systems in the *s*-domain and the *z*-domain.
5. Give your observations.

EOCP 9.9

We have discussed five methods by which we can find an approximation to the *s*-domain system. Use these five methods to find five different discrete transfer functions, $H(z)$, for each of the systems.

$$H(s) = \frac{1}{s+5}$$

$$H(s) = \frac{1}{s^2+7s+14}$$

EOCP 9.10

Consider the systems

$$H(s) = \frac{s+1}{s^2+10}$$

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

1. Choose T_s such that

$$\frac{\pi}{T_s} = \frac{1}{2}|b|$$

   where b is the imaginary part of the complex roots for the two systems
2. Find $H(z)$, the approximation to $H(s)$, with such T_s.
3. Plot using MATLAB the frequency response for both systems in the s-domain and in the z-domain.
4. What comments can you make?

10

Infinite Impulse Response (IIR) Filter Design

10.1 Introduction

After Chapter 8, where we discussed the design process of analogue filters, and Chapter 9, where we presented different methods of frequency transformations, we believe that we are ready to design infinite impulse response (IIR) digital filters. The design of IIR digital filters in this chapter will be based on the design of analogue continuous filters (Chapter 8) and frequency transformations (Chapter 9). The most important frequency transformation used to transform an analogue continuous filter to an IIR digital filter is the Bilinear transformation. In this chapter we will use the Bilinear and the Impulse Invariance transformations in the design process. Basically, we will be designing IIR digital filters from analogue continuous filters using these two frequency transformations. IIR digital filters are the filters in which the output at a certain instant depends on the current and previous inputs and outputs on or prior to that instant. We will be designing proper, rational and stable IIR digital filters only in this chapter (the reasons are explained later in the chapter). Also, satisfying the requirements for the magnitude and the phase in designing IIR filters is very difficult. Thus we will deal with either the magnitude or the phase requirements. In this chapter we will design by satisfying the magnitude requirements, hoping that the damage resulting from not satisfying the phase requirements will be minimal. The digital IIR filter transfer function we will be designing will be obtained using the following steps:

1. Start with an analogue lowpass prototype filter with cut-off frequency of 1 rad/sec.
2. Use frequency transformation to transform the analogue lowpass prototype to a lowpass, highpass, bandpass or bandstop analogue filter with the desired cut-off frequency/frequencies.
3. Use the bilinear or the impulse invariance transformation to transform the analogue filter to its approximate IIR digital filter.

The following is a set of frequency transformations transforming lowpass filters with system function $H(s')$ to a different configuration function $H(s)$.

Lowpass to lowpass: $s' = \dfrac{s}{w_c}$

Lowpass to highpass: $s' = \dfrac{w_c}{s}$

Lowpass to bandpass: $s' = \dfrac{s^2 + w_{cl}w_{cu}}{s\left(w_{cu} - w_{cl}\right)}$

Lowpass to bandstop: $s' = \dfrac{s\left(w_{cu} - w_{cl}\right)}{s^2 + w_{cl}w_{cu}}$

where in the above transformations w_{cl} is the lower cut-off frequency and w_{cu} is the upper cut-off frequency.

Looking at the previous transformations we can see that the transformation is not linear except for the lowpass to lowpass case. However, this nonlinearity provides no difficulty since the filter being transformed is approximately constant in the frequency band of interest. Therefore, while the frequency spacing of the ripple peaks and valleys are affected, the amount of ripple is still the same. Furthermore, in transforming the lowpass filter to bandstop or bandpass filter, the substitution is second order, so the output filter is twice the order as the input filter.

10.2 The Design Process

In designing stable and proper IIR digital filters we will use the wealth of knowledge and experience that is available in literature about designing analogue filters. In particular, we will use the information presented in Chapter 8 to design analogue filters and then use the two transformation methods discussed in Chapter 9 to obtain the IIR digital filter transfer function. The two methods are the bilinear and the impulse invariance methods. The relationship between the analogue frequency w and the digital frequency θ is $w = \theta/T_s$ where T_s is the sampling period.

10.2.1 Design Based on the Impulse Invariance Method

For transforming an analogue continuous filter to an IIR digital filter using the Impulse Invariance method, the digital IIR filter $H(z)$ is obtained by multiplying the sampling period T_s by the z-transform of $H(s)$. If we call the eigenvalue for the continuous system λ, then we have the following transform pair

$$e^{\lambda t} \Leftrightarrow \frac{1}{s-\lambda} \tag{10.1}$$

At $t = nT_s$, $e^{\lambda t} = e^{\lambda nT_s} = (e^{\lambda T_s})^n$ and the z-transform of $(e^{\lambda T_s})^n$ is obtained as in the following pair

$$\left(e^{\lambda T_s}\right)^n \Leftrightarrow \frac{1}{1-e^{\lambda T_s}z^{-1}} \tag{10.2}$$

If the poles of $H(s)$ are all distinct and if $H(s)$ is strictly proper, then $H(s)$ can be expanded in partial fraction form as

$$H(s) = \frac{k_1}{s-\lambda_1} + \frac{k_2}{s-\lambda_2} + \ldots + \frac{k_n}{s-\lambda_n} \tag{10.3}$$

and the impulse response of $H(s)$ is

$$h(t) = k_1 e^{\lambda_1 t} + k_2 e^{\lambda_2 t} + \ldots + k_n e^{\lambda_n t} \tag{10.4}$$

With $t = nT_s$ we have

$$h(n) = k_1 e^{\lambda_1 nT_s} + k_2 e^{\lambda_2 nT_s} + \ldots + k_n e^{\lambda_n nT_s} \tag{10.5}$$

In this case, the z-transform of $h(n)$ in Equation (10.5) is

$$H(z) = \frac{k_1}{1-e^{\lambda_1 T_s}z^{-1}} + \frac{k_2}{1-e^{\lambda_2 T_s}z^{-1}} + \ldots + \frac{k_n}{1-e^{\lambda_n T_s}z^{-1}} \tag{10.6}$$

With Equation (10.6), we can finally write the transfer function of the IIR digital filter using the Impulse Invariance method as

$$H(z)_{\text{II}} = T_s\left[\frac{k_1}{1-e^{\lambda_1 T_s}z^{-1}} + \frac{k_2}{1-e^{\lambda_2 T_s}z^{-1}} + .. + \frac{k_n}{1-e^{\lambda_n T_s}z^{-1}}\right] \tag{10.7}$$

Example 10.1

Let us start with the analogue prototype first-order lowpass filter $H_{\text{plp}}(s') = \frac{1}{s'+1}$. We can use the frequency transformation $s' = s/w_c$ to transform the prototype lowpass analogue filter into a lowpass analogue filter with cut-off frequency w_c. With a cut-off frequency of 3 rad/sec the new transfer function is

$$H(s) = \frac{1}{\frac{s}{3}+1} = \frac{3}{s+3}.$$

Next let us find $H(z)$ that corresponds to $H(s) = \frac{3}{s+3}$. Use the Impulse Invariance method with $T_s = .1$.

Solution

With $H(s) = \frac{3}{s+3}$, the impulse response is $h(t) = 3e^{-3t}$. The sampled $h(t)$ is then

$$h(nT_s) = 3e^{-3nT_s} = 3\left(e^{-3T_s}\right)^n$$

The z-transform of $(e^{-3T_s})^n$ is $\frac{z}{z - e^{-3T_s}}$. Therefore, the digital transfer function is

$$\mathrm{H(z)} = T_s\left[\frac{3z}{z - e^{-3T_s}}\right] = \frac{.3z}{z - e^{-.3}} = \frac{.3z}{z - 0.7408}$$

When the order of the filter is high, the computations become very complex. We will try to use MATLAB to accomplish the impulse invariance transformation. First we need to use partial fraction expansion on $H(s)$. The MATLAB function `residue` is used to perform partial fraction expansion on $H(s)$. Remember that the poles in the z-domain are at $e^{\lambda_n T_s}$ where n is the n^{th} pole. Next, we will use the MATLAB function `residuez` to get the numerator and the denominator coefficients of the transfer function $H(z)$. Then we will multiply the resulting $H(z)$ by T_s to find the final Impulse Invariant transformed digital IIR filter. This is accomplished for the example at hand as in the following MATLAB script.

```
Ts=0.1;
nums= [3];%numerator for H(s)
dens=[1 3];
[res,poles,const]=residue(nums, dens);
[numz denz]=residuez(res, exp(poles*Ts), const);
numz=numz*Ts %Final H(z)
denz
```

The result is the same as we obtained analytically with $H(z) = \frac{.3z}{z - 0.7408}$.

Let us now write a MATLAB function to accomplish the Impulse Invariance transformation since we will be using it over and over in the examples to follow. The function is given next.

```
function [numz denz]=impinv(nums, dens, Ts)%f in function is
 lower case
[res,poles,const]=residue(nums, dens)
[numz denz]=residuez(res, exp(poles*Ts), const);
numz=numz*Ts %to get the Final H(z)
```

We will call the function `impinv` as

```
[numz denz]=impinv(nums, dens, 0.1)
```

MATLAB contains the function `impinvar` that is used to produce IIR digital filters based on the Impulse Invariance transformation. The syntax is

```
[numz,denz] = impinvar(nums,dens,fs)
```

This function creates a digital filter with numerator and denominator coefficients `numz` and `denz` respectively, whose impulse response is equal to the impulse response of the analog filter with coefficients `nums` and `dens` and a sampling frequency `fs`. The `nums` and `dens` coefficients will be scaled by `fs`. If you don't specify `fs`, it defaults to 1 Hz. The syntax

```
[numz,denz] = impinvar(nums,dens,fs,tol)
```

uses the tolerance `tol` for grouping repeated poles together. Default value is 0.001, i.e., 0.1%. The repeated pole case works, but is limited by the ability of the function `roots` to factor such polynomials. Either of the two functions, `impinv`, which is written above or `impinvar`, which is available with in MATLAB, can be used.

10.2.2 Design Based on the Bilinear Transform Method

For transforming an analogue continuous filter to an IIR digital filter using the Bilinear method, the digital IIR filter transfer function $H(z)$ was derived in Chapter 9 and is repeated here as

$$H(z) = H(s)\Big|_{s=\frac{2}{T_s}\frac{z-1}{z+1}} \tag{10.8}$$

where as explained in Chapter 9

$$\theta_c = 2\tan^{-1}\frac{w_c T_s}{2} \tag{10.9}$$

The difference between $\theta_c = 2\tan^{-1} w_c T_s/2$ and $\theta_c = w_c T_s$ depends on the term $\tan^{-1} w_c T_s/2$. This difference is known as warping. If $w_c T_s/2$ is very small,

then $\tan^{-1} w_c T_s/2 = w_c T_s/2$ and thus $\theta_c = w_c T_s$. We can eliminate the effect of warping by prewarping. In prewarping the cut-off frequency of the analogue filter w_c is made equal to the cut-off frequency of the digital filter using the relation that was explained in Chapter 9

$$w_{\text{prewarp}} = \frac{2}{T_s}\tan\left(\frac{\theta_c}{2}\right) \tag{10.10}$$

with $\theta_c = w_c T_s$. With prewarping, we substitute w_{prewarp} for w_c so that the frequency responses before and after mapping match exactly at the frequency point f_{prewarp}. The matching point f_{prewarp} is specified in Hz. With the Bilinear transformation, the stability of the analogue continuous filter is preserved. This means that the poles of $H(s)$ that are in the left-half of the s-plane are mapped into the unit circle of the z-plane.

Example 10.2

Find the $H(z)$ that corresponds to $H(s) = \dfrac{3}{s+3}$, the transfer function discussed in Example 10.1. Use the Bilinear transform method with $T_s = .1$.

Solution

With $s = \dfrac{2}{T_s}\dfrac{z-1}{z+1} = \dfrac{2}{.1}\dfrac{z-1}{z+1} = 20\dfrac{z-1}{z+1}$,

$$H(z) = \frac{3}{20\dfrac{z-1}{z+1}+3} = \frac{3(z+1)}{20z-20+3z+3} = \frac{\dfrac{3}{23}(z+1)}{z-\dfrac{17}{23}} = \frac{0.1304z+0.1304}{z-0.7391}$$

The cut-off frequency of the analogue filter is $w_c = 3$ and the desired corresponding cut-off frequency of the digital filter is $\theta_c = w_c T_s = 0.3$. However, with the Bilinear transform, $\theta_c = 2\tan^{-1} w_c T_s/2 = 0.2978$. Thus, the amount of warping is 0.3 – 0.2978 = 0.002 or 2%. Had we used prewarping with $w_{\text{prewarp}} = 2/T_s \tan(\theta_c/2) = 2.978$ substituted for w_c, the digital transfer function would be

$$H(z) = \frac{3}{20\dfrac{z-1}{z+1}+2.978} = \frac{3(z+1)}{20z-20+2.978z+2.978}$$

$$= \frac{\dfrac{3}{22.978}(z+1)}{z-\dfrac{17}{22.978}} = \frac{0.1306z+0.1306}{z-0.7408}$$

Note that the location of the pole of $H(z)$ obtained using the Impulse Invariance method is at approximately the same location as the pole obtained using the Bilinear transform method.

With higher order filter the calculations get very complex. MATLAB contains the function `bilinear` and is used to obtain the transfer function $H(z)$ that approximates the analogue transfer function $H(s)$.

Within the syntax

```
[zz,pz,kz] = bilinear(zs,ps,ks,fs)
```

the function converts the s-domain transfer function specified by `zs, ps` and `ks` to a z-transform discrete equivalent obtained from the Bilinear transformation where column vectors `zs` and `ps` specify the zeros and poles, scalar `ks` specifies the gain and `fs` is the sampling frequency in Hz. With the syntax

```
[numz,denz] = bilinear(nums,dens,fs)
```

where `nums` and `dens` are row vectors containing the numerator and denominator coefficients of the analogue transfer function $H(s)$, in descending powers of s, the function transforms $H(s)$ to the z-transform coefficients `numz` and `denz` of the IIR digital filter transfer function $H(z)$. The syntax

```
[Az,Bz,Cz,Dz] = bilinear(As,Bs,Cs,Ds,fs)
```

is a state-space version of the bilinear transform.

Each syntax for the bilinear function accepts an optional additional input argument that specifies prewarping. The syntax

```
[zz,pz,kz] = bilinear(zs,ps,ks,fs,fprewarp)
```

applies prewarping before the bilinear transformation so that the frequency responses before and after mapping match exactly at frequency point `fprewarp`. The matching point `fprewarp` is specified in Hz.

We can now use MATLAB to find $H(z)$ with and without warping. With warping we use the script

```
Ts=0.1;
nums= [3];%numerator for H(s)
dens=[1 3];
[numz denz]=bilinear(nums,dens,1/0.1)
```

and the result is

numz = 0.1304 0.1304

denz = 1.0000 –0.7391

With prewarping we use the same script but with

```
[numz denz]=bilinear(nums,dens,1/0.1,2.978/(2*pi))
```

substituted for

```
[numz denz]=bilinear(nums,dens,1/0.1)
```

and the result is

numz = 0.1313 0.1313
denz = 1.0000 –0.7374

You can see that the difference between the two digital transfer functions is minimal due to the fact that $w_c T_s/2$ is very small.

10.3 IIR Filter Design Using MATLAB

We will use two methods to design IIR digital filters using MATLAB. In the first method, we will start with the analogue prototype lowpass filter, then we will convert this filter into any of the lowpass, highpass, bandpass or bandstop analogue filters with any cut-off frequency/frequencies. Next we will use either the Impulse Invariance or the Bilinear transformation methods to design the IIR digital desired filter. In the second method we will translate the digital requirements and use direct design methods available with MATLAB.

TABLE 10.1

Analogue Prototype Functions

Filter Type	Analogue Prototype Function
Butterworth	[zs,ps,ks] = buttap(n)
Chebyshev Type I	[zs,ps,ks] = cheb1ap(n,Rp)
Chebyshev Type II	[zs,ps,ks] = cheb2ap(n,Rs)
Elliptic	[zs,ps,ks] = ellipap(n,Rp,Rs)

10.3.1 From the Analogue Prototype to the IIR Digital Filter

We can use the design methods we learned in Chapter 8 to design any analogue filter of any type, then transform that filter to its IIR digital equivalent using the Impulse Invariance or the Bilinear transformations. The two transform MATLAB functions are `impinvar` and `bilinear`. The syntax for these functions was discussed earlier in this chapter. In this chapter we will

TABLE 10.2

Analogue Transformation Functions

Frequency Transformation	Transformation Function
Lowpass to lowpass	[nums,dens] = lp2lp(nums,dens,W0) [As,Bs,Cs,Ds] = lp2lp(As,Bs,Cs,Ds,W0)
Lowpass to highpass	[nums,dens] = lp2hp(nums,dens,W0) [As,Bs,Cs,Ds] = lp2hp(As,Bs,Cs,Ds,W0)
Lowpass to bandpass	[nums,dens] = lp2bp(nums,dens,W0,Bw) [As,Bs,Cs,Ds]=lp2bp(As,Bs,Cs,Ds,W0,Bw)
Lowpass to bandstop	[nums,dens] = lp2bs(nums,dens,W0,Bw) [As,Bs,Cs,Ds]=lp2bs(As,Bs,Cs,Ds,W0,Bw)

TABLE 10.3

Transformation from Analogue to Digital Filters

Frequency Transformation	Transformation Function
Bilinear	[zz,pz,kz] = bilinear(zs,ps,ks,fs) [zz,pz,kz] = bilinear(zs,ps,ks,fs,fprewarp) [numz, denz] = bilinear(nums,dens,fs) [numz, denz] = bilinear(nums,dens,fs,fprewarp)
Impulse Invariance	[numz,denz] = impinvar(nums,dens,fs) [numz,denz] = impinvar(nums,dens,fs,tol)

start with analogue prototype (1 rad/sec cut-off frequency) lowpass filters, then transform them to other analogue filters of any type and with any desired cut-off frequency/frequencies. Next, we will list the MATLAB functions needed without any further discussion, since they were explained in Chapter 8.

10.3.2 Direct Design

MATLAB has many functions that will help us design digital IIR filters directly. These MATLAB functions are listed in Table 10.4.

In Table 10.4, `n` is the order of the desired IIR digital filter. `wn` is the single-valued cut-off frequency of the lowpass filter. In case of a bandpass or a bandstop digital filter, `wn` is a vector of two values: the lower and the upper cut-off frequencies. `Rp` and `Rs` are the passband and the stopband attenuation ripples in decibels. `ftype` is the type of the desired filter. We use `stop` for stopband digital filters and `high` for highpass digital filters. If you do not specify the `ftype` the default is lowpass. The `wn` frequency/frequencies must be normalized to the range [0 1]. The range [0 1] corresponds to the range [0 π]. `zz, pz` and `kz` are the zeros, poles and gain constants.

We will not discuss Bessel filters in this chapter although they produce linear phase response. We need to use high-order Bessel to get good results. This is to say that Bessel filters have poor behavior and even Butterwoth

TABLE 10.4

Direct Design Matlab Functions for IIR Filters

Filter Type	Analogue Prototype Function
Butterworth	[numz, denz] = butter(n,wn,'ftype') [zz, pz, kz] = butter(n,wn,'ftype')
Chebyshev Type I	[numz, denz] = cheby1(n,Rp,wn,'ftype') [zz, pz, kz] = cheby1(n,Rp,wn,'ftype')
Chebyshev Type II	[numz, denz] = cheby2(n,Rs,wn,'ftype') [zz, pz, kz] = cheby2(n,Rs,wn,'ftype')
Elliptic	[numz, denz] = ellip(n,Rp,Rs,wn,'ftype') [zz, pz, kz] = ellip(n,Rs,Rp,wn,'ftype')

filters produce better results. One of the reasons we do not use FIR filters (finite impulse response filters are discussed in Chapter 11) is that they require high order. But they also produce linear phase and are always stable, unlike IIR filters. At the end of the chapter (EOCE), we will give many examples to illustrate the procedure of designing IIR digital filters.

10.4 Some Insights

10.4.1 The Difficulty in Designing IIR Digital Filters in the *z*-Domain

Designing an irrational IIR digital filter is very difficult and its implementation is very costly. The IIR digital filters we will consider are the IIR filters that are rational, proper, causal and stable. The transfer function representing such digital IIR filters is given as

$$H(z) = \frac{N(z)}{D(z)} = \frac{b_N z^N + b_{N-1} z^{N-1} + b_{N-2} z^{N-2} + \ldots + b_0}{a_N z^N + a_{N-1} z^{N-1} + a_{N-2} z^{N-2} + \ldots + a_0} \tag{10.11}$$

where a_N must not be zero for the filter to be proper and causal. To design such a filter we can either find the constants a_0 through a_N and b_0 through b_N or use a much easier method to find

$$\left|H\left(e^{j\theta}\right)\right|^2 = H\left(e^{j\theta}\right)H^*\left(e^{j\theta}\right) = H\left(e^{j\theta}\right)H\left(e^{-j\theta}\right) = H(z)H\left(z^{-1}\right)\Big|_{z=e^{j\theta}} \tag{10.12}$$

It can be easy to find $|H(e^{j\theta})|^2$ but our goal is to find a rational, proper and stable $H(z)$. Note that not every $|H(e^{j\theta})|^2$ can be factored out as in Equation (10.12) and therefore the search for a rational, proper and stable $H(z)$ is very difficult. However, if we are after a rational, proper and stable analogue transfer function $H(s)$ as in

$$H(s)=\frac{N(s)}{D(s)}=\frac{b_N s^N+b_{N-1}s^{N-1}+b_{N-2}s^{N-2}+\ldots+b_0}{a_N s^N+a_{N-1}s^{N-1}+a_{N-2}s^{N-2}+\ldots+a_0} \tag{10.13}$$

we can see that finding

$$\begin{aligned}\left|H(jw)\right|^2 &= H(jw)H^*(jw)=H(jw)H(-jw)=H(s)H(-s)\big|_{s=jw}\\ &=\frac{N(s)N(-s)}{D(s)D(-s)}\Big|_{s=jw}\end{aligned} \tag{10.14}$$

is very simple if $|H(jw)|^2$ is a proper rational function of w^2. Consider the magnitude-squared response for the analogue transfer function of a Butterworth prototype lowpass filter

$$\left|H(jw)\right|^2=\frac{1}{1+w^{2N}} \tag{10.15}$$

Let $N = 2$ in Equation (10.15). With $(jw)(jw) = -w^2 = s^2$, we can write Equation (10.5) as

$$\left|H(jw)\right|^2=\frac{1}{1+w^{2(2)}}=\frac{1}{1+\left(-s^2\right)^2}=H(s)H(-s) \tag{10.16}$$

We can factor $\dfrac{1}{1+(-s^2)^2}$ by finding the roots of $1+(-s^2)^2 = 0$ or $(-1)^2\, s^{2(2)} = -1$.

Simplifying further we arrive at

$$s^{2(2)}=-1(-1)^2=(-1)^{2+1}=e^{j((2+1)\pi+2\pi n)} \qquad n=0,1,\ldots,2(2)-1 \tag{10.17}$$

The four roots are at

$$s_n=e^{j((2+1)\pi+2\pi n)/2(2)} \qquad n=0,1,\ldots,2(2)-1 \tag{10.18}$$

We can use MATLAB to find these roots. At the MATLAB prompt we type:

```
n=0:3;%four roots
s=exp(j*(3*pi+2*pi*n)/4)%all the roots (4 of them)
%We observe the roots in the left-half s-plane and group them
s=[s(1) s(2)];
den=poly(s) %get the denominator coefficients
```

The desired $H(s)$ is therefore

$$H(s) = \frac{1}{\left[s-(-0.7071+0.7071j)\right]\left[s-(-0.7071-0.7071j)\right]} = \frac{1}{s^2+1.4142s+1} \tag{10.19}$$

You can see the symmetry in the roots. Half of the roots are in the left-half s-plane. These are the roots that will be considered. With this simple example you can see the simplicity of factorizing $|H(e^{j\theta})|^2$ to obtain the rational, proper and stable $H(s)$.

10.4.2 Using the Impulse Invariance Method

Only if $H(s)$ is strictly proper (the degree of the numerator is less than the degree of the denominator) will the Impulse Invariance method give you correct results.

10.4.3 The Choice of the Sampling Interval T_s

If the sampling period T_s is according to the relation

$$\frac{\pi}{T_s} = \frac{1}{2}|b|$$

where b is the imaginary part of the complex pole $a \pm jb$ of the transfer function $H(s)$, the transformation to $H(z)$ may result in loss of some dynamics of the system. The reason is that the mapping from the s-plane to the z-plane is not one-to-one.

10.5 End of Chapter Examples

EOCE 10.1

We are trying to eliminate all frequencies that are higher than 100 Hz from the incoming signal $x(t)$. If the sampling frequency is 1000 Hz, design a digital lowpass IIR filter to accomplish this task.

Solution

The cut-off frequency, w_c, of the equivalent analogue lowpass filter needed is 200π rad/sec. We will carry out the design using two methods and with

each method we will use the bilinear and the Impulse Invariance transformations. We will design second- and sixth-order Butterworth filters. For filters of order 2 or higher the complexity of the mathematics is high. Thus we will use MATLAB in all the design problems.

First method: from the analogue prototype to the IIR digital filter

We will write the following MATLAB script to accomplish the design using this method. We will start with a prototype second-order analogue lowpass Butterworth filter, then convert it to a second-order lowpass analogue Butterworth filter with cut-off frequency of 200π rad/sec. Then we will use the Bilinear and the Impulse Invariance transformations to get $H(z)$, the transfer function of the IIR digital filter.

```
%we will generate the lowpass analogue Butterworth prototypes
 first
for n=2:4:6% N=2 and 6
[zs, ps, ks]=buttap(n);
% put in terms of transfer function
[nums, dens]=zp2tf(zs,ps,ks);
%next we transform to analogue lowpass with cut-off 200pi
[nums, dens]=lp2lp(nums, dens, 200*pi);
%Then we use the Bilinear transform first
fs=1000;
[numzb, denzb]=bilinear(nums, dens, fs)
%then we use the Impulse Invariance method
[numzi, denzi]=impinvar(nums, dens, fs)
%Now we plot the magnitude response of the IIR digital filter
%using the two transformation methods
[Hb, fb]=freqz(numzb, denzb);
[Hi, fi]=freqz(numzi, denzi);
subplot(2,1,1);
plot(fb/pi, abs(Hb));
hold on;
axis([0 1 0 1]);
title('The Magnitude response using the Bilinear transform');
subplot(2,1,2);
plot(fi/pi, abs(Hi));
hold on;
end
title('The Magnitude response using the Invariance transform');
xlabel('frequency in pi units');
axis([0 1 0 1]);
```

The plots are shown in Figure 10.1. You can see that the Bilinear transformation is superior to the Impulse Invariance transformation for the low order

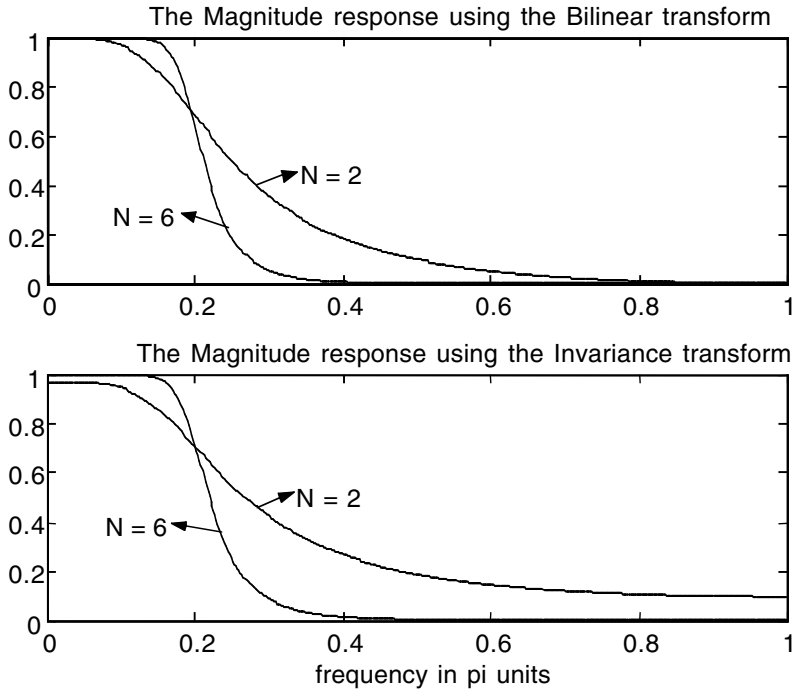

FIGURE 10.1 Plots for EOCE 10.1: Indirect Method.

of 2. The two methods are somehow identical when the order becomes 6. Using the Bilinear transformation method, the numerator and denominator coefficients of the transfer function of the IIR digital second-order filter are

numzb = 0.0640 0.1279 0.0640
denzb = 1.0000 –1.1683 0.4241

and the corresponding IIR filter transfer function is

$$H(z) = \frac{0.0640z^2 + 0.1279z + 0.0640}{z^2 - 1.1683z + 0.4241}$$

For the sixth-order filter we have

numzb = 0.0003 0.0017 0.0043 0.0058 0.0043 0.0017 0.0003
denzb = 1.0000 –3.6543 5.8693 –5.2192 2.6894 –0.7574 0.0908

and the corresponding IIR filter transfer function is

$$H(z) = \frac{0.0003z^6 + 0.0017z^5 + 0.0043z^4 + 0.0058z^3 + 0.0043z^2 + 0.0017z + 0.0003}{z^6 - 3.6543z^5 + 5.8693z^4 - 5.2192z^3 + 2.6894z^2 - 0.7574z + 0.0908}$$

Using the Impulse Invariance method, the numerator and denominator coefficients of the transfer function of the IIR digital second-order filter are

numzi = 0 0.2449
denzi = 1.0000 –1.1580 0.4112

and the corresponding transfer function of the IIR filter is

$$H(z) = \frac{0.2449}{z^2 - 1.1580z + 0.4112}$$

For the sixth-order filter we have

numzi = 0.0000 0.0003 0.0057 0.0096 0.0026 0.0001
denzi = 1.0000 –3.6300 5.7952 –5.1282 2.6316 –0.7386 0.0882

and the corresponding transfer function of the IIR filter is

$$H(z) = \frac{0.0003z^4 + 0.0057z^3 + 0.0096z^2 + 0.0026z + 0.0001}{z^6 - 3.63z^5 + 5.7952z^4 - 5.1282z^3 + 2.6316z^2 - 0.7386z + 0.0882}$$

Second method: Direct Design

Using this method we will use the MATLAB function

```
[numz, denz] = butter(n,wn,'ftype')
```

without `ftype`, since the default type is lowpass, and with `wn` = 0.2 and `n` = 2 and 6. We used `wn` = 0.2 as the normalized frequency because the equivalent cut-off digital frequency is 200π (1/1000) = 0.2π. Using MATLAB, the digital frequencies should be normalized to belong to the digital frequency interval [0 1].

The following MATLAB script will accomplish the design:

```
clf
for n=2:4:6;
[numz, denz]=butter(n, 0.2)%the normalized digital frequency
 is 0.2
%Now we plot the magnitude response of the IIR digital filter
%using the two transformation methods
[H, f]=freqz(numz, denz);
plot(f/pi, abs(H));hold on;
end
title('The Magnitude response using the function butter');
xlabel('frequency in pi units');
axis([0 1 0 1]);
```

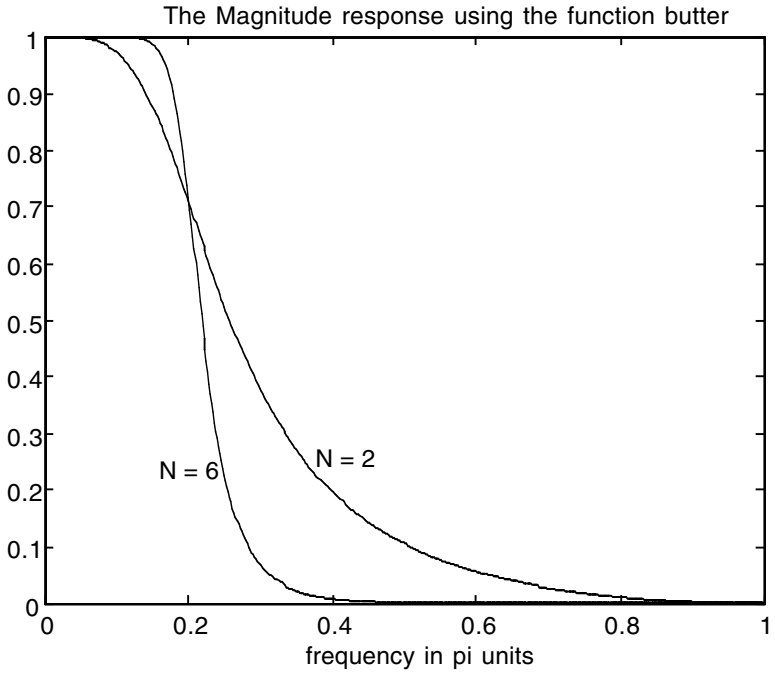

FIGURE 10.2 Plots for EOCE 10.1: Direct Method.

The plots are shown in Figure 10.2. Note the sharpness in the roll-off frequency at 0.2π. In this case, coefficients of the numerator and the denominator of the transfer function for $N = 2$ are

numz = 0.0675 0.1349 0.0675
denz = 1.0000 –1.1430 0.4128

and for $N = 6$ they are

numz = 0.0003 0.0020 0.0051 0.0068 0.0051 0.0020 0.0003
denz = 1.0000 –3.5794 5.6587 –4.9654 2.5295 –0.7053 0.0838

The corresponding digital IIR transfer functions are

$$H(z) = \frac{0.0657z^2 + 0.1349z + 0.0657}{z^2 - 1.1430z + 0.4128}$$

and

$$H(z) = \frac{0.0003z^6 + 0.0020z^5 + 0.0051z^4 + 0.0068z^3 + 0.0051z^2 + 0.0020z + 0.0003}{z^6 - 3.5794z^5 + 5.6587z^4 - 4.9654z^3 + 2.5295z^2 - 0.7053z + 0.0838}$$

It seems that the result of using the `butter` MATLAB function is similar to the result using the Bilinear transformation method.

EOCE 10.2

Consider the input continuous signal

$$x(t) = 2 + \cos(10t)$$

Design a digital second- and sixth-order IIR lowpass Butterworth filter to attenuate the cos(10*t*) term and pass the dc 2 term.

Solution

We will use the MATLAB direct design method using the `butter` function. The highest frequency in the input signal is $f_m = 10/2\pi = 1.59$ Hz. The sampling frequency should be at least $f_s = 2(1.59) = 3.18$ Hz, and this sampling frequency corresponds to the sampling period $T_s = 0.31$ sec. Let us take $T_s = 0.1$ sec. This value still prevents aliasing. The analogue frequency that we are trying to suppress is $w = 10$ rad/sec. An analogue lowpass filter with $w_c = 5$ rad/sec will suppress the cos(10*t*) term. The analogue cut-off frequency of 5 rad/sec corresponds to the digital cut-off frequency $\theta_c = T_s(w_c) = 0.1(5) = 0.5$ rads per sample. When we use the MATLAB `butter` function, we must normalize the cut-off frequency to become $0.5/\pi = 0.1592$. Thus, we will use 0.1592 with the MATLAB function `butter`. The following MATLAB script will carry out the design and plot the input signal before and after passing through the digital IIR lowpass filter.

```
clf
t=0:.01:5;
xt=cos(10*t)+2;%This is the unsampled input to the IIR digital
 filter
n=0:50;
Ts=.1;
xn1=2;%This signal should pass through the filter
xn2=cos(10*n*Ts);%This signal should not pass
xn=xn1+xn2;
subplot(3,1,1); plot(t, xt);
title('The input signal:cos(10*t)+2 before filtering');
for N=2:4:6%digital filter with orders 2 and 6
[numz denz]=butter(N, 0.1592)%The transfer function of the
 digital filter
[H, f]=freqz(numz, denz);
subplot(3,1,2); plot(f/pi, abs(H)); hold on;
```

```
title('The magnitude response of the IIR digital lowpass
 filters');
yn=filter(numz,denz,xn);
subplot(3,1,3); plot(n*Ts,yn); hold on;
end
title('The signal after cos(10*t) term is removed');
xlabel('Time(sec)');
```

The plots are shown in Figure 10.3. The coefficients of the numerator and denominator of the transfer function of the IIR digital filter for $N = 2$ are

numz = 0.0457 0.0915 0.0457
denz = 1.0000 –1.3106 0.4935

with the corresponding transfer function

$$H(z) = \frac{0.0457z^2 + 0.0915z + 0.0457}{z^2 - 1.3106z + 0.4935}$$

and for $N = 6$ are

numz = 0.0001 0.0006 0.0016 0.0021 0.0016 0.0006 0.0001
denz = 1.0000 –4.0713 7.1304 –6.8262 3.7512 –1.1186 0.1411

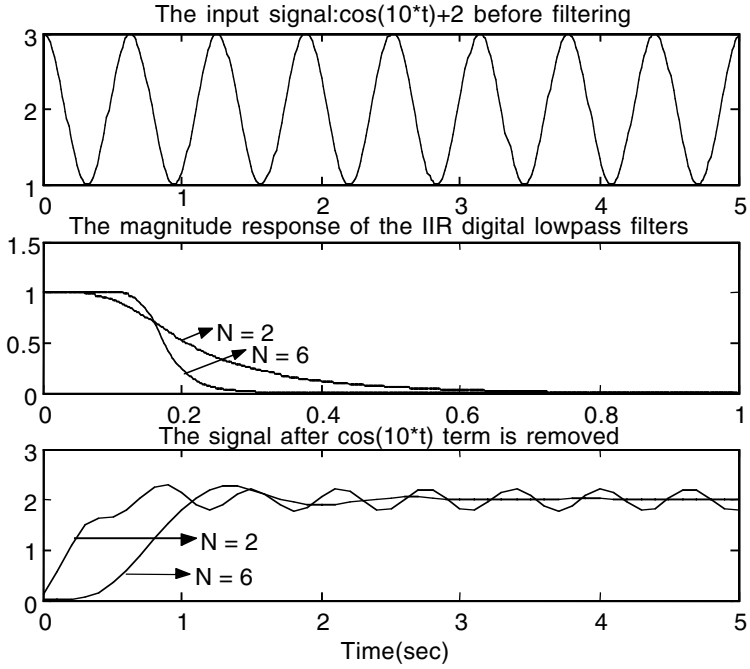

FIGURE 10.3 Plots for EOCE 10.2.

with the corresponding transfer function

$$H(z) = \frac{0.0001z^6 + 0.0006z^5 + 0.0016z^4 + 0.0021z^3 + 0.0016z^2 + 0.0006z + 0.0001}{z^6 - 4.0713z^5 + 7.1304z^4 - 6.8262z^3 + 3.7512z^2 - 1.1186z + 0.1411}$$

You can see clearly in the plots that the filter is doing well for $N = 6$. Only the dc component is present after about 2 sec when the system reaches its steady state.

EOCE 10.3

We are trying to eliminate all frequencies that are lower than 100 Hz from the incoming signal $x(t)$. If the sampling frequency is 1000 Hz, design a digital highpass IIR filter to accomplish this task. Allow 1 dB for the maximum passband attenuation and 40 dB for the minimum stopband attenuation.

Solution

The cut-off frequency, w_c, of the equivalent analogue highpass filter needed is 200π rad/sec. We will carry out the design using two methods and with each method we will use the bilinear and the impulse invariance transformations. We will design second- and sixth-order elliptical filters. For filters of order 2 or higher the complexity of the mathematics is high. Thus we will use MATLAB in all the design problems.

First method: from the analogue prototype to the IIR digital filter

We will write the following MATLAB script to accomplish the design using this method. We will start with a prototype second-order analogue lowpass elliptical filter, then convert it to a second-order highpass analogue elliptical filter with cut-off frequency of 200π rad/sec. Then we will use the Bilinear and the Impulse Invariance transformations to get $H(z)$, the transfer function of the IIR digital highpass filter.

```
%we will generate the lowpass analogue elliptical prototypes
 first
for n=2:4:6 % N=2 and 6
Rp = 1; Rs = 40;
[zs, ps, ks]=ellipap(n, Rp, Rs);
% put in terms of transfer function
[nums, dens]=zp2tf(zs,ps,ks);
%next we transform to analogue highpass with cut-off 200pi
[nums, dens]=lp2hp(nums, dens, 200*pi);
%Then we use the Bilinear transform first
fs=1000;
```

```
[numzb, denzb]=bilinear(nums, dens, fs)
%then we use the Impulse Invariance method
[numzi, denzi]=impinvar(nums, dens, fs)
%Now we plot the magnitude response of the IIR digital filter
%using the two transformation methods
[Hb, fb]=freqz(numzb, denzb);
[Hi, fi]=freqz(numzi, denzi);
subplot(2,1,1);
plot(fb/pi, abs(Hb));
hold on;
title('The Magnitude response using the Bilinear transform');
subplot(2,1,2);
plot(fi/pi, abs(Hi));
hold on;
end
title('The Magnitude response using the Invariance transform');
xlabel('frequency in pi units');
```

The plots are shown in Figure 10.4. You can see that the Bilinear transformation is superior to the Impulse Invariance transformation. The Impulse Invariance method fails with this filter design as you can see from the plot

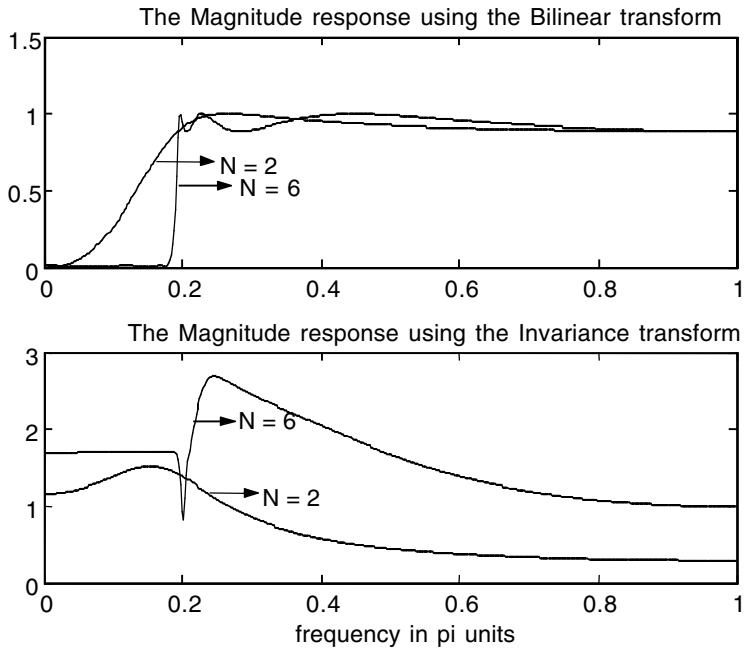

FIGURE 10.4 Plots for EOCE 10.3: Indirect Method.

in Figure 10.4. The reason is that, to apply the Impulse Invariance transformation, the transfer function to be transformed, $H(s)$, should be strictly proper, as we saw in Chapter 9 when we discussed the Impulse Invariance method. In our case (as obtained from the above script at line 8 by removing the semicolon from the end of the MATLAB statement; you may need to type `format long` at the MATLAB prompt before running the script), the analogue transfer function of the highpass filter for order 2 is

$$H(s) = \frac{s^2}{s^2 + 888.5765s + 394784.176}$$

which is clearly a proper transfer function and that is why we did not get the correct results using the Impulse Invariance method. Using the Bilinear transformation method, the numerator and denominator coefficients of the transfer function of the highpass IIR digital second-order filter are

```
numzb = 0.6379  –1.2733  0.6379
denzb = 1.0000  –1.3027  0.5574
```

and the corresponding IIR filter transfer function is

$$H(z) = \frac{0.6379z^2 - 1.2733z + 0.6379}{z^2 - 1.3027z + 0.5574}$$

For the sixth-order filter we have

```
numzb = 0.3586  –1.9515  4.6100  –6.0331  4.6100  –1.9515  0.3586
denzb = 1.0000  –3.6468  6.2536  –6.1380  3.7022  –1.3129  0.2446
```

and the corresponding IIR filter transfer function is

$$H(z) = \frac{0.3586z^6 - 1.9515z^5 + 4.6100z^4 - 6.0331z^3 + 4.6100z^2 - 1.9515z + 0.3586}{z^6 - 3.6468z^5 + 6.2536z^4 - 6.1380z^3 + 3.7002 - 1.3129z + 0.2446}$$

Second method: direct design

Using this method we will use the MATLAB function

```
[numz, denz] = ellip(n,Rp,Rs,wn,'ftype')
```

with `ftype` as `high`, `wn` = 0.2 and `n` = 2 and 6. We used `wn` = 0.2 as the normalized frequency because the equivalent cut-off digital frequency of the highpass filter is 200π (1/1000) = 0.2π Using MATLAB, the digital frequencies should be normalized to belong to the digital frequency interval [0 1]. The following MATLAB script will accomplish the design:

```
Rp=1; Rs=40; wn=0.2;
for n=2:4:6;
[numz, denz] = ellip(n,Rp,Rs,wn,'high')%the normalized
 digital frequency is 0.2
%Now we plot the magnitude response of the IIR digital filter
[H, f]=freqz(numz, denz);
plot(f/pi, abs(H));hold on;
end
title('The Magnitude response using the function ellip');
xlabel('frequency in pi units');
axis([0 1 0 1.1]);
```

The plots are shown in Figure 10.5. Note that as the order of the highpass IIR elliptical filter increases, the ripples get excessive. In this case, coefficients of the numerator and the denominator of the transfer function for $N = 2$ are

numz = 0.6304 –1.2581 0.6304
denz = 1.0000 –1.2785 0.5477

and for $N = 6$ are

numz = 0.3477 –1.8799 4.4243 –5.7831 4.4243 –1.8799 0.3477
denz = 1.0000 –3.5576 6.0199 –5.8453 3.5107 –1.2446 0.2377

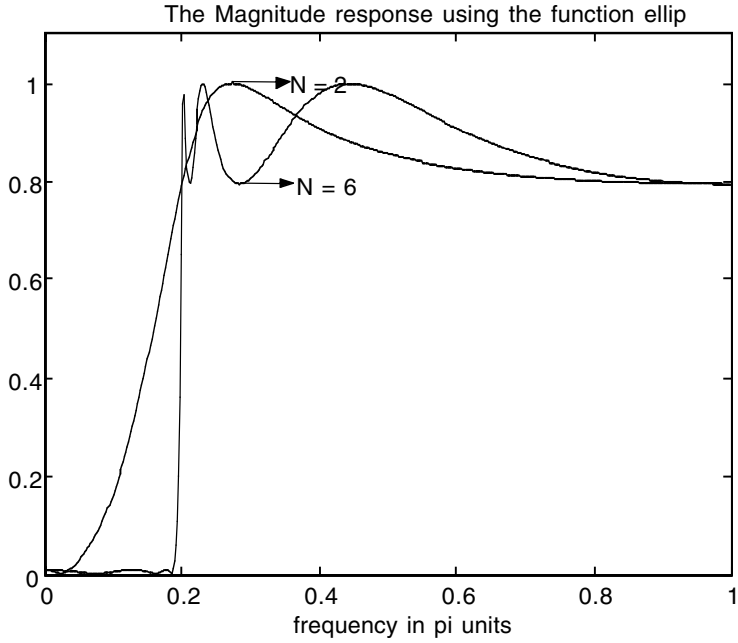

FIGURE 10.5 Plots for EOCE 10.3: Direct Method.

The corresponding digital IIR transfer functions are

$$H(z) = \frac{0.6304z^2 - 1.2581z + 0.6304}{z^2 - 1.2785z + 0.5477}$$

and

$$H(z) = \frac{0.3477z^6 - 1.8799z^5 + 4.4243z^4 - 5.7831z^3 + 4.4243z^2 - 1.8799z + 0.3477}{z^6 - 3.5576z^5 + 6.0199z^4 - 5.8453z^3 + 3.5107z^2 - 1.2446z + 0.2377}$$

EOCE 10.4

Consider the input continuous signal

$$x(t) = 2 + \cos(10t)$$

Design digital second- and sixth-order IIR highpass elliptical filters to attenuate the 2 term and pass the cos(10t) term.

Solution

We will use the MATLAB direct design method using the `ellip` function. The highest frequency in the input signal is $f_m = 10/2\pi = 1.59$ Hz. The sampling frequency should be at least $f_s = 2(1.59) = 3.18$ Hz. This sampling frequency corresponds to the sampling period $T_s = 0.31$ sec. Let us take $T_s = 0.1$ sec. This value still prevents aliasing. The analogue frequency that we are trying to suppress is $w = 0$ rad/sec. This corresponds to the dc component of magnitude 2. An analogue highpass filter with $w_c = 5$ rad/sec will suppress this dc term of 2. The analogue cut-off frequency of 5 rad/sec corresponds to the digital cut-off frequency $\theta_c = T_s(w_c) = 0.1(5) = 0.5$ rads per sample. When we use the MATLAB `ellip` function we must normalize the cut-off frequency to become $0.5/\pi = 0.1592$. Thus we will use = 0.1592 with the MATLAB function `ellip`. The following MATLAB script will carryout the design and plot the input signal before and after passing through the digital IIR highpass filter. We will allow maximum passband ripple of 1 dB and minimum stopband attenuation of 40 dB.

```
clf
t=0:.01:4;
xt=cos(10*t)+2;%This is the unsampled input to the IIR digital
 filter
n=0:40;
Ts=.1;
xn1=2;%This signal should pass through the filter
xn2=cos(10*n*Ts);%This signal should not pass
xn=xn1+xn2;
```

```
subplot(3,1,1); plot(t, xt);title('The input
 signal:cos(10*t)+2 before filtering');
for N=2:4:6%digital filter with orders 2 and 6
[numz denz]=ellip(N,0.1, 40,0.1592,'high')%The transfer
 function of the digital filter
[H, f]=freqz(numz, denz);
subplot(3,1,2); plot(f/pi, abs(H)); hold on;
title('The magnitude response of the IIR digital highpass
 filters');
yn=filter(numz,denz,xn);
subplot(3,1,3); plot(n*Ts,yn); hold on;
end
title('The signal after the constant 2 term is removed');
xlabel('Time(sec)');
```

The plots are shown in Figure 10.6. You can easily see that both filters pass the cos(10*t*) term and block the dc term. With $N = 2$ and $N = 6$, both filters worked nicely with little delay with the $N = 6$ filter. The numerator and denominator coefficients for the $N = 2$ elliptical IIR highpass filter are

numz = 0.8231 –1.6455 0.8231

denz = 1.0000 –1.6321 0.6976

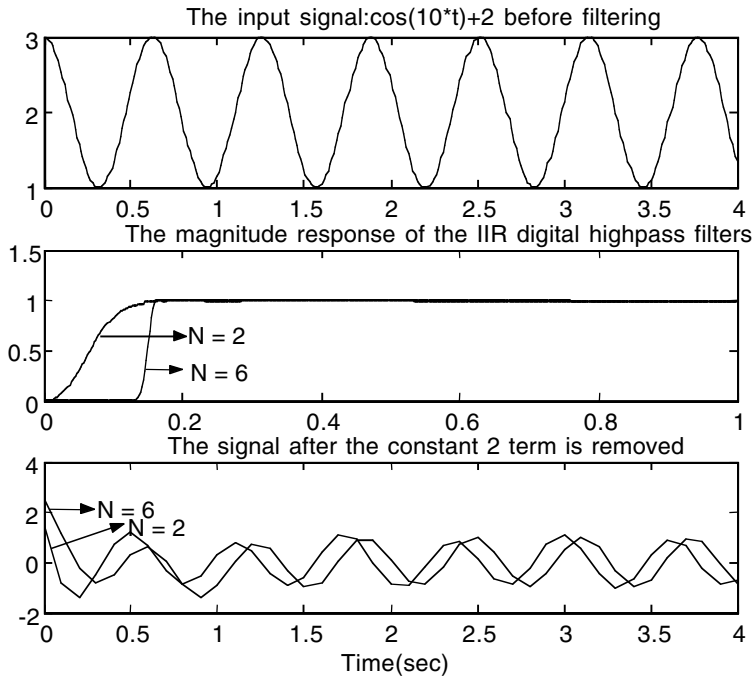

FIGURE 10.6 Plots for EOCE 10.4.

with the corresponding transfer function as

$$H(z) = \frac{0.8231z^2 - 1.6455z + 0.8231}{z^2 - 1.6321z + 0.6976}$$

and with $N = 6$ we get

```
numz = 0.4716  –2.6891  6.5233  –8.6114  6.5233  –2.6891  0.4716
denz = 1.0000  –4.2625  7.9660  –8.2373  4.9564  –1.6432  0.2380
```

with the corresponding transfer function

$$H(z) = \frac{0.4716z^6 - 2.6891z^5 + 6.5233z^4 - 8.6114z^3 + 6.5233z^2 - 2.6891z + 0.4716}{z^6 - 4.2625z^5 + 7.9660z^4 - 8.2373z^3 + 4.9564z^2 - 1.6432z + 0.2380}$$

EOCE 10.5

We are trying to pass all frequencies that are higher than 200 Hz and below 400 Hz from the incoming signal $x(t)$. If the sampling frequency is 1000 Hz (the minimum sampling should be 800 Hz but sampling at higher rates can give better results), design a digital bandpass IIR filter to accomplish this task. Use the Cheby1 filter in the design. Use 2 dB for the maximum allowable ripple in the passband.

Solution

The analogue lower and upper cut-off frequencies are at 400π and 800π rad/sec, respectively. We will carry out the design using two methods and with each method we will use the Bilinear and the Impulse Invariance transformations. We will design a fourth-order Cheby1 filter first.

First method: from the analogue prototype to the IIR digital filter

We will write the following MATLAB script to accomplish the design using this method. We will start with a prototype second-order analogue lowpass Cheby1 filter, then convert it to a bandpass analogue Cheby1 filter with order 4 and with the cut-off frequencies at 400π for w_{cl} and 800π for w_{cu}. Then we will use the Bilinear and the Impulse Invariance transformations to get $H(z)$, the transfer function of the IIR digital filter. We will be using the MATLAB function call

```
[zs, ps, ks]=cheb1ap(N, Rp);
```

with $N = 2$ to get the second-order Cheby1 prototype lowpass filter and the MATLAB transformation

```
[nums,dens] = lp2bp(nums,dens,W0,Bw)
```

The value of W0 will be calculated as $\sqrt{(400\pi)(800\pi)} = 1777$ and the value of Bw is $800\pi - 400\pi = 400\pi$.

```
clf
%we will generate the lowpass analogue Cheby1 prototypes first
Rp = 2; N=2;%filter order, the actual order is 4
i=1;%index for the plots
%next are labels to be displayed later
frequency{1}='1000'; frequency{2}='4000'; frequency{3}='7000';
for fs=1000:3000:7000;
[zs, ps, ks]=cheb1ap(N, Rp);
% put in terms of transfer function
[nums, dens]=zp2tf(zs,ps,ks);
%next we transform to analogue bandpass with cut-off [400pi
 800pi]
[nums, dens]=lp2bp(nums, dens, 1777, 400*pi);
%Then we use the bilinear transform first
[numzb, denzb]=Bilinear(nums, dens, fs)
%then we use the Impulse Invariance method
[numzi, denzi]=impinvar(nums, dens, fs)
%Now we plot the magnitude response of the IIR digital filter
%using the two transformation methods
[Hb, fb]=freqz(numzb, denzb,200);
[Hi, fi]=freqz(numzi, denzi,200);
subplot(3,1,i);
plot(fb/pi, abs(Hb));
hold on;
plot(fi/pi, abs(Hi),'*');
title(['Magnitude Response with N= 4 and fs = 'frequency{i}]);
%preparing for the legend
a=['Bilinear transform'];
b=['Impulse Invariance transform'];
legend(a,b);
i=i+1
end
xlabel('frequency in pi units');
```

The plots are shown in Figure 10.7. You can see that both the Bilinear and the Impulse Invariance transformations for `fs` = 1000 Hz did not give good results. We desire that the normalized digital bandpass for `fs` = 1000 be in the interval [400π/1000π 800π/1000π]. This is not close to the results we obtained for both transforms with `fs` = 1000 Hz. When we increased the sampling frequency, the results are the same and correct for both transforms.

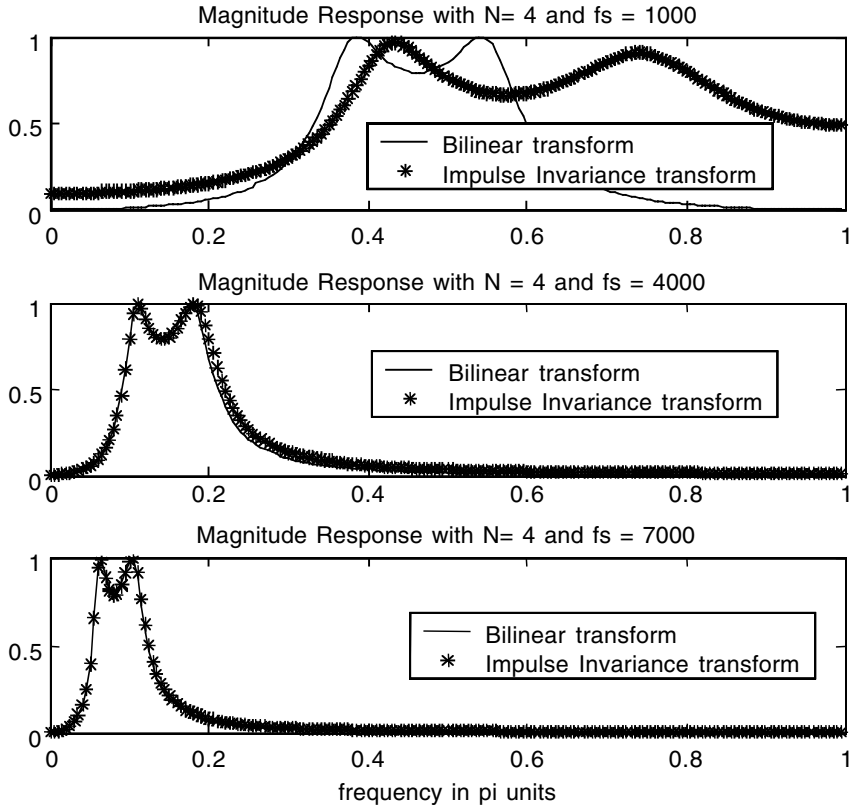

FIGURE 10.7 Plots for EOCE 10.5: Indirect Method 1.

Note that when we change the sampling frequency, the digital bandpass changes too. For example, when fs = 4000 Hz, the band becomes [400π/4000π 800π/4000π]. Thus, for order four, we will use either of the designs with `fs` = 4000 Hz to get good results and a good filter. The numerator and denominator coefficients of the transfer function of the IIR bandpass digital fourth-order filter are obtained from the result of the above script. For `fs` = 4000 Hz, the digital transfer function is obtained for the bilinear transform as

numzb = 0.0129 –0.0000 –0.0257 –0.0000 0.0129
denzb = 1.0000 –3.3737 4.6069 –2.9908 0.7887

For the fourth-order bandpass filter with fs = 4000 Hz and using the Impulse Invariance method we have the coefficients

numzi = 0.0000 0.0525 –0.1054 0.0527
denzi = 1.0000 –3.3446 4.5422 –2.9413 0.7768

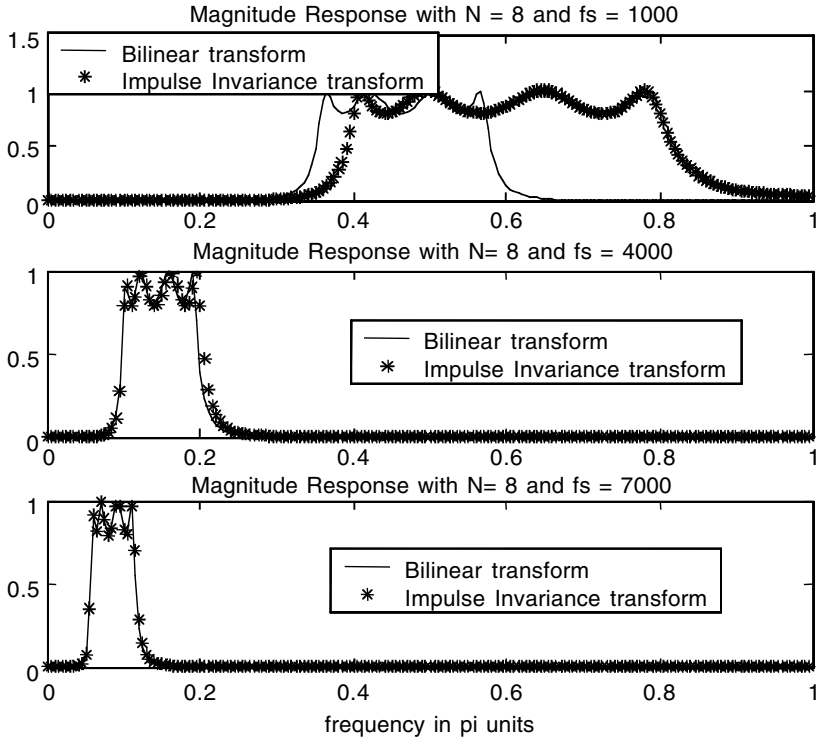

FIGURE 10.8 Plots for EOCE 10.5: Indirect Method 2.

The transfer function of the IIR bandpass digital using the Bilinear transform is

$$H(z) = \frac{0.0129z^4 - 0.0z^3 - 0.0257z^2 - 0.0z + 0.0129}{z^4 - 3.3737z^3 + 4.6069z^2 - 2.9908z + 0.7887}$$

and with the Impulse Invariance method it is

$$H(z) = \frac{0.0525z^2 - 0.1054z + 0.0527}{z^4 - 3.3446z^3 + 4.5422z^2 - 2.9413z + 0.7768}$$

When we increase the order of the filter to 8 we will use $N = 4$ in the script above. We will continue to use the sampling frequencies 1000, 4000 and 7000. The plots are shown in Figure 10.8. With `fs` = 1000 Hz we see that the Impulse Invariance method gives a good approximation. The coefficients of the numerator and the denominator of the digital IIR bandpass filter with order eight ($N = 4$ in the script for bandpass or bandstop filters) and `fs` = 1000 Hz using the Impulse Invariance method are

numzi = 0.0000 0.0222 –0.0398 –0.0209 0.1061 –0.0529 –0.0153 0.0154

denzi = 1.0000 1.5751 2.9703 3.2812 3.6898 2.7001 2.0365 0.8565 0.4066

If we want to use the Bilinear transform method we have to use the coefficients that are obtained with `fs` = 4000. This is evident from the second picture in Figure 10.8. In this case, the coefficients using the bilinear transform are

numzb = 0.0000722 0 –0.0002887 0 0.0004330 0 –0.0002887 0 0.0000722

denzb = 1.0000 –6.9782 22.0341 –41.0188 49.1861 –38.8887 19.8060 –5.9477 0.8084

Second method: direct design

Using this method we will use the MATLAB function

```
[numz, denz] = cheby1(n,Rp,wn,'ftype')
```

with `ftype` as `bandpass`, `wn` = [0.4 0.8] using `fs` = 1000 and `n` = 2 and 4 to correspond to the fourth- and eighth-order bandpass IIR digital filters.

We used `wn` = [0.4 0.8] as the normalized frequency because the equivalent cut-off digital frequencies are at 400π (1/1000) = 0.4π and at 800π (1/1000) = 0.8π. Using MATLAB, the digital frequencies should be normalized to belong to the digital frequency interval [0 1]. The following MATLAB script will accomplish the design.

```
clf
Rp=2;
%sampling frequency is 1000 Hz
wn=[0.4 0.8];%normalized to fit in [0 1]
for n=2:2:4;
[numz, denz] = cheby1(n,Rp,wn,'bandpass')%the normalized
 digital frequency is 0.2
%Now we plot the magnitude response of the IIR digital filter
[H, f]=freqz(numz, denz);
plot(f/pi, abs(H));hold on;
end
title('The Magnitude response using the function cheby1');
xlabel('frequency in pi units');
axis([0 1 0 1.1]);
```

The plots are shown in Figure 10.9. It is evident that the MATLAB function `cheby1` gave good results for the eighth-order filter and not-so-good results for the fourth-order filter with `fs` = 1000 Hz. The coefficients for the numerator and the denominator of the bandpass IIR digital filter for order four using the `cheby1` MATLAB function are

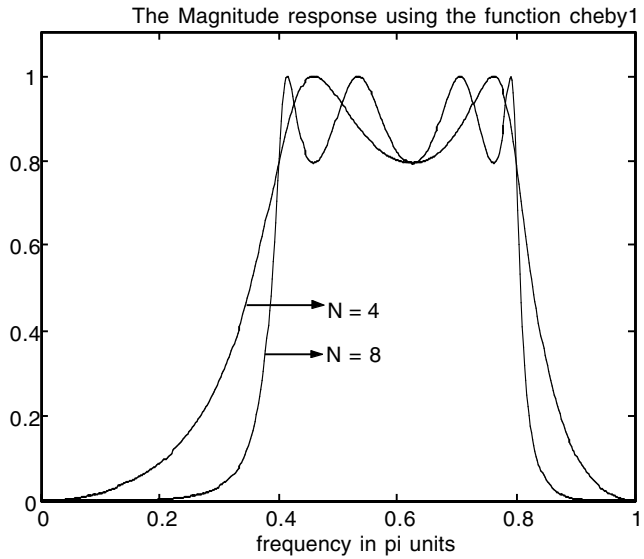

FIGURE 10.9 Plots for EOCE 10.5: Direct Method.

```
numz = 0.1710      0  –0.3419      0  0.1710
denz = 1.0000  0.9780  0.8495  0.5359  0.4213
```

and those with order eight are

```
numz = 0.0187      0  –0.0747      0  0.1120      0  –0.0747      0  0.0187
denz = 1.0000  2.2192  3.7778  4.4718  4.7505  3.6762  2.5025  1.1172  0.4108
```

If we run the script in the first method with fs as 3000, 4000 and 5000 Hz, we will obtain the plots in Figure 10.10. You can see in Figure 10.10 that with $N = 4$ and fs = 3000 Hz we have reasonable results using both the Bilinear and the Impulse Invariance transformation. The transfer functions of the IIR bandpass filter for this case can still be obtained from the same script.

EOCE 10.6

We are interested in passing the term $\sin(1500t)$ from the input signal

$$x(t) = \sin(10t) + \sin(1500t) + \sin(5000t)$$

Design a bandpass IIR Cheby1 digital filter to accomplish this task. Plot the input signal along with the magnitude response of the filter and its output.

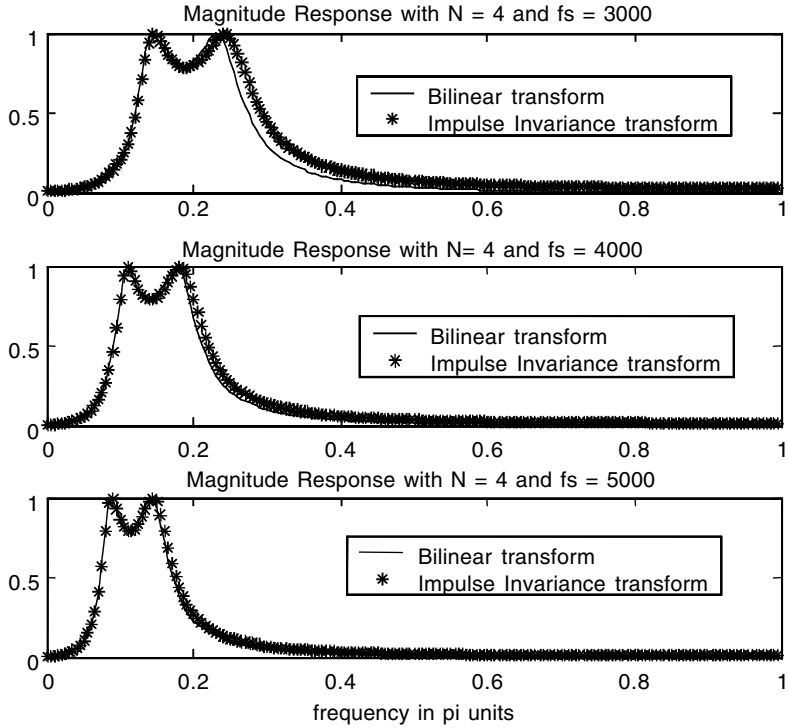

FIGURE 10.10 Plots for EOCE 10.5: Indirect Method 3.

Solution

Before the signal $x(t)$ is presented as an input to the IIR bandpass digital filter, it has to be sampled at a sampling rate that is at least two times the highest frequency in the input signal, which is $f_m = 5000/2\pi = 2500/\pi$ Hz. Thus, we will sample at least at $f_s = 2(2500/\pi) = 5000/\pi$ Hz. Let us sample at $f_s = 5000$ Hz. This corresponds to the sampling interval $T_s = 1/5000$ sec. The three analogue frequencies that are present in the input signal are located at 10, 1500 and 5000 rad/sec. Our goal is to pass only the 1500 rad/sec term. Thus, an analogue IIR filter with a passband located at [400π 800π] will pass the 1500 rad/sec term and suppress the other terms. The corresponding digital passband frequencies are at [400π(T_s) 800π(T_s)] = [400π /5000 800π/5000] = [0.08π 0.16π] rads per sample. When we use the MATLAB function `cheby1` we need to normalize this digital passband to lie in the interval [0 1]. So, when we use the function `cheby1` we will use the passband digital interval [0.08 0.16].

The following MATLAB script will carryout the design for fourth- and eighth-order IIR digital bandpass filters. It will also plot the input as well as the output signals of the filter.

```
clf
t=0:.0001:0.02;
%This is the unsampled input to the IIR digital filter
xt=sin(10*t)+sin(1500*t)+sin(5000*t);
n=0:100; fs=5000;
Ts=0.0002;
ts=n*Ts;
xn1=sin(10*ts);%This signal should not pass through the filter
xn2=sin(1500*ts);%This signal should pass
xn3=sin(5000*ts);%this signal should not pass
xn=xn1+xn2+xn3;
subplot(3,1,1); plot(t, xt);axis([0 .02 -2 2.3])
title('The input signal:sin(10*t)+sin(1500t)+sin(5000t)
 before filtering');
Rp=2;
%sampling frequency is 5000 Hz
%the normalized digital frequency is [400/5000 800/5000]
wn=[0.08 0.16];%normalized to fit in [0 1]
for n=2:2:4;
[numz, denz] = cheby1(n,Rp,wn,'bandpass');
%Now we plot the magnitude response of the IIR digital bandpass
 filter
[H, f]=freqz(numz, denz);
subplot(3,1,2); plot(f/pi, abs(H));hold on;
title('The Magnitude response of the bandpass filter using
 cheby1');
xlabel('frequency in pi units');
yn=filter(numz,denz,xn);
subplot(3,1,3); plot(ts,yn);hold on;
end
title('The output of the filter after only the sin(1500t) is
 passed');
xlabel('Time(sec)'); axis([0 .02 -2 2.3])
```

The plots are shown in Figure 10.11. You can see that the fourth-order filter can do the job. Notice that the frequencies in the input signal are very well separated and it is easy for this low-order filter to achieve reasonable results. If the frequencies in the input signal were not separated nicely, then a high-order filter with sharp edges might be needed.

EOCE 10.7

We are trying to suppress all frequencies that are higher than 200 Hz and below 400 Hz from the incoming signal $x(t)$. If the sampling frequency is 2000 Hz (the minimum sampling should be 800 Hz but sampling at higher

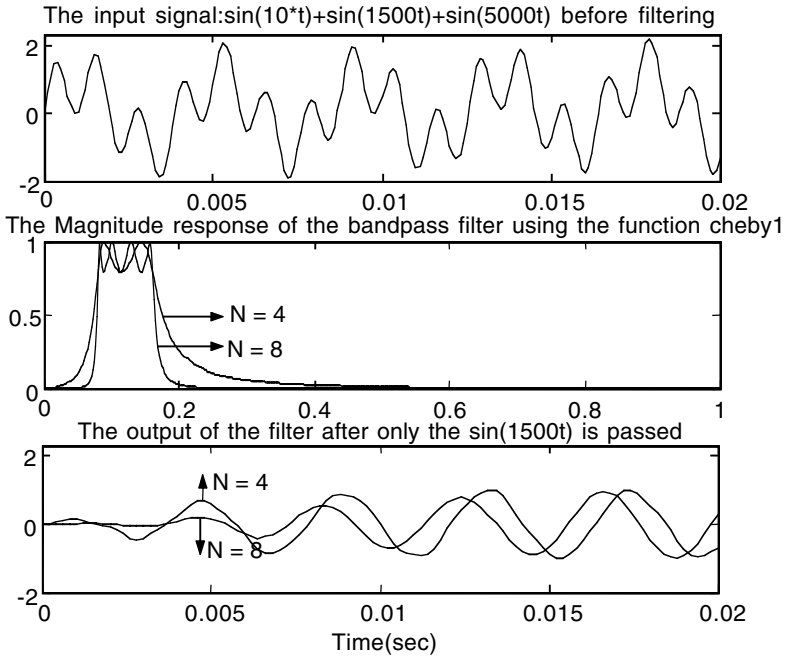

FIGURE 10.11 Plots for EOCE 10.6.

rates can give better results), design a digital bandstop IIR filter to accomplish this task. Use the Cheby2 filter in the design. Use 40 dB for the minimum allowable ripple in the stopband.

Solution

The analogue lower and upper cut-off frequencies are at 400π and 800π rad/sec, respectively.

First method: from the analogue prototype to the IIR digital filter

We will write the following MATLAB script to accomplish the design using this method. We will start with a prototype third-order analogue lowpass Cheby2 filter, then convert it to a sixth-order bandstop analogue Cheby2 filter with the cut-off frequencies at 400π for w_{cl} and 800π for w_{cu}. Then we will use the Bilinear and the Impulse Invariance transformations to get $H(z)$, the transfer function of the IIR digital filter. We will be using the MATLAB function call

```
[zs, ps, ks]=cheb2ap(N, Rs);
```

with $N = 3$ to get the third-order Cheby2 prototype lowpass filter and the MATLAB transformation

```
[nums,dens] = lp2bs(nums,dens,W0,Bw)
```

to get the sixth-order bandstop filter. The value of `W0` will be calculated as $\sqrt{(400\pi)(800\pi)} = 1777$, and the value of `Bw` is $800\pi - 400\pi = 400\pi$. With `fs` = 2000 Hz we expect the digital normalized bandstop cut-off frequencies at [400/2000 800/2000] = [0.2 0.4].

```
clf
%we will generate the lowpass analogue Cheby2 prototypes first
Rs = 40; N=3;%filter order, the actual order is 6
i=1;%index for the plots
%next are labels to be displayed later
frequency{1}='2000'; frequency{2}='4000'; frequency{3}='6000';
for fs=2000:2000:6000;
[zs, ps, ks]=cheb2ap(N, Rs);
% put in terms of transfer function
[nums, dens]=zp2tf(zs,ps,ks);
%next we transform to analogue bandpass with cut-off [400pi
 800pi]
[nums, dens]=lp2bs(nums, dens, sqrt(400*pi*800*pi), 400*pi);
%Then we use the Bilinear transform first
[numzb, denzb]=bilinear(nums, dens, fs)
%then we use the Impulse Invariance method
[numzi, denzi]=impinvar(nums, dens, fs)
%Now we plot the magnitude response of the IIR digital filter
%using the two transformation methods
[Hb, fb]=freqz(numzb, denzb,150);
[Hi, fi]=freqz(numzi, denzi,150);
subplot(3,1,i);
plot(fb/pi, abs(Hb));
hold on;
plot(fi/pi, abs(Hi),'*');
title(['Magnitude Response with N= 6 and fs = 'frequency{i}]);
%preparing for the legend
a=['Bilinear transform'];
b=['Impulse Invariance transform'];
legend(a,b);
i=i+1
axis([0 1 0 1.1]);
end
xlabel('frequency in pi units');
```

The plots are shown in Figure 10.12. You can see that the Bilinear transformation is again superior to the Impulse Invariance transformation. We do not even see the curves that correspond to the Impulse Invariance transformation.

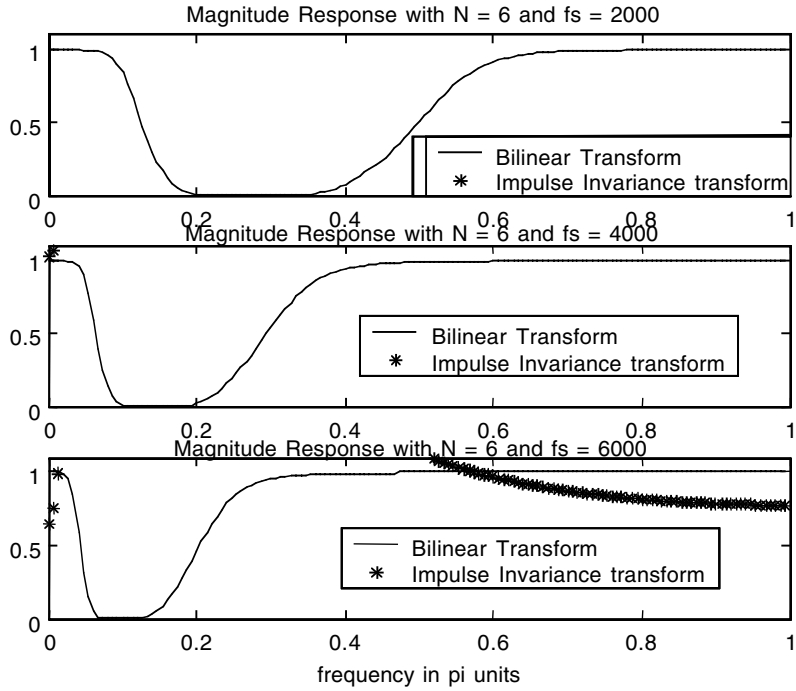

FIGURE 10.12 Plots for EOCE 10.7: Indirect Method.

The digital normalized bandstop cut-off frequencies of 0.2 and 0.4 are seen clearly in the first plot in Figure 10.12. The numerator and denominator coefficients of the transfer function of the IIR bandstop digital filter using the Bilinear transformation with $N = 6$ and `fs` = 2000 Hz are

numzb = 0.2564 –0.9974 2.0332 –2.5146 2.0332 –0.9974 0.2564
denzb = 1.0000 –2.3335 2.4483 –1.7895 1.0773 –0.3863 0.0536

The transfer function of the IIR bandpass digital using the Bilinear transform is

$$H(z) = \frac{0.2564z^6 - 0.9974z^5 + 2.0332z^4 - 2.5146z^3 + 2.0332z^2 - 0.9974z + 0.2564}{z^6 - 2.3335z^5 + 2.4483z^4 - 1.7895z^3 + 1.0773z^2 - 0.3863z + 0.0536}$$

Second method: direct design

Using this method we will use the MATLAB function

```
[numz, denz] = cheby2(n,Rs,wn,'ftype')
```

with `ftype` as `stop`, `wn` = [0.2 0.4] using `fs` = 2000, and `n` = 2 and 3 to correspond to the fourth- and sixth-order bandstop IIR digital filters.

We used `wn = [0.2 0.4]` as the normalized frequency because the equivalent cut-off digital frequencies are at $400\pi\ (1/2000) = 0.2\pi$ and at $800\pi\ (1/2000) = 0.4\pi$. Using MATLAB, the digital frequencies should be normalized to belong to the digital frequency interval [0 1]. The following MATLAB script will accomplish the design.

```
clf
Rs=40;
%sampling frequency is 2000 Hz
wn=[0.2 0.4];%normalized to fit in [0 1]
for n=2:1:3;%for fourth and sixth order digital bandstop
 filters
[numz, denz] = cheby2(n,Rs,wn,'stop')
%Now we plot the magnitude response of the IIR digital filter
[H, f]=freqz(numz, denz);
plot(f/pi, abs(H));hold on;
end
title('The Magnitude response using the function cheby2');
xlabel('frequency in pi units');
axis([0 1 0 1.1]);
```

The plots are shown in Figure 10.13. It is evident that the MATLAB function `cheby2` gave good results for the sixth-order filter and not-so-good result

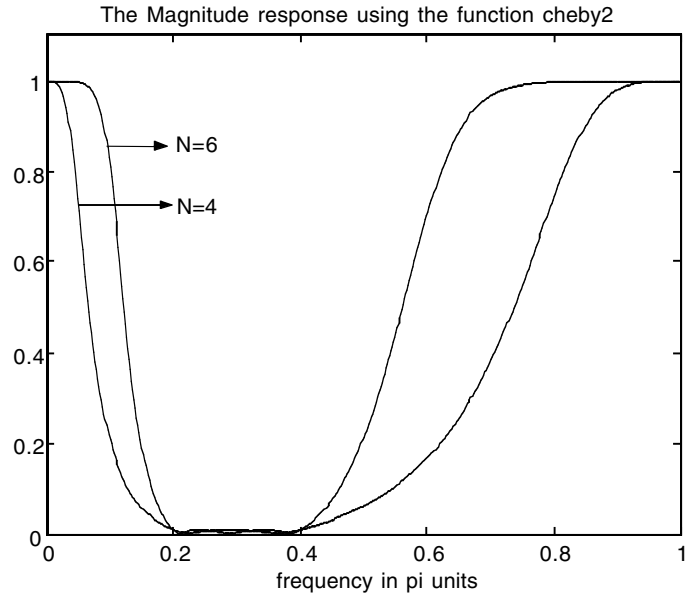

FIGURE 10.13 Plots for EOCE 10.7: Direct Method.

for the fourth-order filter with fs = 2000 Hz. The coefficients for the numerator and the denominator of the bandstop IIR digital filter for order six using the `cheby2` MATLAB function are

```
numz = 0.1968  –0.6942  1.3688  –1.6615  1.3688  –0.6942  0.1968
denz = 1.0000  –1.9095  1.4398  –0.8944  0.6772  –0.2460  0.0143
```

with the transfer function

$$H(z) = \frac{0.1968z^6 - 0.6942z^5 + 1.3688z^4 - 1.6615z^3 + 1.3688z^2 - 0.6942z + 0.1968}{z^6 - 1.9095z^5 + 1.4398z^4 - 0.8944z^3 + 0.6772z^2 - 0.2460z + 0.0143}$$

EOCE 10.8

We are interested in suppressing the term sin(1500*t*) from the input signal

$$x(t) = 1 + \sin(1500t) + \sin(5000t)$$

Design a bandstop IIR Cheby2 digital filter to accomplish this task. Plot the input signal along with the magnitude response of the filter and its output.

Solution

Before the signal $x(t)$ is presented as an input to the IIR bandstop digital filter, it has to be sampled at a sampling rate that is at least two times the highest frequency in the input signal, which is $f_m = 5000/2\pi = 2500/\pi$ Hz. Thus, we will sample at least at $f_s = 2(2500/\pi) = 5000/\pi$ Hz. Let us sample at $f_s = 5000$ Hz. This corresponds to the sampling interval $T_s = 1/5000$ sec. The three analogue frequencies that are present in the input signal are located at 0, 1500 and 5000 rad/sec. Our goal is to suppress only the 1500 rad/sec term. Thus, an analogue IIR filter with a stopband located at [300π 800π] will suppress the 1500 rad/sec term and pass the other terms. The corresponding digital stopband frequencies are at [300π(T_s) 800π(T_s)] = [300π/5000 800π/5000] = 0.06π 0.16π] rads per sample. When we use the MATLAB function `cheby2` we need to normalize this digital stopband to lie in the interval [0 1]. So when we use the function `cheby2` we will use the stopband digital interval [0.06 0.16].

The following MATLAB script will carryout the design for fourth- and eighth-order IIR digital bandstop filters. It will also plot the input as well as the output signals of the filter. We will allow a minimum attenuation of 40 dB in the stopband.

```
clf
t=0:.0001:0.01;
%This is the unsampled input to the IIR digital filter
```

```
xt=1+sin(1500*t)+sin(5000*t);
n=0:50; fs=5000;
Ts=1/fs;
ts=n*Ts;
xn1=1;%This signal should pass through the filter
xn2=sin(1500*ts);%This signal should not pass
xn3=sin(5000*ts);%this signal should pass
xn=xn1+xn2+xn3;
subplot(3,1,1); plot(t, xt);
title('The input signal:1+sin(1500t)+sin(5000t) before
 filtering');
Rs=40;
%sampling frequency is 5000 Hz
%the normalized digital frequency is [300/5000 800/5000]
wn=[0.06 0.16];%normalized to fit in [0 1]
N=2;%order is indeed 4 for we have a bandstop filter
[numz, denz] = cheby2(N,Rs,wn,'stop');
%Now we plot the magnitude response of the IIR digital bandpass
 filter
[H, f]=freqz(numz, denz);
subplot(3,1,2); plot(f/pi, abs(H));hold on;
title('The Magnitude response of the bandstop filter with
 cheby2 and N=4');
yn=filter(numz,denz,xn);
subplot(3,1,3); plot(ts,yn);hold on;
plot(ts, xn1+xn3, '*');
legend('The output of the filter', 'What should pass:
 1+sin(5000t)',0);
title('The output of the filter after only the sin(1500t) is
 suppressed');
xlabel('Time(sec)'); axis([0 0.01 -1 3]);
```

The plots are shown in Figure 10.14. You can see that the fourth-order filter could not do the job of eliminating the sin(1500t) term as seen in the lower plot of Figure 10.14. If we increase the order of the filter to eight, we will have the plots shown in Figure 10.15. In this figure we have the sin(1500t) term suppressed and the filter is working nicely.

EOCE 10.9

Consider a random signal (noise) with a dc component. The dc component has a magnitude of 4. Assume that the noise signal is limited to 10π rad/sec. Design a lowpass Butterworth digital filter to smooth the noise and uncover the dc component.

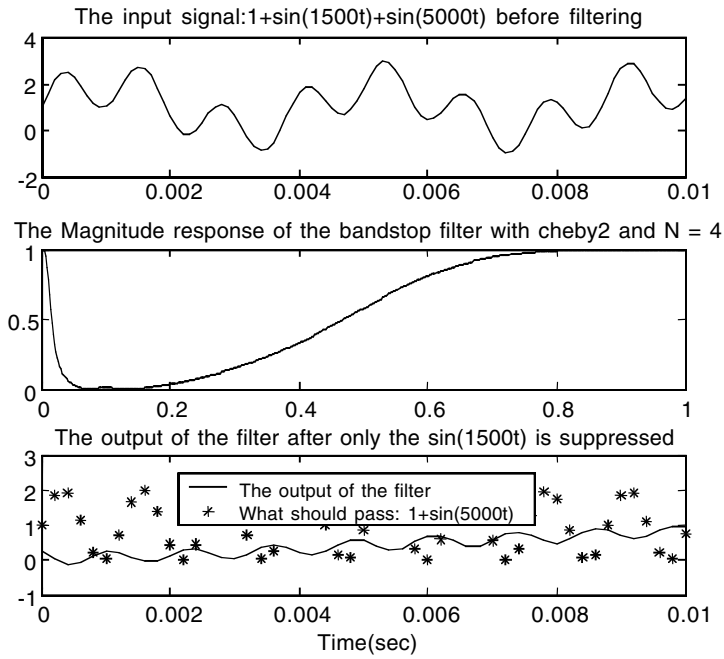

FIGURE 10.14 Plots for EOCE 10.8 filter order 4.

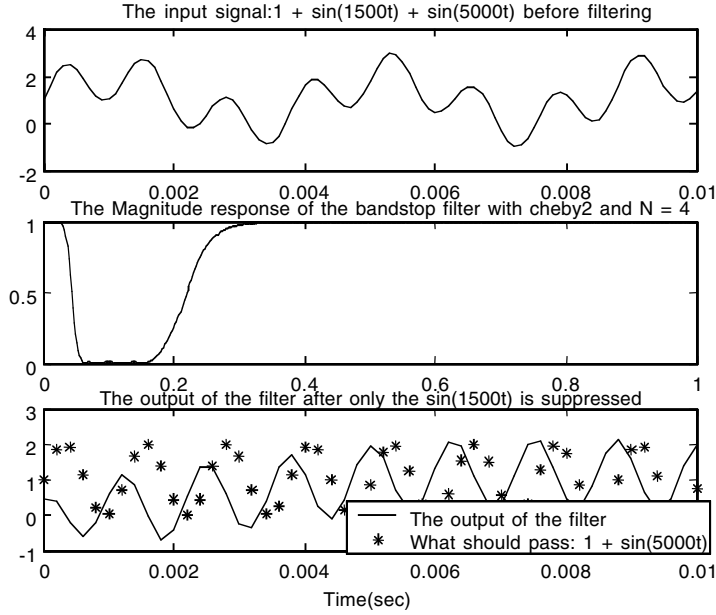

FIGURE 10.15 Plots for EOCE 10.8 filter order 8.

Solution

When the highest frequency in the input analogue noise is 100π rad/sec, we will sample the input noise at a minimum frequency of $f_s = 2[100\pi/(2\pi)] = 50$ Hz. Let us sample at $f_s = 200$ Hz. Let us use a cut-off frequency of 10π rad/sec for the analogue equivalent lowpass filter. The digital lowpass will have the corresponding 10π/200 rads per sample digital cut-off frequency. The following MATLAB script will generate the discrete input signal, pass it through the lowpass Butterworth digital filter, and produce a plot for the input, the filter magnitude, and the output.

```
clf
n=0:200; fs=200;
Ts=1/fs;
ts=n*Ts;
xn1=4;%This signal should pass through the filter
xn2=rand(1, length(n));%This signal should not pass(the noise)
xn=xn1+xn2;%the total input
subplot(3,1,1); plot(ts, xn);
title('The input signal:dc component (4) + the noise before
 filtering');
wn=1/20;%normalized to fit in the digital frequency interval
 [0 1]
N=1;%order of the Butterworth lowpass filter
[numz, denz] = butter(N,wn);
%Now we plot the magnitude response of the IIR digital lowpass
 filter
[H, f]=freqz(numz, denz);
subplot(3,1,2); plot(f/pi, abs(H));hold on;
title('The Magnitude response of the lowpass filter with
 butter and N=1');
xlabel('frequency in pi units');
yn=filter(numz,denz,xn);
subplot(3,1,3); plot(ts,yn);hold on;
plot(ts, xn1, '*');
legend('The output of the filter', 'What should pass: the dc
 4',0);
title('The output of the filter where the noise is smoothed
 out');
xlabel('Time(sec)');
```

The plots are shown in Figure 10.16. The filter is working and that is evident from the last plot in Figure 10.16 where approximately only the dc term is passed. Notice that a first order filter worked very nicely here.

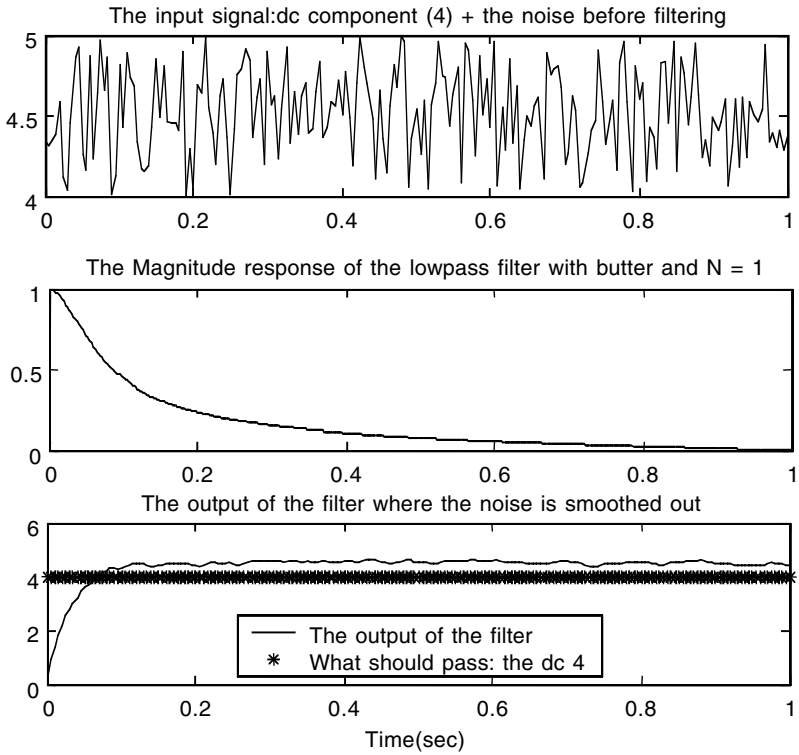

FIGURE 10.16 Plots for EOCE 10.9.

EOCE 10.10

Consider the same input signals as in EOCE 10.9, but now we want to get rid of the dc component and let only the noise pass. Design a high pass Butterworth digital filter to accomplish this.

Solution

We will still sample the analogue input at 200 Hz. The equivalent highpass filter required can have a cut-off frequency of 10π rad/sec which is equivalent to the highpass digital frequency of $10\pi/200$ rads per sample. The following MATLAB script will accomplish the design.

```
clf
n=0:200; fs=200;
Ts=1/fs;
ts=n*Ts;
xn1=4;%This signal should pass through the filter
xn2=rand(1, length(n));%This signal should not pass(the noise)
```

```
xn=xn1+xn2;%the total input
subplot(3,1,1); plot(ts, xn);
title('The input signal:dc component (4) + the noise before
 filtering');
wn=1/20;%normalized to fit in the digital frequency interval
 [0 1]
N=1;%order of the Butterworth lowpass filter
[numz, denz] = butter(N,wn,'high')
%Now we plot the magnitude response of the IIR digital
 highpass filter
[H, f]=freqz(numz, denz);
subplot(3,1,2); plot(f/pi, abs(H));hold on;
title('The Magnitude response of the highpass filter with
 butter and N=1');
xlabel('frequency in pi units');
yn=filter(numz,denz,xn);
subplot(3,1,3); plot(ts,yn);hold on;
plot(ts, xn2, '*');
legend('The output of the filter', 'What should pass: the
 noise',0);
title('The output of the filter where dc component is
 removed');
xlabel('Time(sec)');
```

The plots are shown in Figure 10.17. Again a first-order digital highpass filter has worked very nicely. Notice that the dc component of magnitude 4 is totally eliminated and the random signal is preserved.

EOCE 10.11

Design 12-order Butterworth and elliptic IIR digital bandstop filters and plot the magnitude and the phase. We desire a digital cut-off frequency of [0.2π 0.5π] rads per sample for both filter types. We also desire a 1-dB attenuation in the passband and 40-dB attenuation in the stopband for the elliptic filter.

Solution

We will use MATLAB to carryout the design and produce the plots. We will use [0.2 0.5] rads per sample with MATLAB to normalize the cut-off frequency to lie in the interval [0 1]. We will use the following MATLAB script.

```
clf%refresh the escreen
n=6 % filter order
[numz1, denz1]=butter(n,[0.2 0.5],'stop');
[numz2, denz2]=ellip(n,1,40,[0.2 0.5],'stop');
```

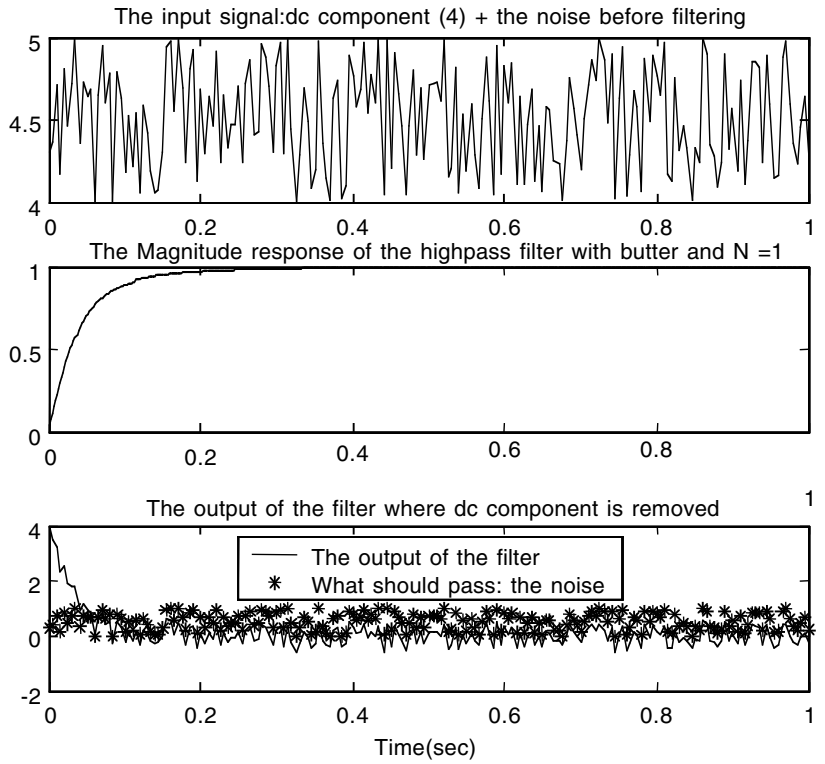

FIGURE 10.17 Plots for EOCE 10.10.

```
[H1, f]=freqz(numz1, denz1);
[H2, f]=freqz(numz2, denz2);
subplot(2,2,1);
plot(f/pi, abs(H1));
title('Magnitude: Butterworth N = 12')
hold on;
subplot(2,2,3); plot(f/pi, angle(H1));
title('Phase(radians)Butterworth');
xlabel('frequency in pi units');
hold on;
subplot(2,2,2);
plot(f/pi, abs(H2));
title('Magnitude: Elliptic N = 12')
hold on;
subplot(2,2,4); plot(f/pi, angle(H1));
title('Phase(radians) Elliptic');
hold on;
xlabel('frequency in pi units');
```

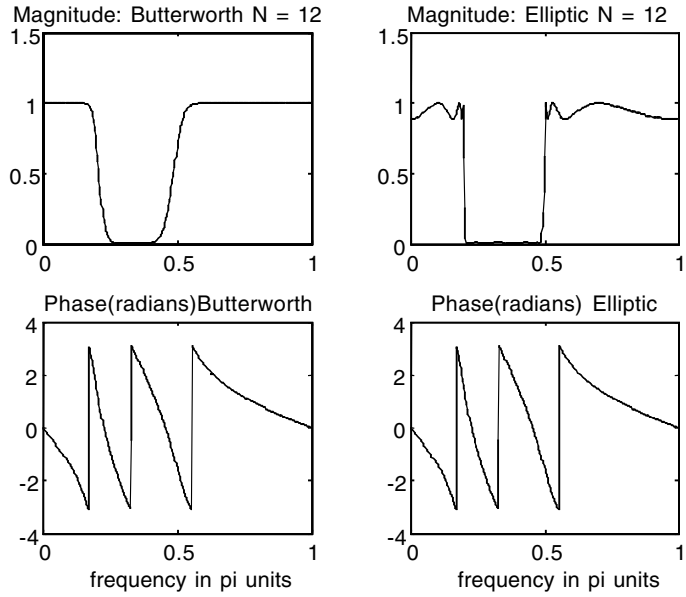

FIGURE 10.18 Plots for EOCE 10.11.

The plots are shown in Figure 10.18. You can see clearly the nonlinearity of phase, something inherent with IIR recursive digital filters. As we have mentioned earlier, a nonlinear phase can distort signals passing through the filter.

10.6 End of Chapter Problems

EOCP 10.1

We are trying to eliminate all frequencies that are higher than 1000 Hz from the incoming signal $x(t)$. Design a digital IIR filter to accomplish this task. Use all types of filters discussed in this chapter.

1. Start with the analogue IIR filter then get its equivalent digital IIR filter using the transformation methods discussed in this chapter.
2. Use MATLAB and directly design the IIR digital filter after obtaining the digital specifications.
3. Plot the magnitude responses in each case.

EOCP 10.2

Consider the input continuous signal

$$x(t) = \sin(1000t) + \cos(10t)$$

Design digital second- and fourth-order IIR lowpass Butterworth filters to attenuate the $\cos(1000t)$ term. Plot the input signal and the output of the filter.

EOCP 10.3

We are trying to eliminate all frequencies that are lower than 1000 Hz from the incoming signal $x(t)$. If the highest frequency in the incoming analogue signal is 1500 Hz, design a digital IIR filter to accomplish this task. Allow 5 dB for the maximum passband attenuation and 50 dB for the minimum stopband attenuation. Plot the magnitude response.

EOCP 10.4

Consider the input continuous signal

$$x(t) = \cos(100t) + \cos(150t) + \sin(400t)$$

1. Design a digital second- and third-order IIR elliptical filter to attenuate the $\cos(150t)$ term.
2. Design a digital second- and third-order IIR elliptical filter to attenuate the $\cos(100t)$ term.
3. Design a digital second- and third-order IIR elliptical filter to attenuate the $\cos(400t)$ term.
4. Plot the input signal and the output of the filter for each case above.

EOCP 10.5

We are trying to pass all frequencies that are higher than 10 Hz and below 50 Hz from the incoming signal $x(t)$. Design a digital IIR filter to accomplish this task. Use the Cheby1 and Cheby2 filters in the design. Use 5 dB for the maximum allowable ripple in the passband and 50 dB for the minimum allowable ripple in the stopband. Plot the magnitude responses of the digital filters.

EOCP 10.6

We are interested in passing the term $\sin(5000t)$ from the input signal

$$x(t) = \sin(10t) + \sin(1500t) + \sin(5000t)$$

Design an IIR Butterworth digital filter to accomplish this task. Plot the input signal along with the magnitude response of the filter and its output.

EOCP 10.7

We are trying to suppress all frequencies that are higher than 2000 Hz and below 4000 Hz from the incoming signal $x(t)$. Design a digital IIR filter to accomplish this task. Use the Cheby2 filter in the design. Use 60 dB for the minimum allowable ripple in the stopband. Plot the magnitude response of the digital filter.

EOCP 10.8

We are interested in suppressing the term $\sin(10t)$ from the input signal

$$x(t) = 1 + \sin(10t) + \sin(5000t)$$

Design an IIR Cheby2 digital filter to accomplish this task. Plot the input signal along with the magnitude response of the filter and its output.

EOCP 10.9

Consider the dc and the single frequency signal

$$x(t) = 1/2 + \sin(10t)$$

Assume the signal is corrupted by a noise (random noise). Assume that the noise signal is limited to 100π rad/sec.

1. Design a digital filter to smooth the noise and uncover the signal $x(t) = 1/2 + \sin(10t)$.
2. Design a digital filter to pass the $\sin(10t)$ term only.
3. Design a digital filter to pass the dc term only.
4. Plot the input and the output of the corresponding filter in each case.

EOCE 10.10

Consider the same corrupted input signal as in EOCP 10.9, but now we want to get rid of the dc component and the single frequency term. Design a Butterworth digital filter to accomplish this. Plot the input and the output of the designed filter.

EOCP 10.11

An elliptic bandstop digital IIR filter is to be designed. The corresponding analogue IIR filter should fulfill the following specifications:

$$1 >= |H(jw)|^2 >= 0.95 \qquad w <= 1200 \text{ and } w >= 1800$$
$$|H(jw)|^2 <= 0.02 \qquad 800 >= w >= 2200$$

1. Estimate the order and the cut-off frequencies of the analogue IIR filter.
2. Find the transfer function of the digital filter using the Bilinear and the Invariance methods.
3. Plot the magnitude response of the filter as well as the phase.

EOCP 10.12

Plot the magnitude and phase responses of Butterworth, Chebyshev Type I, Chebyshev Type II, and elliptic digital filters. Let the order be 5 and the cut-off frequency be 1 for the corresponding analogue counterparts. Also assume R_p = 3 dB and R_s = 60 dB for the analogue IIR filters.

EOCP 10.13

Design a digital bandstop IIR filter where the analogue counterpart should have a bandwidth β of 1000 rad/sec. The analogue IIR filter should reject the component sin(1414t) from the following signal:

$$x(t) = \sin(500t) + \sin(1414t) + \cos(2500t)$$

The analogue filter should ensure attenuation of at least 45 dB of the rejected component. Plot the input and the output of the digital filter.

EOCP 10.14

Given the following signal

$$x(t) = 1 + \sin(t) + \sin(6t)$$

design a digital filter that eliminates the component sin(t). Plot the filter magnitude response. Assume 40 dB allowable ripple in the stopband. Plot the input and the output of the digital filter.

EOCP 10.15

Consider the circuit in Figure 10.19. Use 1 Henry (H) for L, 1 Ω for R_1 and 1 Ω for R_f. Design a digital filter to approximate the input–output characteristics of the circuit. Plot the input and the output of the filter by choosing appropriate input. Use the Bilinear transformation with T_s = 0.1. Repeat for different values for T_s. What do you conclude?

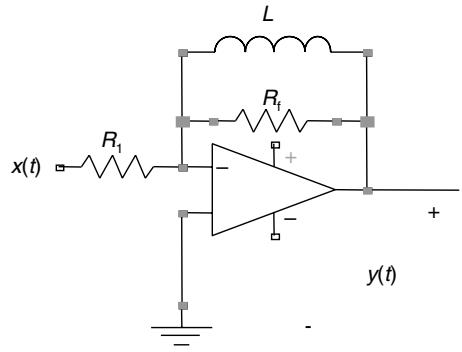

FIGURE 10.19 Circuit for EOCP 10.15.

EOCP 10.16

Consider the same circuit in Figure 10.19. Replace the inductor by a 1-F capacitor. Use 1 Ω for R_1 and 1 Ω for R_f. Design a digital filter to approximate the input–output characteristics of the circuit. Plot the input and the output of the filter by choosing appropriate input. Use Bilinear transformation with $T_s = 0.1$. Repeat for different values for T_s. What do you conclude?

EOCP 10.17

Consider the same circuit as in Figure 10.19. Add a capacitor of 1 F in series with the 1-H inductor. Use 1 Ω for R_1 and 1 Ω for R_f. Design a digital filter to approximate the input–output characteristics of the circuit. Plot the input and the output of the filter by choosing appropriate input. Use Bilinear transformation with $T_s = 0.1$. Repeat for different values for T_s. What do you conclude?

EOCP 10.18

Consider the following differential equation:

$$y''(t) + y'(t) + y(t) = x(t)$$

Design a digital filter to approximate the input–output characteristics of the system. Plot the input and the output of the filter by choosing appropriate input. Use the Bilinear transformation with $T_s = 0.1$. Repeat for different values for T_s. What do you conclude?

EOCP 10.19

Consider the following differential equation:

$$y''(t) + y(t) = x(t)$$

Design a digital filter to approximate the input–output characteristics of the system. Plot the input and the output of the filter by choosing appropriate input. Use the Bilinear transformation with T_s = 0.1. Repeat for different values for T_s. What do you conclude?

EOCP 10.20

Consider the following differential equation:

$$y''(t) + y'(t) + y(t) = x(t) + x'(t)$$

Design a digital filter to approximate the input–output characteristics of the system. Plot the input and the output of the filter by choosing appropriate input. Use the Bilinear transformation with T_s = 0.1. Repeat for different values for T_s. What do you conclude?

EOCP 10.21

The transfer functions of an integrator and a differentiator are

$$H_1(s) = \frac{1}{s} \text{ and } H_2(s) = s$$

1. Use the Bilinear transformation to approximate the two transfer functions with T_s as the sampling interval. Find $H_1(z)$ and $H_2(z)$.
2. Use the Impulse Invariance transformation to approximate the two transfer functions with T_s as the sampling interval. Find $H_1(z)$ and $H_2(z)$.
3. Plot the magnitude responses for $H_1(s)$ and $H_2(s)$.
4. For different values of T_s, plot the magnitude responses for $H_1(z)$ and $H_2(z)$.
5. Give comments on the results.

11

Finite Impulse Response (FIR) Digital Filters

11.1 Introduction

At the beginning of this chapter we give a brief description of the FIR digital filter and an example that we hope will motivate us in approaching this huge topic of digital filtering.

11.1.1 What Is an FIR Digital Filter?

An FIR digital filter is a system with input $x(n)$ and output $y(n)$. The output $y(n)$ at any discrete value n depends on the present and past values of $x(n)$ and not on the current or past values of $y(n)$. This type of system is known as a *nonrecursive* system.

As we will see later, FIR digital filters are known for their finite duration responses where the impulse response, $h(n)$, of an FIR filter is nonzero for $n = 0, \ldots, N - 1$ where N is the number of samples in $h(n)$. Also, FIR filters (systems) are found to be always stable, and thus very popular. In addition to stability, FIR filters can have linear phase. A nonlinear phase system can distort the input signal to the system. A disadvantage to the FIR digital filter is the complexity of the implementation if N is very large. As we will see late in this chapter, the larger the N is, the better the frequency response of the FIR filter.

11.1.2 A Motivating Example

Put yourself in the city of Baghdad sitting on a hill and counting the F16 jet fighters crossing over. Let us say that we are counting the number of jets passing over every minute and are interested in calculating the average number of jets every minute for the last 4 minutes. Let us say that after 10 minutes we have collected the following data.

TABLE 11.1

Minutes	Jets
1	2
2	3
3	0
4	1
5	5
6	1
7	4
8	0
9	3
10	5

Next we will find the average every minute for the past 4 minutes. The following table is obtained.

TABLE 11.2

Minutes	Jets	Average
1	2	—
2	3	—
3	0	—
4	1	(2 + 3 + 0 + 1)/4 = 6/4
5	5	(3 + 0 + 1 + 5)/4 = 9/4
6	1	(0 + 1 + 5 + 1)/4 = 7/4
7	4	(1 + 5 + 1 + 4)/4 = 11/4
8	0	(5 + 1 + 4 + 0)/4 = 10/4
9	3	(1 + 4 + 0 + 3)/4 = 8/4
10	5	(4 + 0 + 3 + 5)/4 = 12/4

If we look at Table 11.1 we can see that there are sudden jumps in the number of jets per minute. This sudden jumping indicates high frequency components in the signal $x(n)$ used to represent the number of jets crossing over per minute. However, if we look at the average values we do not see a serious jump. On the contrary, we see somehow only small deviations in the data. These small deviations or gradual changes indicate low frequency in the output $y(n)$ used to represent the average values. Note also that we have to wait until the fourth minute to find the first average value. So the first average value is

$$y(4) = \left[\frac{1}{4}\big(x(1) + x(2) + x(3) + x(4)\big)\right]$$

Observe also that if we stop counting after the tenth minute,

$$y(14) = \frac{1}{4}(x(11) + x(12) + x(13) + x(14) = 0$$

indicates that if the input vanishes after sometime, the output will also vanish after a different time. In general, we can write the following for the average $y(n)$:

$$y(n) = \frac{1}{4}\left[x(n-3) + x(n-2) + x(n-1) + x(n)\right] = \frac{1}{4}\sum_{k=n-3}^{n} x(k) \tag{11.1}$$

We can also write $y(n)$ in a different and "interesting" way. We write

$$y(n) = \frac{1}{4}x(n-3) + \frac{1}{4}x(n-2) + \frac{1}{4}x(n-1) + \frac{1}{4}x(n) = \sum_{k=n-3}^{n} \frac{1}{4}x(k) \tag{11.2}$$

At this point we realize that we have a lowpass system (filter) with input $x(n)$ and output $y(n)$ given by Equation (11.2). What is the impulse response $h(n)$ for this filter? We usually start indexing at zero. So let us adopt this method and rewrite Equation (11.2) as

$$y(n) = \sum_{k=0}^{k=3} \frac{1}{4}x(n-k) = \frac{1}{4}\left[x(n) + x(n-1) + x(n-2) + x(n-3)\right] \tag{11.3}$$

Equation (11.3) is the same as Equation (11.1). Consider now the following well-known convolution sum equation

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k) \tag{11.4}$$

where N is the number of coefficients $h(n)$. If we now compare Equation (11.3) and (11.4) we see that in (11.4) we have N $h(n)$ coefficients and in Equation (11.3) we have $N - 1 = 3$ or $N = 4$ coefficients. We also see that the $h(n)$ coefficients in Equation (11.3) are all the same constant $1/4$ with $h(0) = h(1) = h(2) = h(3) = 1/4$. Thus the impulse response for our average example is

$$h(n) = \begin{cases} 1/4 & 0 \le n \le 3 \\ 0 & \text{otherwise} \end{cases}$$

To be convinced that the average filter is a lowpass system, we can look at the frequency response $|H(k)|$, the magnitude of the fft of the signal $h(n)$. Is this filter with the constant coefficients of $1/4$ the best filter? To answer this question, let us play with the coefficients of $h(n)$ and consider

$$h(n) = \begin{cases} 1/8 & n = 0 \\ 1/4 & n = 1 \\ 1/4 & n = 2 \\ 1/8 & n = 3 \end{cases}$$

We will use the following MATLAB script to look at the frequency responses $|H(k)|$, the magnitude of the fft of the signal $h(n)$, for the two impulse responses (given above). The script follows.

```
h1n=[1/4 1/4 1/4 1/4]; h2n=[1/8 1/4 1/4 1/8];
X1=fft(h1n,1024);X2=fft(h2n,1024);
N=1024; n1=1:ceil((N+1)/2);
n2=ceil((N+1)/2)+1:N;%ceil(2.1)=3
k=1/(N/2)*(ceil(-(N-1)/2):ceil((N-1)/2));
X1=[X1(n2) X1(n1)];X2=[X2(n2) X2(n1)];
plot(k,abs(X1)/max(abs(X1))); hold on;
plot(k,abs(X2)/max(abs(X2)),'--');
legend('h=[1/4 1/4 1/4 1/4]','h=[1/8 1/4 1/4 1/8]',0);
title('The frequency response of the two averaging filters');
xlabel('Frequency index k').
```

The plots are shown in Figure 11.1. In this figure you can see the difference in the frequency responses. The one with constant coefficients (sharp edges) has a narrower passband and high ripples. The realizations of the FIR filters for the average example are shown in Figures 11.2 and 11.3. In Figure 11.2 we have constant $h(n)$ and in Figure 11.3 we have variable $h(n)$.

11.2 FIR Filter Design

The frequency response of a linear time-invariant system is periodic with period 2π where we have $H(e^{j\theta}) = H(e^{j(\theta+2k\pi)})$. We have seen that the impulse response and the frequency response are related according to

$$H\left(e^{j\theta}\right) = \sum_{n=-\infty}^{\infty} h(n)e^{-j\theta n} \tag{11.5}$$

For causal and nonrecursive filters, the difference equation is given by

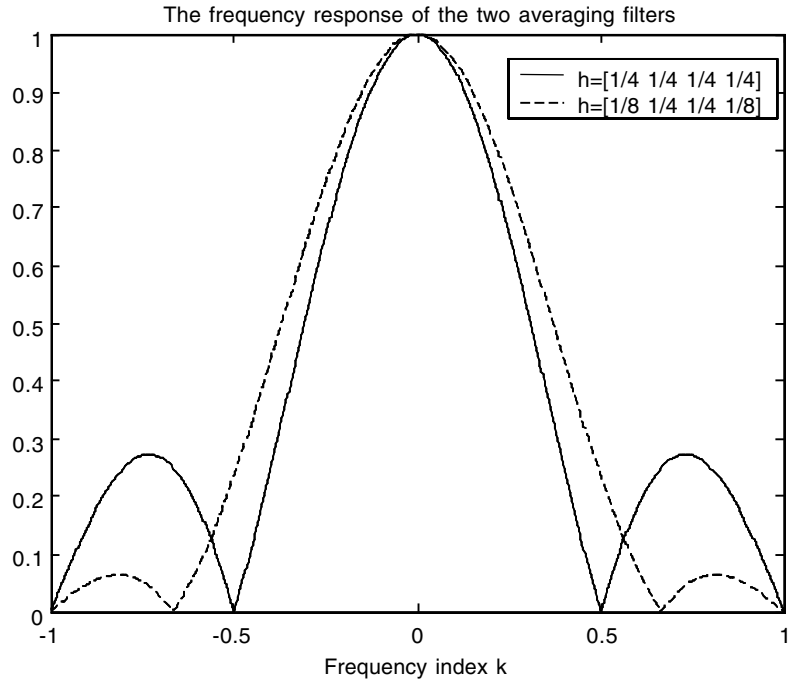

FIGURE 11.1 Comparison of two frequency responses.

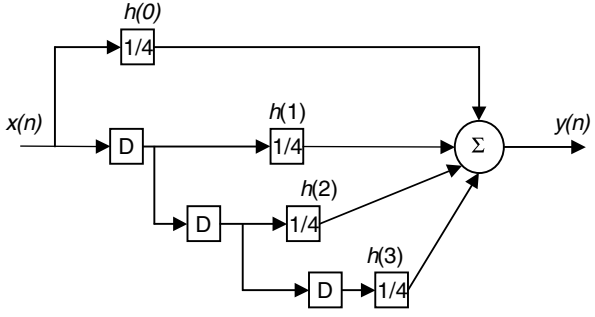

FIGURE 11.2 Constant coefficients realization.

$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k) \tag{11.6}$$

Now if we attempt to find the steady-state output of the system in Equation (11.6) with $x(n) = e^{j\theta n}$, we rewrite Equation (11.6) as

$$e^{j\theta n} H\left(e^{j\theta}\right) = b_0 e^{j\theta n} + b_1 e^{j\theta(n-1)} + \ldots + b_{N-1} e^{j\theta(n-(N-1))} \tag{11.7}$$

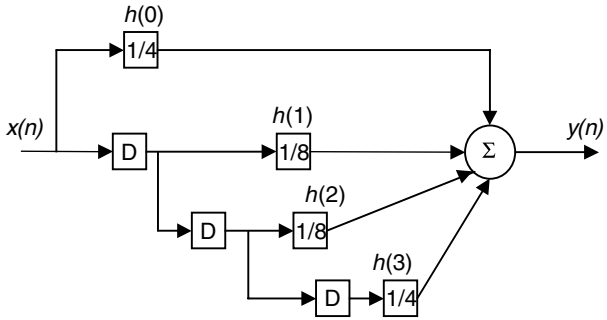

FIGURE 11.3 Variable coefficients realization.

Equation (11.7) can be simplified into the equation

$$H\left(e^{j\theta}\right) = b_0 + b_1 e^{-j\theta} + \ldots + b_{N-1} e^{-j\theta(N-1)} \tag{11.8}$$

Now Equation (11.8) can be written in compact form as

$$H\left(e^{j\theta}\right) = \sum_{n=0}^{N-1} b_n e^{-j\theta n} \tag{11.9}$$

From Equation (11.9), if we consider a causal filter where $h(n) = 0$ for $n < 0$ and $h(n) = 0$ for $n > N - 1$, we can rewrite (11.5) as

$$H\left(e^{j\theta}\right) = \sum_{n=0}^{N-1} h(n) e^{-j\theta n} \tag{11.10}$$

At this point we can see very clearly (by comparing Equations (11.10) and (11.9)) that $h(n) = b_n$. In this case, the constants $h(n)$ are the same as the nonrecursive difference equation coefficients given in Equation (11.6). Knowing the difference equation as in (11.6), we can implement the FIR filter using a simple computer program.

11.2.1 Stability of FIR Filters

If we multiply and divide Equation (11.8) by $e^{j\theta(N-1)}$ we will have

$$H\left(e^{j\theta}\right) = \frac{b_0 e^{j\theta(N-1)} + b_1 e^{j\theta(N-2)} + \ldots + b_{N-1}}{e^{j\theta(N-1)}} \tag{11.11}$$

From Equation (11.11) we can write the transfer function of the FIR filter with $e^{j\theta} = z$ as

$$H(z) = \frac{b_0 z^{N-1} + b_1 z^{N-2} + \ldots + b_{N-1}}{z^{N-1}} \tag{11.12}$$

With $h(n) = b_n$ we have

$$H(z) = \frac{h(0) z^{N-1} + h(1) z^{N-2} + \ldots + h(N-1)}{z^{N-1}} \tag{11.13}$$

In Equation (11.13) we see a very important property. This property is evident in the denominator of $H(z)$. We have $N - 1$ poles at $z = 0$. This indicates that the FIR filter is always stable since all the poles lie within the unit circle.

11.2.2 Linear Phase of FIR Filters

Another desired property of the FIR filter is the linear phase. We know that a linear phase corresponds to a delay in the output of the filter only. However, a nonlinear phase FIR filter will distort the shape of the output of the filter.

From Equation (11.10) we see that the length of the filter (the number of the $h(n)$ points) is N. Set the length N as $2m + 1$, where m is an integer or an integer divided by two. Thus, N can be an odd or even number. The filter will have linear phase if the impulse response $h(n)$ has even symmetry with $h(n) = h(2m - n)$. To see that, let us substitute $h(n) = h(2m - n)$ in Equation (11.10). We have

$$H\left(e^{j\theta}\right) = \sum_{n=0}^{N-1} h(n) e^{-j\theta n} = \sum_{n=0}^{N-1} h(2m-n) e^{-j\theta n} \tag{11.14}$$

If we substitute $k = 2m - n$, with $N = 2m + 1$, in Equation (11.14) we get

$$H\left(e^{j\theta}\right) = \sum_{k=0}^{2m} h(k) e^{-j\theta(2m-k)} = e^{-j\theta 2m} \sum_{k=0}^{2m} h(k) e^{j\theta k} \tag{11.15}$$

Note that

$$\left(\sum_{k=0}^{2m} h(k) e^{j\theta(k)} \right)^* = \sum_{k=0}^{2m} h(k) e^{-j\theta k}$$

Thus Equation (11.15) can be written as

$$H\left(e^{j\theta}\right) = e^{-j\theta 2m}\left(H\left(e^{j\theta}\right)\right)^{*} \tag{11.16}$$

It is clear from Equation (11.16) that the magnitude on both sides is the same. However, the angle is not. From Equation (11.16) we have

$$\angle H\left(e^{j\theta}\right) = \angle e^{-j\theta 2m} + \angle H\left(e^{j\theta}\right)^{*} \tag{11.17}$$

With the knowledge that

$$\angle H\left(e^{j\theta}\right) = -\angle H\left(e^{j\theta}\right)^{*}$$

we write Equation (11.17) as

$$\angle H\left(e^{j\theta}\right) = -\theta 2m - \angle H\left(e^{j\theta}\right) \tag{11.18}$$

Equation (11.18) clearly says that

$$\angle H\left(e^{j\theta}\right) = -m\theta \tag{11.19}$$

This is a clear indication that $h(n)$ has even symmetry by choosing $N = 2m + 1$. Again, N can be an odd or even integer.

11.3 Design Based on the Fourier Series: The Windowing Method

Let us say that we are after a certain frequency response $H(e^{j\theta})$. This desired frequency response is periodic with period 2π and thus can be written using the Fourier series technique as

$$H\left(e^{j\theta}\right) = \sum_{n=\infty}^{+\infty} h(n)e^{-j\theta n} \tag{11.20}$$

We can see from Equation (11.20) that the Fourier series coefficients are the points in the desired impulse response $h(n)$. These points are obtained by using the exponential form of the Fourier series as

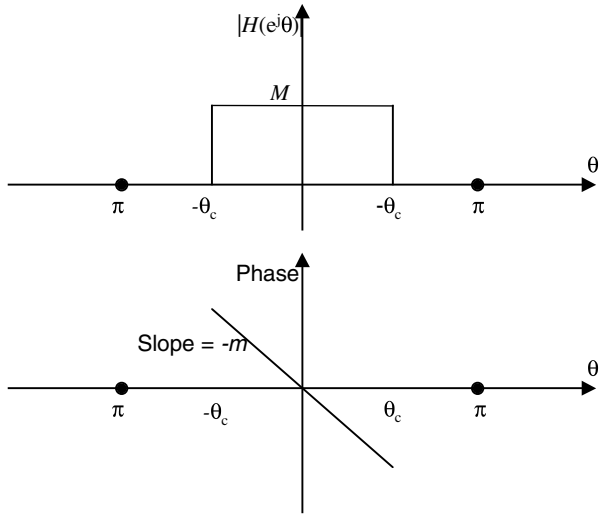

FIGURE 11.4 Ideal lowpass filter.

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H\left(e^{j\theta}\right)e^{j\theta n}d\theta \tag{11.21}$$

For the majority of practical application, $H(e^{j\theta})$ has sharp transition. This will produce an infinite length and noncausal impulse response $h(n)$. Once we find a causal $h(n)$ with finite duration, the design is complete.

11.3.1 Ideal Lowpass FIR Filter Design

Consider the desired frequency response for the ideal lowpass FIR filter shown in Figure 11.4

$$H_{lp}\left(e^{j\theta}\right) = \begin{cases} Me^{-jm\theta} & |\theta| \le \theta_c \\ 0 & \theta_c < \theta \le \pi \end{cases} \tag{11.22}$$

where θ_c is the cutoff frequency. The desired impulse response can be computed using Equation (11.21) as

$$h_{lp}(n) = \frac{1}{2\pi}\int_{-\theta_c}^{\theta_c} Me^{-jm\theta}e^{j\theta n}d\theta = \frac{M}{2\pi}\int_{-\theta_c}^{\theta_c} e^{j\theta(n-m)}d\theta \tag{11.23}$$

To find this causal $h_c(n)$ we can define a window $w(n)$ in the interval $0 \le n \le N-1$ where N is the number of the Fourier coefficients $h_c(n)$. We write

$$w(n) = \begin{cases} 1 & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.24}$$

With this window we write the causal impulse response $h_c(n)$ as

$$h_c(n) = h(n)w(n) \tag{11.25}$$

We know that multiplication in the time domain as seen in Equation (11.25) is convolutionary in the frequency domain. We then write

$$H_c\left(e^{j\theta}\right) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H\left(e^{jr}\right)W\left(e^{j(\theta-r)}\right)dr \tag{11.26}$$

In Equation (11.26), $H_c(e^{j\theta})$ is the convolution between the frequency response $H(e^{j\theta})$ and the frequency response of the window $w(n)$. If we consider the case when $w(n) = 1$ for all n, then this is equivalent to no truncation at all. In this case, $W(e^{j\theta})$ is a pulse train, and $W(e^{j(\theta-r)})$ in Equation (11.26) can be replaced by $\delta(\theta - r)$. With this value for $W(e^{j(\theta-r)})$, $H_c(e^{j\theta})$ in Equation (11.26) will be the same as $H(e^{j\theta})$.

Based on the discussion above we see that if we choose $w(n)$ to be finite in duration, then we will have a narrow concentration of frequency points, and we will not have the pulse train as the case with $w(n) = 1$ for all n. Thus, with a finite duration $w(n)$, $H_c(e^{j\theta})$ will resemble $H(e^{j\theta})$ everywhere except at the rapid sharp edges in $H(e^{j\theta})$, the desired frequency responses. For practical issues in the implementation of the digital filter, we will not take too may samples in the window $w(n)$.

We can simplify Equation (11.23) to get

$$h_{lp}(n) = \frac{M}{\pi(n-m)}\sin\left(\theta_c(n-m)\right) \quad -\infty \le n + \infty \tag{11.27}$$

where M is the magnitude and m in the delay for the ideal lowpass FIR filter. For practical impulse response we truncate the response in Equation (11.27) for $n < 0$ and for $n > N - 1 = 2m$ to get

$$h_{lp}(n) = \begin{cases} \dfrac{M}{\pi(n-m)}\sin(\theta_c(n-m) & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.28}$$

where N is the length of the lowpass FIR filter. Notice that this truncation is simply obtained by multiplying $h(n)$ in Equation (11.27) by the rectangular window, $w(n) = 1$ for $0 \le n \le N - 1$. We will have more to say about other types of windows later in the chapter.

11.3.2 Other Ideal Digital FIR Filters

Consider the magnitude responses for highpass, bandpass, and bandstop digital filters as shown in Figures 11.5, 11.6, and 11.7. We can use Equation (11.21) to find $h(n)$ for the highpass filter. Using Equation (11.21) we write

$$h(n)_{hp} = \frac{1}{2\pi}\int_{-\pi}^{-\pi+\theta_c} Me^{j\theta n}d\theta + \frac{1}{2\pi}\int_{\pi-\theta_c}^{\pi} Me^{j\theta n}d\theta \tag{11.29}$$

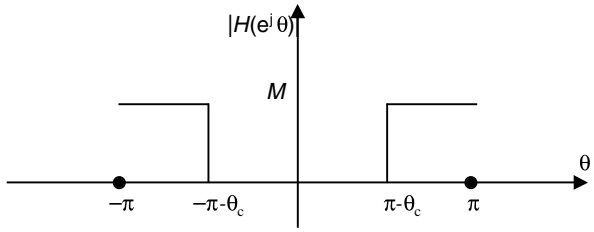

FIGURE 11.5 Ideal highpass filter.

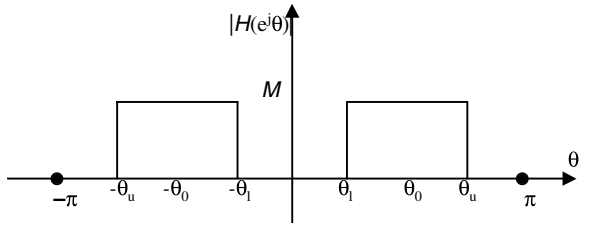

FIGURE 11.6 Ideal bandpass filter.

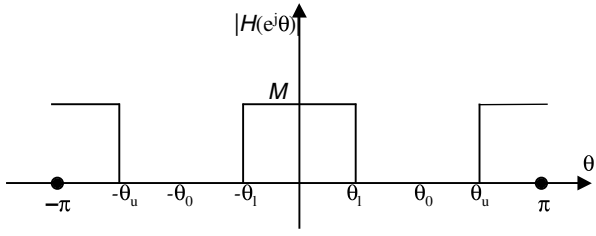

FIGURE 11.7 Ideal bandstop filter.

where we have assumed a zero linear phase shift for now. We will compensate this linear phase shift later. From Equation (11.29) we get

$$h_{hp}(n) = \frac{M}{2\pi}\left[\frac{e^{jn(-\pi+\theta_c)} - e^{-jn\pi}}{jn}\right] + \frac{M}{2\pi}\left[\frac{e^{jn\pi} - e^{jn(\pi-\theta_c)}}{jn}\right]$$

Simplifying further with $e^{jn\pi} = e^{-jn\pi}$ we get

$$h_{hp}(n) = \frac{M2}{2\pi n}\cos(n\pi)\left[\frac{e^{jn\theta_c} - e^{-jn\theta_c}}{2j}\right] = \frac{M}{n\pi}(-1)^n \sin(n\theta_c) \tag{11.30}$$

θ_c in Equation (11.30) is measured from $\theta = \pi$, the center of symmetry of the highpass filter.

Now to make $h_{hp}(n)$ causal and of finite length, we shift $h_{hp}(n)$ by m samples and finally write

$$h_{hp}(n) = \frac{M}{(n-m)\pi}(-1)^{n-m}\sin((n-m)\theta_c) \tag{11.31}$$

Other types can be derived using Equation (11.21). For the bandpass filter we can think of it as a lowpass filter with cut-off frequency θ_u minus another lowpass filter with cut-off frequency θ_l. Thus the causal N-terms bandpass filter in Figure 11.6 can be approximated by the impulse response

$$h_{bp}(n) = \frac{M}{(n-m)\pi}\left[\sin(\theta_u((n-m) - \sin(\theta_l(n-m))\right] \qquad 0 \le n \le N-1 \tag{11.32}$$

The bandstop causal and linear phase FIR filter in Figure 11.7 can be approximates as

$$h_{bs}(n) = \begin{cases} \left[\dfrac{\pi - (\theta_u - \theta_l)}{\pi}\right]M & n = m \\ M\left[\dfrac{\sin(\theta_l(n-m))}{(n-m)\pi} - \dfrac{\sin(\theta_u(n-m))}{(n-m)\pi}\right] & 1 \le n \le N-1 \end{cases} \tag{11.33}$$

11.3.3 Windows Used in the Design of the Digital FIR Filter

We have seen earlier in this chapter that the impulse response $h(n)$ that was obtained using Equation (11.21) is of infinite length. We had to truncate this $h(n)$ to obtain a finite length $h(n)$.

In previous sections we have used the rectangular window, where we multiplied the infinite length $h(n)$ by $w(n)$, the rectangular window of length N. In the case of the design of the causal linear phase lowpass filter we had

$$h_{lp}(n) = w_R(n)h(n) \tag{11.34}$$

where $h(n)$ in Equation (11.34) is of infinite length. It is known that multiplying by the rectangular window will produce ripples in the magnitude plots of $H_{lp}(e^{j\theta})$. To reduce these ripples we can make the sharp transition in $w(n)$ smoother. With the goal to reduce these ripples different windows were found. Some of these windows are given next including the rectangular window.

1. The rectangular window with

$$w(n) = \begin{cases} 1 & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.35}$$

2. The Hanning window with

$$w(n) = \begin{cases} \frac{1}{2}\left[1 - \cos\left(\frac{2\pi n}{N-1}\right)\right] & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.36}$$

3. The Hamming window with

$$w(n) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right) & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.37}$$

4. The Blackman window with

$$w(n) = \begin{cases} 0.42 - 0.5\cos\left(\frac{2\pi n}{N-1}\right) + 0.08\cos\left(\frac{4\pi n}{N-1}\right) & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.38}$$

5. The Kaiser window with

$$w(n) = \begin{cases} \dfrac{I_0\left(0.5(N-1)\beta\sqrt{\left(0.5(N-1)\right)^2 - \left(n - 0.5(N-1)\right)^2}\right)}{I_0\left(0.5(N-1)\beta\right)} & 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{11.39}$$

It is known that $I_o(x)$ is the zeroth-order Bessel function and is given by

$$I_0(x) = 1 + \sum_{k=1}^{\infty} \left(\frac{0.5x}{k!} \right)^2 \tag{11.40}$$

It is known that $I_o(x)$ can converge rapidly with as much as $k = 20$. β is a parameter that is used to adjust the trade-off between the main lobe width and the side lobe level in the magnitude plot of the frequency response of the FIR filter. You can easily see that if $\beta = 0$, the Kaiser window reduces to the rectangular window.

11.3.4 Which Window Gives the Optimal *h*(*n*)?

Next we will show that the rectangular window, despite the ripples it introduces in the magnitude plot of its frequency response, gives the best or the optimal $h(n)$. Consider the desired frequency response $H_d(e^{j\theta})$ for all θ in the interval $[-\pi, \pi]$. The desired impulse response $h_d(n)$ is obtained from $H_d(e^{j\theta})$ using the discrete time Fourier integral

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d\left(e^{j\theta}\right) e^{j\theta n} d\theta \qquad -\infty \le n \le +\infty \tag{11.41}$$

and the desired frequency response $H_d(e^{j\theta})$ is obtained from the impulse response $h_d(n)$ using the summation equation

$$H_d\left(e^{j\theta}\right) = \sum_{n=-\infty}^{+\infty} h_d(n) e^{-j\theta n} \tag{11.42}$$

In most cases $h_d(n)$ is two sided and infinite in length. But we need a practical $h_d(n)$ that is causal and of finite length. Let us truncate $h_d(n)$ and write

$$h_t(n) = h_d(n) \qquad 0 \le n \le N - 1 \tag{11.43}$$

The discrete Fourier transform of $h_t(n)$ is $H_t(e^{j\theta})$. Our goal then is to see to what degree $H_t(e^{j\theta})$ approximates $H_d(e^{j\theta})$. Let us define the error between $H_t(e^{j\theta})$ and $H_d(e^{j\theta})$ for all θ as

$$e = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| H_t\left(e^{j\theta}\right) - H_d\left(e^{j\theta}\right) \right|^2 d\theta \tag{11.44}$$

which is a commonly used approximation criteria in minimizing the integral squared error. Using the Parseval's Theorem we have

$$\frac{1}{2\pi}\int_{-\pi}^{\pi}\left|H_t\left(e^{j\theta}\right)-H_d\left(e^{j\theta}\right)\right|^2 d\theta=\sum_{n=-\infty}^{+\infty}\left|h_t(n)-h_d(n)\right|^2 \tag{11.45}$$

Thus the error in Equation (11.44) can be written as

$$e=\sum_{n=-\infty}^{+\infty}\left|h_t(n)-h_d(n)\right|^2 \tag{11.46}$$

Keep in mind that $h_t(n)$ is defined only in the interval $0 \le n \le N-1$ and that the length of $h_t(n)$ is $N = 2m + 1$ as defined in Section 11.2 where m is an integer or an integer divided by two. With this we write Equation (11.46) as

$$e=\sum_{n=-m}^{m}\left|h_t(n)-h_d(n)\right|^2+\sum_{n=-\infty}^{-m-1}h_d{}^2(n)+\sum_{n=m+1}^{\infty}h_d{}^2(n) \tag{11.47}$$

To minimize the error in Equation (11.47) we set $h_t(n) = h_d(n)$ in the interval $-m \le n \le m$. Thus we see that $h_t(n)$ is obtained from the desired $h_d(n)$ by truncation. This truncation gives a finite length signal $h_t(n)$ that is not causal. To make $h_t(n)$ causal we delay $h_t(n)$ by m samples. Thus, our discrete impulse response $h(n)$ is obtained as

$$h(n)=h_t(n-m) \tag{11.48}$$

which has the same magnitude as $h_t(n)$ and also has a linear phase shift of θm.

11.3.5 Design of a Digital FIR Differentiator

We may be interested in calculating the rate of change of a signal to see how the signal is changing. One application where digital differentiation may occur is in the design of a proportional-integral-derivative control. Given the signal

$$x(t)=X\cos(wt+\varphi) \tag{11.49}$$

as an input to a differentiator system, the output $y(t)$ will be

$$y(t)=-wX\sin(wt+\varphi)=wX\cos\left(wt+\varphi+\frac{\pi}{2}\right) \tag{11.50}$$

Using phasors, we can find the ratio of the output to the input in an attempt to find the transfer function $H(jw)$. The phasor of $x(t)$ is $X \angle \varphi$ and that of the output is $wX \angle(\varphi + \pi/2)$. The ratio then is

$$H(jw) = \frac{wX\angle\left(\varphi + \frac{\pi}{2}\right)}{X\angle\varphi} = w\angle\left(\varphi + \frac{\pi}{2} - \varphi\right) = w\angle\frac{\pi}{2} \tag{11.51}$$

We can write the ratio in Equation (11.51) as

$$H(jw) = w\angle\frac{\pi}{2} = w\cos\left(\frac{\pi}{2}\right) + jw\sin\left(\frac{\pi}{2}\right) = jw \tag{11.52}$$

Thus, the transfer function of the differentiator is

$$H(jw) = jw \tag{11.53}$$

In Equation (11.53) w is the continuous frequency. The digital frequency θ is related to the continuous frequency w by the relation

$$w = \theta T_s \tag{11.54}$$

where T_s is the sampling period. Thus, the digital transfer function is

$$H\left(e^{j\theta}\right) = j\frac{\theta}{T_s} \tag{11.55}$$

The magnitude of $H(e^{j\theta})$ is a line with slope $1/T_s$ in the range $[0, \pi]$ and with slope $-1/T_s$ in the range $[-\pi, 0]$. The inverse transform of (11.55) is

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} j\frac{\theta}{T_s}e^{j\theta n}d\theta = \frac{j}{2\pi T_s}\int_{-\pi}^{\pi}\theta e^{j\theta n}d\theta \tag{11.56}$$

Integrating by parts leads to

$$h(n) = \frac{j}{2\pi T_s}\left[\frac{\theta}{jn}e^{j\theta n}\Big|_{-\pi}^{\pi} - \int_{-\pi}^{\pi}\frac{1}{jn}e^{j\theta n}d\theta\right]$$

$$= \frac{j}{2\pi T_s}\left[\frac{\pi}{jn}e^{j\pi n} + \frac{\pi}{jn}e^{-j\theta n} - \left[\frac{2j}{jnjn}\frac{\left(e^{jn\pi} - e^{-jn\pi}\right)}{2j}\right]\right]$$

We can simplify more to get

$$h(n) = \frac{2j\pi}{2\pi T_s jn}\cos(n\pi) - \frac{2j(j)}{jnjn2\pi T_s}\sin(n\pi) = \frac{1}{nT_s}\cos(n\pi) - \frac{1}{T_s n^2 \pi}\sin(n\pi)$$

$$h(n) = \frac{1}{nT_s}\left[\cos(n\pi) - \frac{\sin(n\pi)}{n\pi}\right] = \frac{1}{nT_s}\left[(-1)^n - \delta(n)\right]$$

Finally we have

$$h(n) = \begin{cases} 0 & n = 0 \\ \dfrac{1}{nT_s}(-1)^n & \text{otherwise} \end{cases} \tag{11.57}$$

11.3.6 Design of Comb FIR Filters

With the linear phase property, we can design comb digital filters to remove unwanted frequencies and their harmonics. Let us say that we are interested in removing the frequencies at 50 Hz, 2(50) = 100 Hz and 3(50) = 150 Hz. In this case we will sample at 2(150) = 300 Hz to satisfy the Nyquist rate. In this case the three frequencies we want to remove are at $\theta = 2\pi f/f_s$. They are at the digital frequencies $2\pi 50/300 = \pi/3$, $2\pi 100/300 = 2\pi/3$ and $2\pi 150/300 = \pi$.

We know that a zero at $\theta = \pi/3$ on the unit circle will eliminate the digital frequency of $\pi/3$ and a zero at $2\pi/3$ will eliminate the digital frequency $2\pi/3$ and so on. We also know that zeros must appear in conjugates when they are complex. Therefore, to have symmetry in order to have the linear phase we will add another zero at $\theta = 0$. This zero will also eliminate any dc component in the input signal. The transfer function of this particular filter has the numerator $N(z)$ given by (remember to add the complex conjugate zeros)

$$N(z) = \left(z - e^{j\frac{\pi}{3}}\right)\left(z - e^{+j\frac{2\pi}{3}}\right)\left(z - e^{j\pi}\right)\left(z - e^{j0}\right)\left(z - e^{j\frac{4\pi}{3}}\right)\left(z - e^{j\frac{5\pi}{3}}\right)$$

Simplifying the above equation will yield in $N(z) = z^6 - 1$. With the denominator of $H(z)$ chosen as $D(z) = z^6$, we will have the transfer function

$$H(z) = \frac{z^6 - 1}{z^6} = 1 - z^{-6} \tag{11.58}$$

With $H(z) = Y(z)/X(z) = 1 - z^{-6}$, we have

$$Y(z) = X(z)\left(1 - z^{-6}\right)$$

Taking the inverse z-transform we get the difference equation

$$y(n) = x(n) - x(n-6) \tag{11.59}$$

The impulse response for the system in Equation (11.59) is obtained by setting $x(n) = \delta(n)$ and $h(n) = y(n)$ to get

$$h(n) = \delta(n) - \delta(n-6) \tag{11.60}$$

In Equation (11.58) we have six zeros equally spaced on the unit circle and six poles all at the origin of the unit circle. These six poles at the origin are needed to make the system causal. Having chosen $D(z) = 1$, we would get

$$y(n) = x(n+6) - x(n) \tag{11.61}$$

which is a noncausal filter. Thus, if we are interested in designing a comb filter with m zeros on the unit circle, the impulse response will be

$$h(n) = M\left[\delta(n) - \delta(n-m)\right] \tag{11.62}$$

The frequency response of Equation (11.62) is given as

$$H\left(e^{j\theta}\right) = M\left(1 - e^{-jm\theta}\right) \tag{11.63}$$

In Equation (11.63) the frequency response can be simplified and put in a nice form as

$$H\left(e^{j\theta}\right) = Me^{\frac{-j\theta m}{2}}\left(e^{\frac{j\theta m}{2}} - e^{\frac{-j\theta m}{2}}\right) = 2jMe^{\frac{-j\theta m}{2}}\sin\left(\frac{\theta m}{2}\right) \tag{11.64}$$

Equation (11.64) can be simplified further to get

$$H\left(e^{j\theta}\right) = 2M\sin\left(\frac{\theta m}{2}\right)e^{j\left(\frac{\pi}{2} - \frac{\theta m}{2}\right)} \tag{11.65}$$

The real part of $H(e^{j\theta})$ in Equation (11.65) is $2m\ \sin(\theta m/2)$ and the phase of the filter is linear with the slope of $-m/2$. The frequencies that will be removed using the FIR filter is Equation (11.65) are at

$$\theta = 0, \frac{2\pi}{m}, \frac{4\pi}{m}, K, \ldots, \pi. \tag{11.66}$$

11.3.7 Design of a Digital Shifter: The Hilbert Transform Filter

The Hilbert transform filter is an FIR digital filter with the transfer function

$$H\left(e^{j\theta}\right) = \begin{cases} -j & 0 < \theta < \pi \\ j & -\pi < \theta < 0 \end{cases} \tag{11.67}$$

The magnitude of $H(e^{j\theta})$ in Equation (11.67) is always unity. This filter will introduce a phase shift of –90° for $0 < \theta < \pi$ and a position 90° for $-\pi < \theta < 0$. To design the filter, we need a reliable and causal $h(n)$. To get this $h(n)$, let us first take the inverse transform of Equation (11.67) to get

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{0} je^{j\theta n}d\theta + \frac{1}{2}\int_{0}^{\pi}(-j)e^{j\theta n}d\theta \tag{11.68}$$

By simplifying Equation (11.68) we arrive at

$$h(n) = \frac{j}{2\pi nj}e^{j\theta n}\Big|_{-\pi}^{0} - \frac{j}{nj2\pi}e^{j\theta n}\Big|_{0}^{\pi} \tag{11.69}$$

Carrying out the calculation in Equation (11.69) we get

$$h(n) = \frac{1}{2\pi n}\left[2 - e^{-jn\pi} - e^{jn\pi}\right] = \frac{1}{n\pi}\left[1 - \cos(n\pi)\right]$$

Finally we get the impulse response $h(n)$ as

$$h(n) = \begin{cases} 0 & n \quad \text{even} \\ \dfrac{2}{n\pi} & n \quad \text{odd} \end{cases} \tag{11.70}$$

The filter in Equation (11.70) is not realizable. We can truncate $h(n)$ in Equation (11.70) with $N = 2m + 1$, where again m is an integer or an integer divided by 2, and then shift by m to make the filter realizable.

The Hilbert transform filter finds its importance in digital transmission. If the signal $x(n)$ to be transmitted is bandlimited to θ_b, then after being modulated it becomes $x(n)\cos(\theta_0 n)$. The bandwidth of the modulated signal is $2\theta_b$. Using the Hilbert transform filter, we can cut this bandwidth by 1/2.

11.4 From IIR to FIR Digital Filters: An Approximation

An FIR digital filter is a filter where the output $y(n)$ depends only on current and previous values of the input $x(n)$. An IIR digital filter is a filter where the ouput $y(n)$ depends on current and previous value for $x(n)$ and previous values of $y(n)$. Consider the difference equation for a digital FIR filter

$$y(n) = a_0 x(n) + a_1 x(n-1) + a_2 x(n-2) + \Lambda + a_{N-1} x\big(n-(N-1)\big) \qquad (11.71)$$

where the impulse response $h(n)$ has N sample points. We can z-transform the Equation in (11.71) to obtain the transfer function $H(z)$ as

$$H_{FIR}(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \Lambda + a_{N-1} z^{-(N-1)} \qquad (11.72)$$

and the constants a_0 through a_{N-1} are the impulse response sample values in $h(n)$. Now we can take an IIR filter transfer function $H_{IIR}(z) = N(z)/D(z)$ and use long division to get an equation for $H_{IIR}(z)$ similar to the one in Equation (11.72), which is an equation for an FIR filter. The equation for $H_{IIR}(z)$ that we will get as the result of the long division will have infinite number of terms. It is

$$H_{IIR}(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \Lambda + a_{N-1} z^{-(N-1)} + a_N z^{-N} + a_{N+1} z^{-(N+1)} + \Lambda \qquad (11.73)$$

If the filter represented by $H(z)$ in Equation (11.73) is stable then after some point in time, a_N will have negligible magnitude. In this case we can consider only the N terms (a_0 through a_N) and in this case Equation (11.72) for the FIR filter will be an approximation to Equation (11.73) for the IIR filter.

11.5 Frequency Sampling and FIR Filter Design

Given a mathematical expression for the magnitude transfer function of an FIR digital filter, we can use the equation

$$h(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H\left(e^{j\theta}\right) e^{j\theta n} d\theta \qquad (11.74)$$

to get the impulse response samples. These samples are infinite in length and can be truncated and delayed to get the realizable difference equation for implementation. However in some cases the integration in Equation (11.74) is not so easy to be carried out. In other cases we might have only

the samples of $H(e^{j\theta})$ equally spaced in the 2π period. We say in the 2π period because the magnitude of $H(e^{j\theta})$ is periodic of period 2π.

Once we collect the frequency samples of the desired $H(e^{j\theta})$, we can present these points to the inverse fast Fourier transform equation to get the impulse response $h(n)$. This is explained as in the following. If we sample $H(e^{j\theta})$ at $\theta = 2\pi k/N$, where N is the number of samples ($N = 2m + 1$), Equation (11.74) can be approximated to get the noncausal $h(n)$ as

$$h_{nc}(n) = \frac{1}{N}\sum_{k=-m}^{m} H\left(e^{j\frac{2\pi k}{N}}\right)e^{\frac{j2\pi k}{N}n} \qquad -m \le n \le m \tag{11.75}$$

Equation (11.75) is the inverse DFT (we will use fft for efficient calculation). Once we have $h(n)$ we can use a window function and delay the impulse response by m to get the realizable causal $h(n)$ as

$$h_c(n) = w(n)h_{nc}(n-m) \qquad n = 0, 1, KN-1 \tag{11.76}$$

11.6 FIR Digital Design Using MATLAB

There are several functions in MATLAB that help in the design of FIR digital filters. These functions are explained next.

11.6.1 Design Using Windows

The function `fir1` in MATLAB uses the Fourier series approach in designing lowpass, highpass, bandpass and bandstop FIR digital filters. The most general syntax is

```
h=fir1(n,f,'ftype',w,'noscale')
```

`h` is the vector of the filter coefficients of length $N + 1$. These are the impulse response coefficients.

The order of the filter is `n`. For the design of bandstop or highpass FIR filters, the order `n` must be even because for odd `n` we can have a zero at $z = -1$ on the unit circle (a zero at $\theta = \pi$). `f` is the normalized cut-off frequency/frequencies between 0 and 1. In case of bandpass or bandstop filter design, `f` is a vector that has two components: `f =[f1 f2]`. Both `f1` and `f2` should be normalized to lie between 0 and 1. `f1` and `f2` are the edge frequencies of the stopband or the passband. `ftype` between single quotes is the type of the filter to be used. For highpass we use `high` and for bandstop we use `stop`.

w is a vector of length n+1. It contains the specific window coefficients to be used. We use the following syntax for generating the window coefficients.

```
w=hanning(n+1);  w=hamming(n+1);
w=blackman(n+1);  w=ones(1,n+1);
```

The vector w that results from the Hanning, the Hamming and the Blackman windows is a column vector. The vector w that results from w=ones(1,n+1); is a row vector. noscale between single quotes makes a request to MATLAB not to use the default scaling used by MATLAB. With fir1(n,f,'ftype',w) MATLAB will use the default scaling. With fir1(n,f,'ftype') MATLAB will use the Hamming window. With fir1(n,f) MATLAB will use the Hamming window with a lowpass filter with the default scaling given by MATLAB.

11.6.2 Design Using Least-Squared Error

MATLAB uses the function firls to design any multiband FIR filter using the least-squares technique. The syntax is

```
h=firls(n,f,m,'ftype')
```

The order of the filter is n and h contains the n + 1 impulse response coefficients. ftype can be hilbert for a FIR digital Hilbert transform, or it can be differentiator for FIR digital differentiator. We can still use stop and high for ftype. f is a vector of frequency points that are at the edge of the band. These frequency points must be between 0 and 1, and they should include the 0 and the 1 point. These points are normalized. f is a vector that contains the magnitudes corresponding to the frequency points in the vector f. f and m should be of equal lengths. The lengths should be even.

11.6.3 Design Using the Equiripple Linear Phase

MATLAB uses the function remez that uses the algorithm developed by Park–McClellan. This function can be used to design FIR filters with equiripples, linear phase and multiband. The syntax for the remez function is

```
h=remez(n,f,m,'ftype')
```

h, n, f, m and ftype are explained in the function firls. The remez design is more popular and is the most used.

11.6.4 How to Obtain the Frequency Response

MATLAB uses the function freqz to produce the frequency response values. The syntax for FIR filters is

```
[H f]=freqz(h,1,fpts,fs);
```

The result of executing this function is a vector of frequency response values evaluated at the frequency vector `f`. `fpts` is the number of frequency points used by `freqz` and `fs` is the sampling frequency. `h` is the impulse response of the FIR filter and 1 indicates that the filter is nonrecursive ($D(z)$ in $H(z) = N(z)/D(z)$ is 1).

11.7 Some Insights

11.7.1 Comparison with IIR Filters

In Chapter 10 we considered designing IIR digital filters and we noticed that few coefficients were needed in order to achieve satisfactory design. Requiring few coefficients means we need to store few coefficients in the computer memory to implement the design. We have also noted that current outputs in IIR filters depended on previous outputs. Thus, in terms of implementation, we need fewer multiplications to obtain outputs at a particular point in time. This indicates efficiency and speed in the implementation of IIR digital filters. A problem with IIR filters is that we may encounter instability. This problem can be overcome by using the series method of implementation discussed in earlier chapters. The origin of this problem is that with higher order IIR filters the accuracy in manipulating the coefficients may become serious and this can cause instability. By implementing the IIR filter as blocks of series subsystems we can minimize this inaccuracy. Care must be taken when implementing the IIR filter design with series subsystems, especially when attempting to use complex poles and/or zeros in implementation of the subsystems. We need to remember that complex poles or zeros should occur in complex conjugates.

A more serious problem in designing IIR digital filters is the possibility of having nonlinear phase. This can happen especially with Butterworth, Chebysheb and elliptical IIR filters. In some cases, this nonlinearity can be tolerated and in other cases, it cannot be. This phase distortion can be reduced but not eliminated completely by using delay equalizers.

FIR digital filters require more coefficients than the IIR digital filters for the same design requirements. They are always stable and they have perfect linear phase characteristics.

11.7.2 The Different Methods Used in the FIR Filter Design

We have looked at different methods to design FIR digital filters. Windowing is one that is based on the Fourier series. The best window in the design is the one that gives the fewest ripples. To get fewer ripples in the stopband

or the passband we desire a window that falls gradually at the edges. The rectangular window is the worst in this sense but it gives the optimal and the best minimal error. Another method is the sampling frequency method. With the efficiency and the speed of the fft, this method is sometimes desirable. Its success depends on the number of frequency sampling points taken.

One of the main characteristics of FIR digital filters is the linear phase they produce. The best method discussed in this regard is the method by Parks and McClellan. This method is implemented by MATLAB and was used in the examples in this chapter. It works directly with the desired frequency response plot and can be applied to frequency responses with multiple bands with sharp and smooth transitions. This method is considered optimal since it produces the smallest error between desired and actual responses.

11.8 End of the Chapter Examples

EOCE 11.1

Consider the following difference equation

$$y(n) = x(n) + x(n-1) + x(n-2) + x(n-3) + x(n-4)$$

1. Is this a recursive or nonrecursive system?
2. Use the fft to plot the frequency response of the filter.

You can see that the impulse response for the given filter is

$$h(n) = \{1 \quad 1 \quad 1 \quad 1 \quad 1\}$$

If the coefficients in the above difference equation change we can have

$$h(n) = \{0 \;\; 1/2 \;\; 1 \;\; 1 \;\; 1 \;\; 1/2 \;\; 0\}$$

Use the `fft` to plot the frequency response of the new filter.

Solution

The output $y(n)$ depends only on current and previous values for $x(n)$. Thus the filter is nonrecursive. The plot for the magnitude frequency response is obtained using the following MATLAB script for the two impulse responses.

```
h1n=[1 1 1 1 1]; h2n=[0 1/2 1 1 1 1/2 0];
X1=fft(h1n,1024);X2=fft(h2n,1024);
```

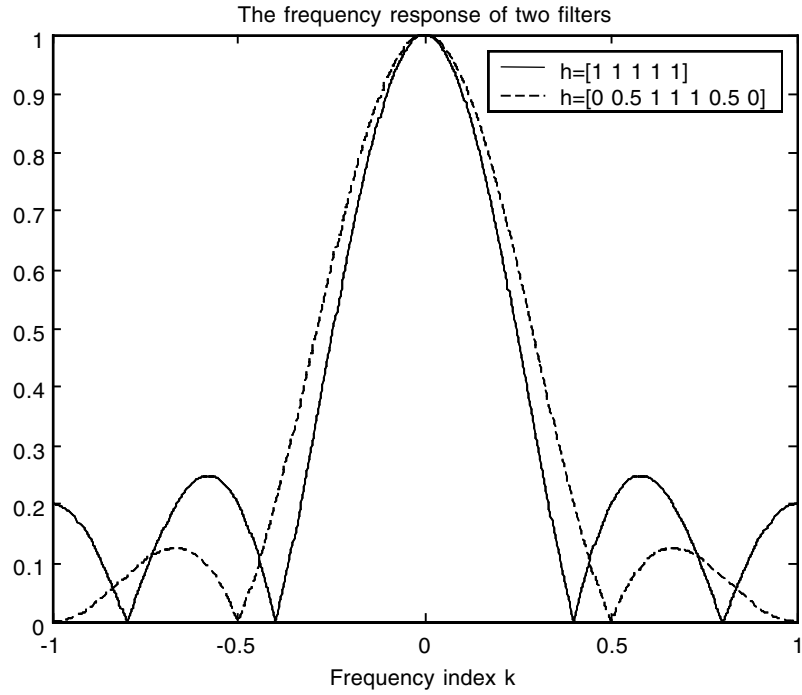

FIGURE 11.8 Plots for EOCE 11.1.

```
N=1024; n1=1:ceil((N+1)/2); n2=ceil((N+1)/2)+1:N;%ceil(2.1)=3
k=1/(N/2)*(ceil(-(N-1)/2):ceil((N-1)/2));
X1=[X1(n2) X1(n1)];X2=[X2(n2) X2(n1)];
plot(k,abs(X1)/max(abs(X1))); hold on;
plot(k,abs(X2)/max(abs(X2)),'--');
legend('h=[1 1 1 1 1]','h=[0 0.5 1 1 1 0.5 0]',0);
title('The frequency response of two filters');
xlabel('Frequency index k');
```

The plots are shown in Figure 11.8. We can see that both filters are lowpass and the second is better since its passband is wider and the magnitude of the lobes is less.

EOCE 11.2

Plot the magnitude frequency response of the rectangular window, the Hanning window, the Hamming window and the Blackman window.

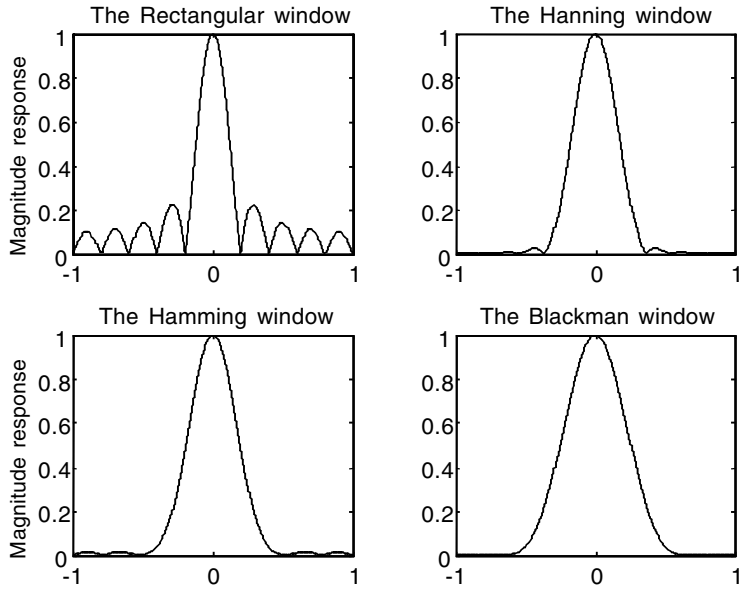

FIGURE 11.9 Plots for EOCE 11.2.

Solution

We will write the following MATLAB script to plot the responses.

```
x1=ones(1,10);x2=hanning(10); x3=hamming(10); x4=blackman(10);
X1=fft(x1,1024);X2=fft(x2,1024);X3=fft(x3,1024);X4=fft(x4,10
 24);
N=1024; n1=1:ceil((N+1)/2); n2=ceil((N+1)/2)+1:N;%ceil(2.1)=3
k=1/(N/2)*(ceil(-(N-1)/2):ceil((N-1)/2));
X1=[X1(n2) X1(n1)];X2=[X2(n2)' X2(n1)'];
X3=[X3(n2)' X3(n1)'];X4=[X4(n2)' X4(n1)'];
subplot(2,2,1);plot(k,abs(X1)/max(abs(X1)));
title('The Rectangular window');ylabel('Magnitude response');
subplot(2,2,2);plot(k,abs(X2)/max(abs(X2)));
title('The Hanning window');
subplot(2,2,3);plot(k,abs(X3)/max(abs(X3)));
title('The Hamming window');ylabel('Magnitude response');
subplot(2,2,4);plot(k,abs(X4)/max(abs(X4)));
title('The The Blackman window');
```

The plots are shown in Figure 11.9. You can see that the Blackman window allows the maximum main lobe width and has negligible ripples at the side lobes.

EOCE 11.3

Consider the following continuous signal

$$x(t) = 2 + \cos(10t)$$

Design a lowpass digital filter to filter out the sinusoidal term. Use different windows in the design. Find the coefficients of the filter, $h(n)$, using the rectangular window.

Solution

When we sample $x(t)$ we should sample at a frequency that is at least twice the maximum frequency in the signal. The maximum frequency in $x(t)$ is $f_m = 10/2\pi = 1.59$ Hz. Thus f_s should be at least 2(1.59) = 3.18 Hz. This sampling frequency corresponds to the sampling period 1/3.18 = .31 sec. Let us take $T_s = .1$ as our sampling period. This value of T_s still prevents aliasing.

The analogue frequency that we would like to suppress is the $w = 10$ rad/sec. An analogue lowpass filter of $w_c = 5$ rad/sec will suppress the cos(10t) term. This $w_c = 5$ rad/sec analogue frequency corresponds to the digital frequency $\theta_c = w_c(T_s) = 5(.1) = .5$. Thus, we will design our lowpass digital filter with $\theta_c = .5$ as the cutoff frequency. The following MATLAB script will be used for the design and plotting.

```
clf
%we will use N=11 then 21
N=11; %filter order
m=(N-1)/2;
thetac=.5;
n=0:N-1;
h=sin(thetac*(n-m+eps))./(pi*(n-m+eps));
%eps is used to avoid dividing by zero
h1=h.*(ones(1,N));%we window the filter coefficients
h2=h.*(hanning(N))';
n=0:100;
Ts=.1;
xn1=2;%This signal should pass through the filter
xn2=cos(10*n*Ts);%This signal should not pass
xn=xn1+xn2;
t=0:.1:10;
xt=cos(10*t)+2;
y1=filter(h1,1,xn);
y2=filter(h2,1, xn);
subplot(3,1,1); plot(t,xt);
```

```
title('cos(10*t)+2 before filtering');
[H1 f1]=freqz(h1,1,100); subplot(3,1,2);plot(f1/pi,abs(H1));
 hold on;
[H2 f2]=freqz(h2,1,100);plot(f2/pi,abs(H2),'*');
legend('Rectangular','Hanning',0);
ylabel('Filter: Order 11');
subplot(3,1,3); plot(n*Ts,y1);
title('The signal after cos(10*t) term is removed: Filter
 order is 11');
xlabel('Time(sec)'); hold on; plot(n*Ts, y2, '*');
legend('Rectangular','Hanning',0);
```

The plots are shown in Figures 11.10 and 11.11 for $N = 11$ and $N = 21$. We can see that when $x(n)$ is present as an input to the lowpass filter only the dc component is approximately passing. Thus, the design using the rectangular window works.

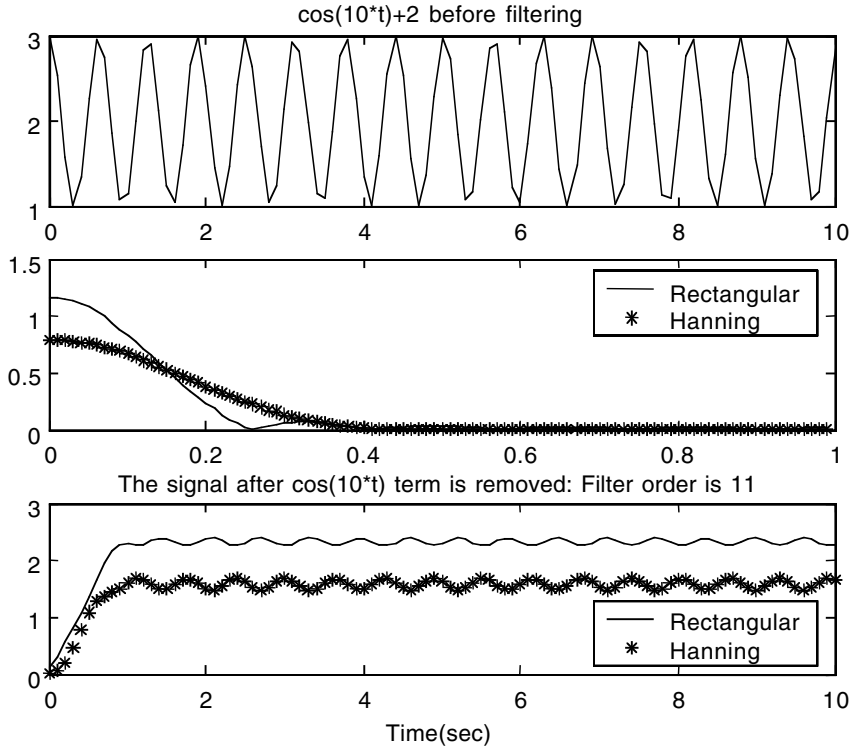

FIGURE 11.10 Plots for EOCE 11.3.

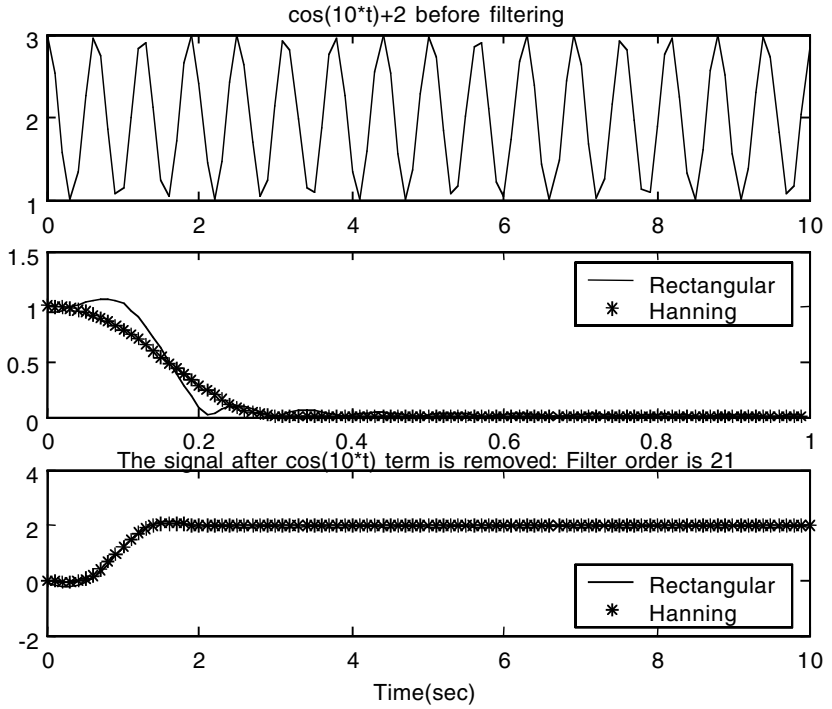

FIGURE 11.11 Plots for EOCE 11.3.

The coefficients of the filter are obtained by removing the semicolon from the end of the statement

```
h1=h.*(ones(1,N));
```

and then running the script. The result is

```
h =
Columns 1 through 9
0.0381  0.0724  0.1058  0.1339  0.1526  0.1592  0.1526  0.1339  0.1058
Columns 10 through 11
0.724 0.0381
```

with $h(0) = 0.0381$, $h(1) = 0.0724$, $h(2) = 0.1339$, and $h(10) = 0.0381$.

EOCE 11.4

Consider the analogue signal

$$x(t) = \sin(t) + \sin(10t)$$

We are interested in suppressing the $\sin(t)$ term. Design a digital highpass filter to accomplish that. Use different windows in the design.

Solution

With $f_m = 10/2\pi$ Hz, we will take our sampling period as $T_s = .1$ as we did in EOCE 11.3. An analogue highpass filter with analogue cutoff frequency $w_c = 5$ rad/sec will suppress the $\sin(t)$ term and pass the $\sin(10t)$ term. $w_c = 5$ corresponds to $T_s\,(w_c) = .1(5) = .5$. Thus, we will design a highpass digital filter with cutoff frequency of $\theta_c = .5$ away from $\theta = 0$. The following MATLAB script will be used to design the filter and produce the plots.

```
clf
%we will use N=11 then 21
N=11; %filter order
m=(N-1)/2;
theta=.5; % away from the theta = 0 point
thetac=pi-theta;%The cutoff frequency of the highpass filter
 away from theta=pi
%The thetac value is used in equation 10.28
n=0:N-1;
mm=n-m+eps; %To avoid division by zero
h=cos(mm*pi).*(sin((thetac)*(mm))./(pi*(mm)));
h1=h.*(hamming(N))';
h2=h.*(blackman(N))';
n=0:50;
Ts=.1;
xn=sin(n*Ts)+sin(10*n*Ts);
t=0:.01:5;
xt=sin(t)+sin(10*t);
y1=filter(h1,1,xn);
y2=filter(h2,1,xn);
subplot(3,1,1); plot(t,xt);
title('The original signal x(t)=sin(t)+sin(10t) before
 filtering');
[H1 f1]=freqz(h1,1,100); subplot(3,1,2);plot(f1/pi,abs(H1));
 hold on;
[H2 f2]=freqz(h2,1,100);plot(f2/pi,abs(H2),'*');
legend('Hamming','Blackman',0);
ylabel('The filter: N=11');
```

```
subplot(3,1,3); plot(n*Ts,y1);
title('The signal after the sin(t) term is removed: Filter
 order is 11');
xlabel('Time(sec)'); hold on; plot(n*Ts, y2, '*');
legend('Hamming','Blackman',0);
```

The plots are shown in Figure 11.12 and Figure 11.13. We can see that when $x(n)$ is present as an input to the highpass filter, the output is close to the $\sin(t)$ term and the best approximation is achieved with $N = 21$.

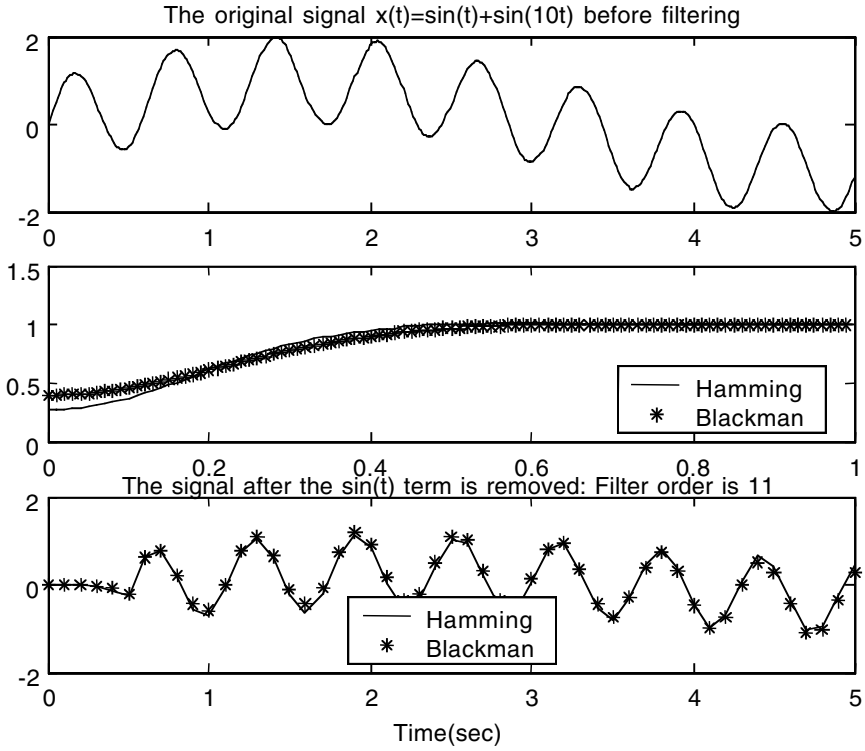

FIGURE 11.12 Plots for EOCE 11.4.

EOCE 11.5

Noise comes always at high frequencies. A differentiator amplifies the input signal more and more as the frequency of the signal increases. If we are processing a signal that contains noise plus low frequency components and a differentiator is used in the process, we would like to limit the differentiation to a certain range to prevent excessive amplification of the noise components. Let us say that we are interested in the range $-\theta_c \leq \theta \leq \theta_c$. Give the differentiator impulse response.

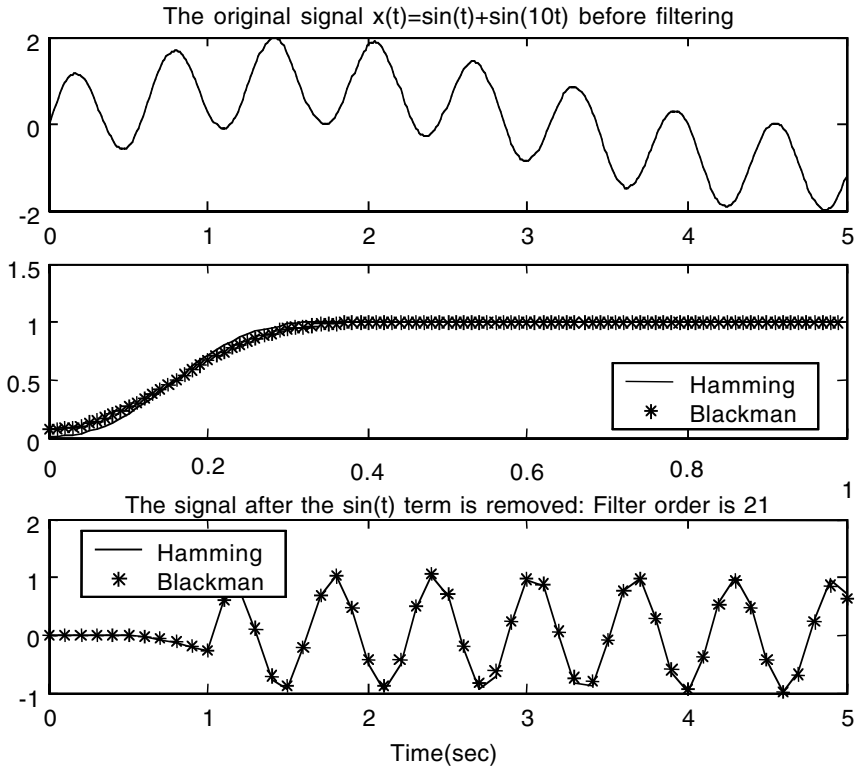

FIGURE 11.13 Plots for EOCE 11.4.

Solution

We can start with Equation (11.56) with $-\theta_c = -\pi$ and $\theta_c = \pi$ to get

$$h(n) = \frac{1}{2\pi}\int_{-\theta_c}^{\theta_c} j\frac{\theta}{T_s}e^{j\theta n}d\theta \tag{11.77}$$

We will simplify the integration in Equation (11.77) as we did in Section 11.3.5 to get

$$h(n) = \frac{j}{2\pi T_s}\left[\frac{\theta_c}{jn}e^{jn\theta_c} + \frac{\theta_c}{jn}e^{-j\theta_c n} - \left[\frac{2j}{jnjn}\left(e^{jn\theta_c} - e^{-jn\theta_c}\right)\right]\right]$$

$$h(n) = \frac{2j\theta_c}{2\pi T_s jn}\cos\left(n\theta_c\right) - \frac{2j(j)}{2\pi T_s jnjn}\sin\left(n\theta_c\right)$$

Finally we have

$$h(n)=\begin{cases}\dfrac{\theta_c}{\pi n T_s}\cos(n\theta_c)-\dfrac{1}{T_s n^2\pi}\sin(n\theta_c) & |n|=1,2,3\ldots\\ 0 & n=0\end{cases} \tag{11.78}$$

With $\theta_c = \pi/2$ and $T_s = 1$ and by considering the order of the filter to be $N = 21$ with m, the delay, of $(N - 1)/2 = 10$, we have the realizable digital differentiator as

$$h(n)=\frac{1}{2(n-m)}\cos\left((n-m)\frac{\pi}{2}\right)-\frac{1}{(n-m)^2\pi}\sin\left((n-m)\frac{\pi}{2}\right)$$

We will use MATLAB to obtain the frequency response using the rectangular and the Hamming windows. The MATLAB script follows.

```
clf
for N=11:90:101%Filter order 11 and 101
m=(N-1)/2;
thetac=pi/2;
n=0:N-1;
mm=n-m+eps; %To avoid dividing by zero
h=(thetac*cos(mm*thetac))./(mm*pi) -
 sin(mm*thetac)./(pi*(mm.^2));
h1=h.*ones(1,N);
h2=h.*hamming(N)';
[H1 f1]=freqz(h1,1,200); plot(f1/pi,abs(H1));hold on;
[H2 f2]=freqz(h2,1,200); plot(f2/pi,abs(H2),'*');
end
legend('Rectangular: N=11 and 101','Hamming: N=11 and 101',0);
title('Magnitude response for a differentiator');
xlabel('radian frequency in pi units')
gtext('Cutoff frequency = pi/2');
```

The plots are in Figure 11.14.

EOCE 11.6

We are interested in removing the unwanted frequencies 60 Hz, 120 Hz and 180 Hz from the continuous signal $x(t)$. Design a digital filter that will eliminate these frequencies. Assume that the highest frequency in $x(t)$ is 200 Hz.

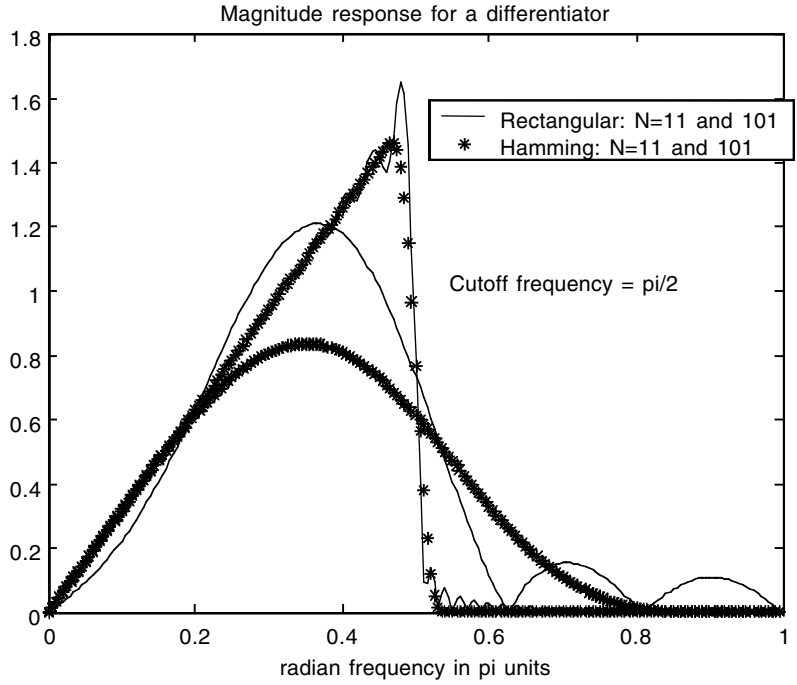

FIGURE 11.14 Plots for EOCE 11.5.

1. Consider the input to the comb filter as

$$x(t) = \cos(2\pi 60t) + \cos(2\pi 120t) + \cos(2\pi 360t)$$

   and show the output of the filter.
2. Consider next the input given above plus $\cos(100t)$. What would be the output of this filter?

Solution

Since the highest frequency in $x(t)$ is 200 Hz, we will sample at least at f_s = 2(200) = 400 Hz. Let us sample at f_s = 1200 Hz. The digital frequencies corresponding to the analogue frequencies of 60, 120 and 180 are calculated in the following way. For f = 60 Hz, $\theta = 2\pi f/f_s = 2\pi(60)/1200 = 0.1\pi$.

Thus the separation between the zeros on the unit circle is 0.1π . We will have m zeros with $2\pi/m = 0.1\pi$. This gives $m = 20$. Thus the transfer function of the comb filter is

$$H(z) = 1 - z^{-20}$$

and the impulse response is

$$h(n) = \delta(n) - \delta(n - 20)$$

We will write the following MATLAB script to do the design and produce the plots:

```
t=0:.001:.1;
x=cos(2*pi*60*t)+cos(2*pi*120*t)+cos(2*pi*360*t);
fs=1200; Ts=1/fs;
n=0:120;
nTs=n*Ts;
xn=cos(2*pi*60*nTs)+cos(2*pi*120*nTs)+cos(2*pi*360*nTs);
subplot(3,1,1);plot(t,x);
title('The original signal');
h=[1 zeros(1,19) -1];
[H,f]=freqz(h,1);
subplot(3,1,2);plot(f/pi,abs(H)/max(abs(H)));
title('Magnitude response of a comb filter:20 zeros');
y=filter(h,1,xn);
subplot(3,1,3); plot(nTs,y);
title('The output of the filter'); xlabel('Time(sec)');
```

The plots are shown in Figure 11.15 and Figure 11.16. In Figure 11.5 and after some delay, the output is zero as expected since the frequencies 60, 120 and

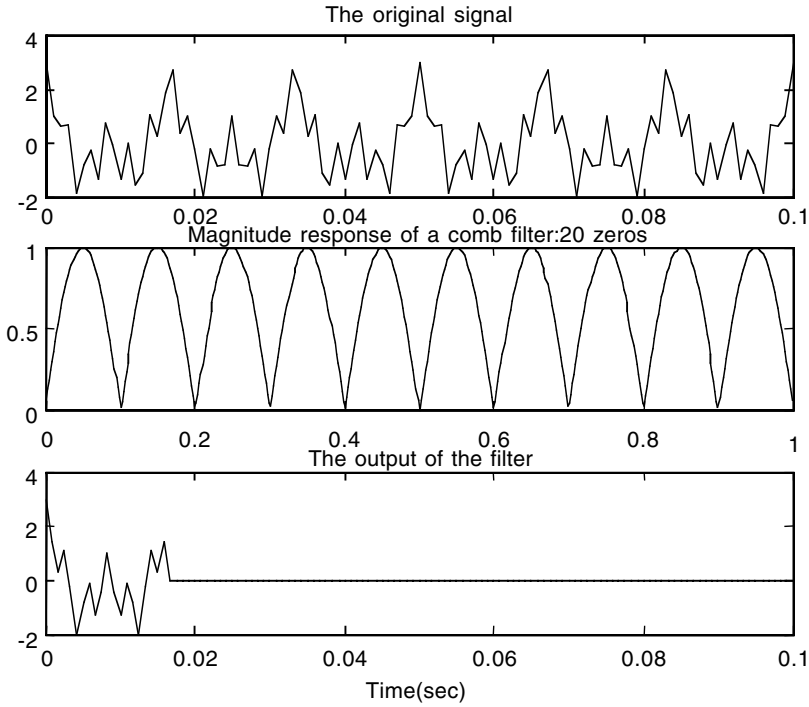

FIGURE 11.15

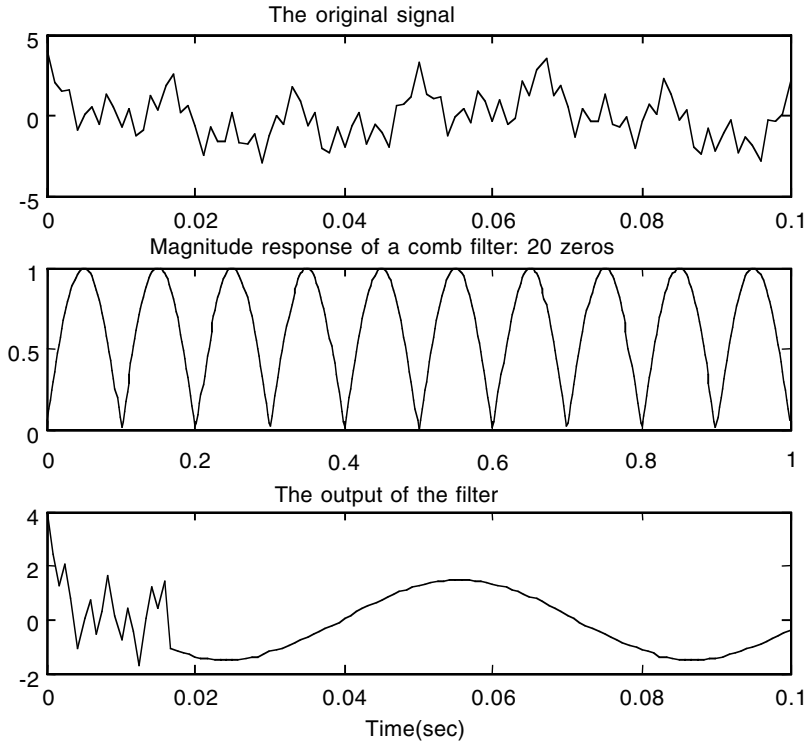

FIGURE 11.16 Plots for EOCE 11.6.

360 are notched out. In Figure 11.16 only the term cos(100*t*) is present after about 0.02 sec. The plots in Figure 11.16 are obtained using the above MATLAB script with the two expressions:

```
x=cos(2*pi*60*t)+cos(2*pi*120*t)+cos(2*pi*360*t)+cos(100*t);
xn=cos(2*pi*60*nTs)+cos(2*pi*120*nTs)+cos(2*pi*360*nTs)+cos
  (nTs*100);
```

EOCE 11.7

Consider the following IIR digital filter

$$H(z) = \frac{z^2 - 1}{z^2 - \frac{3}{4}z + \frac{1}{8}}$$

Derive an FIR digital filter to approximate the digital IIR filter. Comment on the results.

Solution

The IIR digital filter is a stable one. Thus we expect the result of the long division to converge at some point. Carrying out the long division will result in

$$H(z) = 1 + \frac{3}{4}z^{-1} + \left(-9/16\right)z^{-2} + \Lambda$$

The first three samples of the FIR impulse response are 1, +3/4 and –9/16. It is much easier to get the remaining terms using MATLAB. To do that we write the following script that will plot the magnitude response of the IIR filter and its FIR approximation

```
%The IIR filter
nIIR=[1 0 -1]; dIIR=[1 -3/4 1/8];
%and its frequency response is
[HIIR fIIR]=freqz(nIIR,dIIR,100);%100 points calculated
plot(fIIR/pi, abs(HIIR)/max(abs(HIIR))); hold on
%next we will use the dimpulse function to get the first 50
%terms in the long division process
terms=dimpulse(nIIR,dIIR,50);
%Let us take 4 point in approximating the FIR filter
nFIR=terms(1:4);%tenth order FIR
[HFIR fFIR]=freqz(nFIR,1,100);%100 points calculated
plot(fFIR/pi, abs(HFIR)/max(abs(HFIR)),'*');
%Let us take 10 point in approximating the FIR filter
nFIR=terms(1:10);%tenth order FIR
[HFIR fFIR]=freqz(nFIR,1,100);%100 points calculated
plot(fFIR/pi, abs(HFIR)/max(abs(HFIR)),'o');
xlabel('Frequency in pi units');title('FIR Approximation to
 an IIR filter');
legend('The IIR Filter','FIR approximation: N=5','FIR
 approximation:N=10',0);
```

The plots are shown in Figure 11.17. You can see that as the number of terms is $h(n)$ for the FIR filter increases, the better is the approximation.

EOCE 11.8

Consider the digital filter magnitude responses shown in Figure 11.18. Use the frequency sampling method to design an FIR lowpass filter to approximate the one in Figure 11.18.

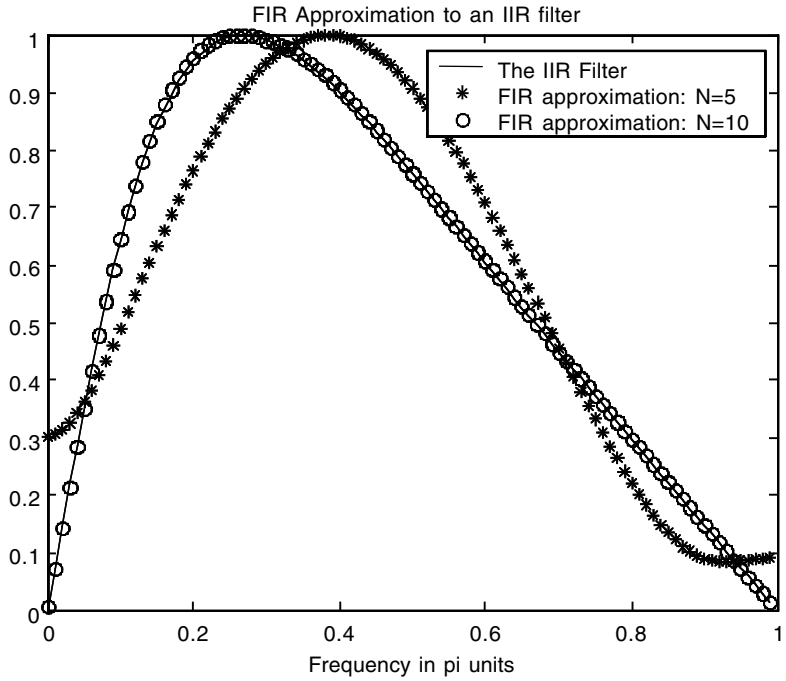

FIGURE 11.17 Plots for EOCE 11.7.

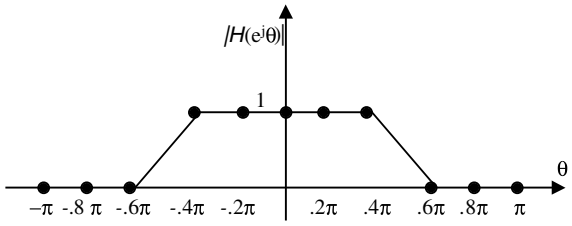

FIGURE 11.18 Filter for EOCE 11.8.

Solution

Recall that the fft equation takes data points in the range $0 \leq n \leq N-1$. We know that the discrete Fourier transform is periodic with period 2π. Thus we can redraw the filter in Figure 10.18 as Figure 11.19. We can sample Figure 11.19 and get the 11 samples in the interval $[0\ \ 2\pi]$. Notice that the sample at the 2π point is not taken because the inverse fft starts its cycle at 2π. With 11 samples we write the following MATLAB script to plot the approximation to the filter in Figure 11.18.

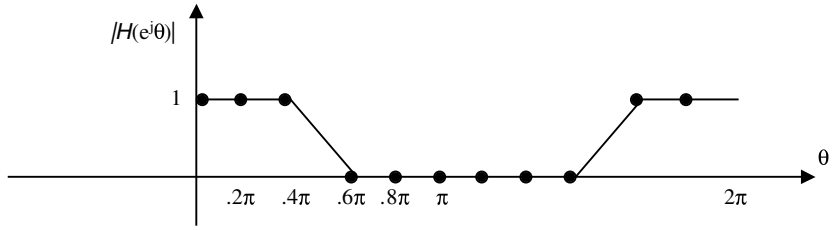

FIGURE 11.19 Filter for EOCE 11.8.

```
n=11;%11 frequency samples
Hsamp=[1 1 1 0 0 0 0 0 0 1 1];
h=ifft(Hsamp);
n1=(n-(n-1)/2+1):n; %shifting the last five samples to the
 front of h
n2=1:n1-1;
h=[h(n1) h(n2)] %The realizable filter coefficients
[H,f]=freqz(h,1,100);%100 frequency points only in [0 pi];
plot(f/pi,abs(H)); hold on;
h=h.*hamming(n)';
[H,f]=freqz(h,1,100);
plot(f/pi,abs(H),'*');
legend('Using Rectangular window','Using Hamming window',0);
title('Approximation using frequency sampling: 11 samples');
xlabel('Frequency in pi units');
```

The plot is shown in Figure 11.20. The impulse response is obtained from the script as

```
h =
Columns 1 through 9
0.0694  –0.0540  –0.1094  0.0474  0.3194  0.4545  0.3194  0.0474  –0.1094
Columns 10 through 11
–0.0540  0.0694
```

and the realizable transfer function is

$$H(z) = 0.0694\left(1 + z^{-10}\right) - 0.054\left(z^{-1} + z^{-9}\right) - 0.1094\left(z^{-2} + z^{-8}\right) + 0.0474\left(z^{-3} + z^{-7}\right)$$

$$+ 0.3194\left(z^{-4} + z^{-6}\right) + .4545z^{-5}$$

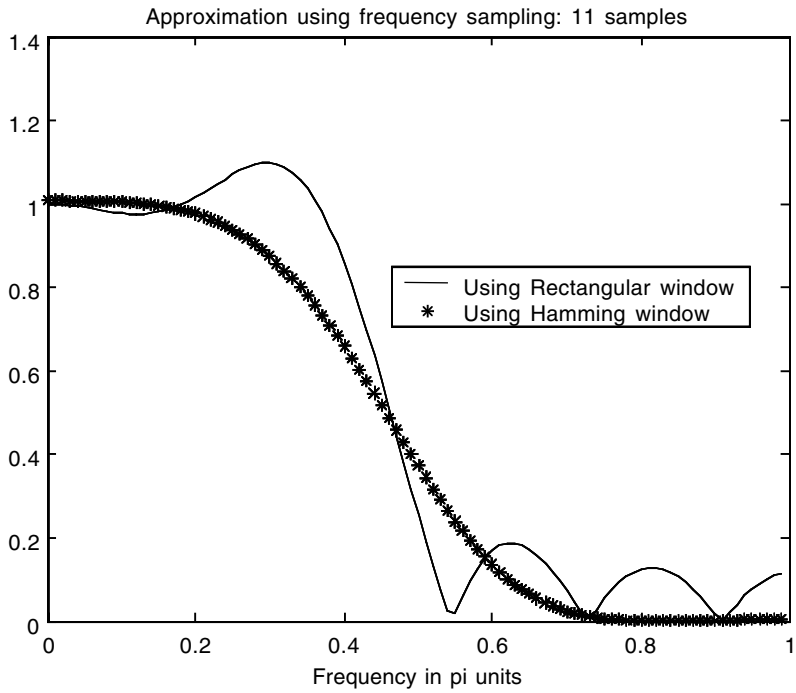

FIGURE 11.20 Plots for EOCE 11.8.

EOCE 11.9

We are interested in removing the frequencies that are higher than 10 kHz from the analogue signal $x(t)$. Design a digital filter to accomplish that. Use a sampling frequency of 100 kHz.

Solution

The maximum digital frequency that the digital filter will have is $\theta_{max} = \pi$. This corresponds to the analogue frequency $f_s/2 = 50$ kHz. This digital filter, therefore, will try to suppress analogue frequencies between 10 and 50 kHz. This corresponds to the digital frequencies

$$\frac{2\pi f}{f_s} \leq \theta \leq \frac{2\pi f_s/2}{f_s}$$

or $0.2\pi \leq \theta \leq \pi$.

The cutoff frequency of the lowpass digital filter is taken as $\theta_c = 0.1\pi$ so as to suppress the digital frequencies starting at $\theta = 0.2\pi$. The following MATLAB script will finish the design and draw the frequency response of the digital filter. It will also show the filter output when the input is

$$x(t) = \sin(2\pi 50000t) + \cos(2\pi 5000t)$$

```
clf
%we will use N=11 then 21
N=21; %filter order
m=(N-1)/2;
thetac=.1*pi;
n=0:N-1;
h=sin(thetac*(n-m+eps))./(pi*(n-m+eps));
%eps is used to avoid dividing by zero
h1=h.*(ones(1,N));%we window the filter coefficients
h2=h.*(hanning(N))';
n=0:100;
Ts=1/100000;
xn1=sin(2*pi*50000*n*Ts);%This signal should pass through the
 filter
xn2=cos(2*pi*5000*n*Ts);%This signal should not pass
xn=xn1+xn2;
t=0:.000001:1/1000;
xt=sin(2*pi*50000*t)+cos(2*pi*5000*t);
y1=filter(h1,1,xn);
y2=filter(h2,1, xn);
subplot(3,1,1); plot(t,xt);
title('Input: sin(2*pi*50000*t)+cos(2*pi*5000*t) before
 filtering');
[H1 f1]=freqz(h1,1,100); subplot(3,1,2);plot(f1/pi,abs(H1));
 hold on;
[H2 f2]=freqz(h2,1,100);plot(f2/pi,abs(H2),'*');
legend('Rectangular','Hanning',0);
ylabel('Filter: Order 21');
subplot(3,1,3); plot(n*Ts,y1);
title('The signal after sin(2*pi*50000*t) term is removed');
xlabel('Time(sec)'); hold on; plot(n*Ts, y2, '*');
legend('Rectangular','Hanning',0);
```

The plot is shown in Figure 11.21. The period of the output in Figure 11.21 is $1/5000 \cong .2x10^{-3}$ sec. This indicates that the filter is working well.

EOCE 11.10

We wish to pass the range of frequency 400 Hz $\leq f \leq$ 800 Hz in the analogue signal $x(t)$. Suppose that the highest frequency is $x(t)$ in 800 Hz. Design a bandpass digital filter to do just that.

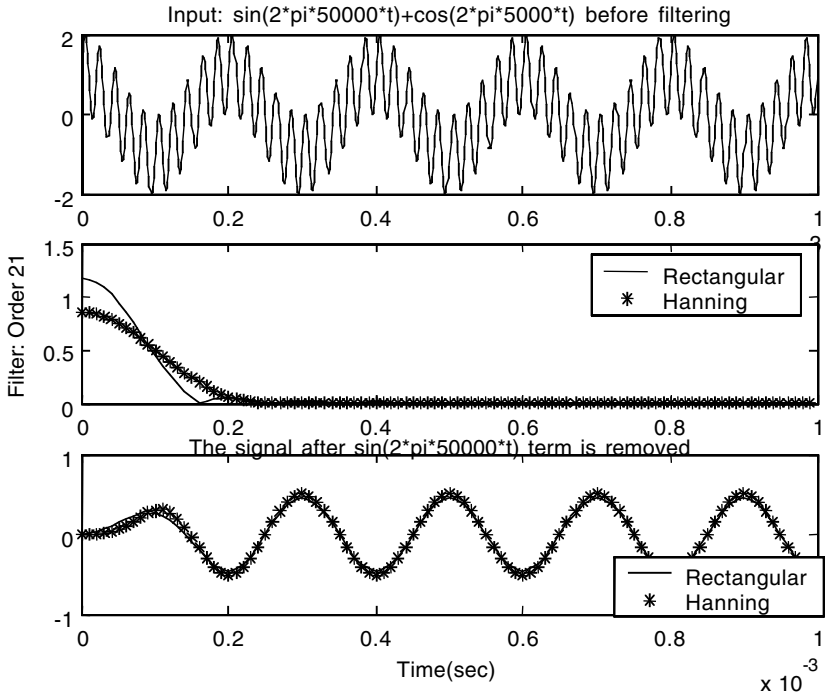

FIGURE 11.21 Plots for EOCE 11.9.

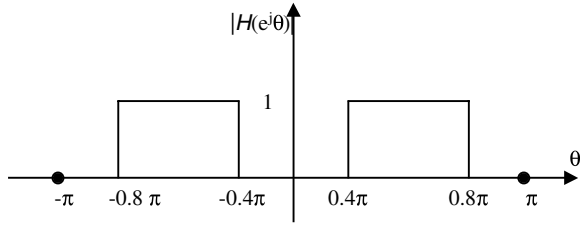

FIGURE 11.22 Filter for EOCE 11.10.

Solution

The sampling frequency is at least 2(800) = 1600Hz. Let us take f_s as 2000 Hz. In this case, the band of digital frequencies will be $\theta_l = 2\pi 400/2000 = 0.4\pi$ and $\theta_u = 2\pi 800/2000 = 1600\pi/2000 = .8\pi$. The proposed digital filter is shown in Figure 11.22.

We will design the filter in Figure 11.22 using the Fourier series method with the rectangular and Hamming windows and using the frequency sampling method. As in EOCE 11.8, we will sample in the interval [0 2π]. The following MATLAB script will be used to do that. The plots are seen in Figure 11.23.

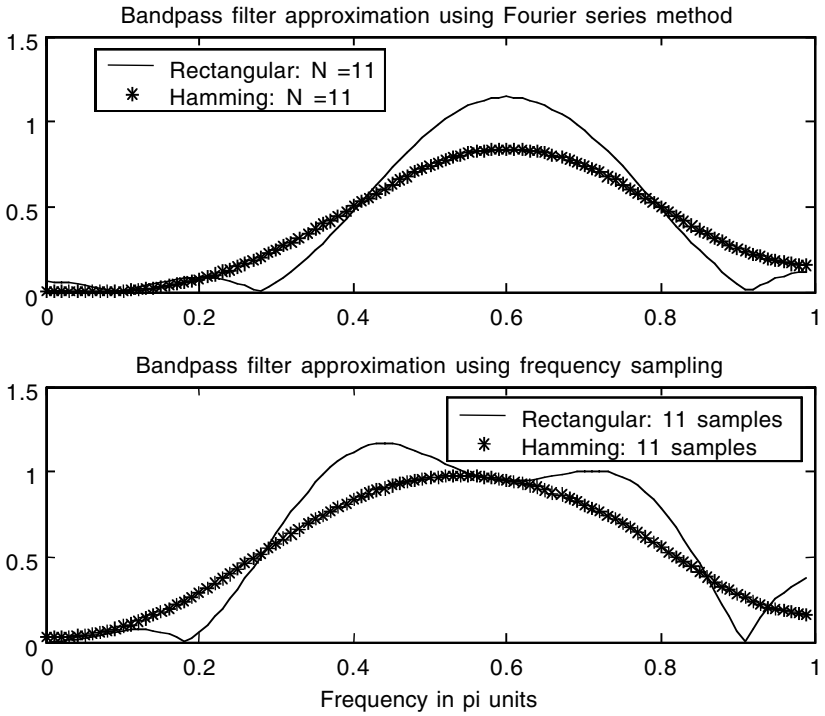

FIGURE 11.23 Plots for EOCE 11.10.

```
clf
%we will use N=11 then 21
N=11; %filter order
m=(N-1)/2;
thetal=.4*pi; %lower cutoff frequency
thetau=.8*pi; %upper cutoff frequency
n=0:N-1;
h=(sin(thetau*(n-m+eps))-sin(thetal*(n-m+eps))).(pi*(n-
 m+eps));
%eps is used to avoid dividing by zero
h1=h.*(ones(1,N));%we window the filter coefficients
h2=h.*(hamming(N))';
[H1 f1]=freqz(h1,1,100); subplot(2,1,1);plot(f1/pi,abs(H1));
 hold on;
[H2 f2]=freqz(h2,1,100);plot(f2/pi,abs(H2),'*');
title('Bandpass filter approximation using Fourier series
 method');
legend('Rectangular: N=11','Hamming: N=11',0);
%Frequency sampling method
%n=21;
```

```
n=11
Hsamp=[0 0 1 1 1 0 0 1 1 1 0];
%Hsamp=[0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 ];
h=ifft(Hsamp);
n1=(n-(n-1)/2+1):n; %shifting the last five samples to the
 front of h
n2=1:n1-1;
h=[h(n1) h(n2)] %The realizable filter coefficients
[H,f]=freqz(h,1,100);%100 frequency points only in [0 pi];
subplot(2,1,2);plot(f/pi,abs(H)); hold on;
h=h.*hamming(n)';
[H,f]=freqz(h,1,100);
plot(f/pi,abs(H),'*');hold on;
xlabel('Frequency in pi units');
legend('Rectangular: 11 samples','Hamming: 11 samples',0);
title('Bandpass filter approximation using frequency
 sampling');
```

EOCE 11.11

Use MATLAB to design a lowpass filter of order 21 that will pass frequencies from 0 to 100 Hz with unity magnitude. We expect zero magnitude for other frequencies. Assume that the highest frequency coming to this filter is 150 Hz. Use the functions `fir1, firls` and `remez` for the design. Use the Hamming window.

Solution

Using the function `fir1` we need to use the form `h = fir1 (n, f)`. The Hamming window is the default window and since no filter type is specified, it will use the lowpass. `n` is 21 and f is obtained as in the following. The highest frequency in the incoming analogue signal is 150. This means a sampling frequency of at least 300 Hz. Let us consider f_s = 1000Hz. The cutoff frequency of the analogue filter is 100 Hz. This corresponds to the normalized digital frequency according to the relation

$$f = \frac{f_c}{f_s/2}$$

or f = 100/500 = 0.2. This value can also be obtained by finding first the corresponding digital frequency: $\theta_c = T_s w_c = 1/1000\ (2\pi(100)) = .2\pi$. We now normalize by dividing by π to get 0.2 as the normalized digital frequency.

Using the function `firls`, we will use the syntax `h = firls (n, f, m)` for a lowpass FIR filter. n is 21. We will take two bands of frequencies, [0 .2]

and [.3 1], with the corresponding magnitude pairs, [1 1] and [0 0]. Next we will use the `remez` function with the same data that was used with the other two functions. The MATLAB script is given next.

```
%Design using the fir1 function
h=fir1(21,0.2);
[H f]=freqz(h,1,100);
plot(f/pi, abs(H)); hold on;
title('Lowpass filter with normalized cutoff frequency of 0.2
 and N=21');
%Design using the firls function
n=21;
f=[0 0.2 0.3 1]; m=[1 1 0 0];
h=firls(n,f,m);
[H f]=freqz(h,1,100);
plot(f/pi, abs(H),'*');
%Design using the remez function
f=[0 0.2 0.3 1]; m=[1 1 0 0];
h=remez(n,f,m);
[H f]=freqz(h,1,100);
plot(f/pi, abs(H),'d');
legend('fir1 function','firls function','remez function',0);
xlabel('Frequency in pi units');
```

The plot is shown in Figure 11.24. You can see that the MATLAB function `firls` produced the best approximation. It produced wider band and fewer ripples.

EOCE 11.12

We are interested in designing a bandstop FIR filter with edge frequencies at 0.3π and 0.6π. The gain in the passbands is unity and in the stopband is zero. Use the functions `fir1, firls` and `remez` for the design. Use the Hamming window.

Solution

We will use `n = 22` since we are designing a bandstop filter. In using the `firls` function we will take the frequency points at 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.9, and 1. This is an even number of points as required by `firls`. The frequency vector is f = [0 .1 .2 .3 .4 .5 .6 .7 .9 1], and the corresponding m vector is m = [1 1 1 0 0 0 0 1 1 1]. The MATLAB script for the complete design and for producing the plots is given next.

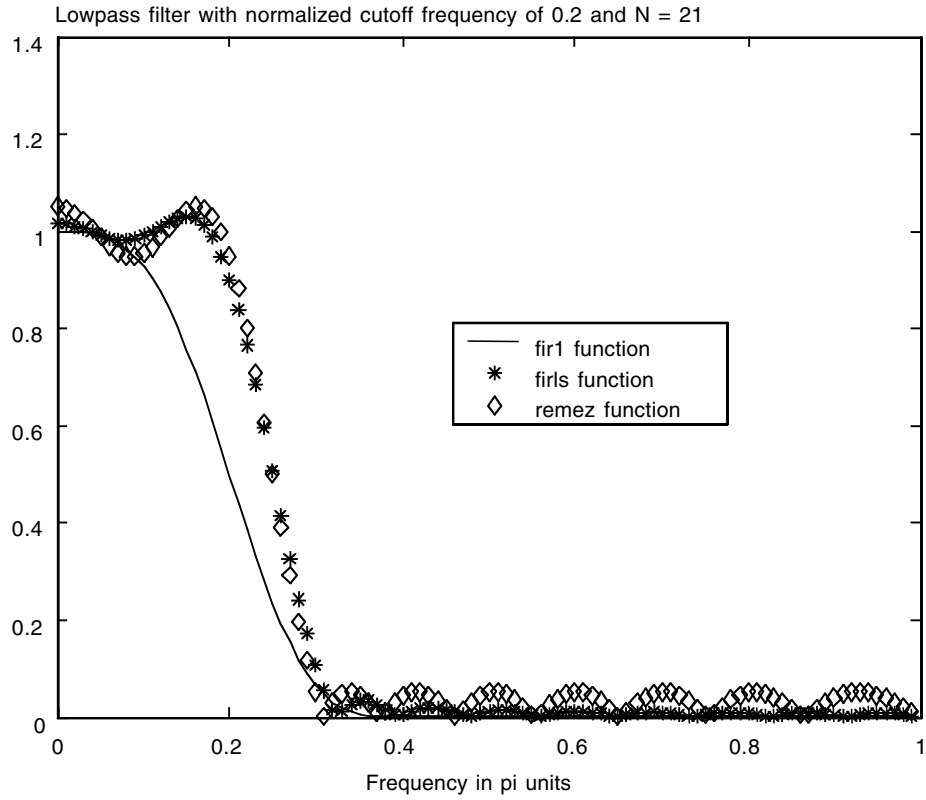

FIGURE 11.24 Plots for EOCE 11.11.

```
%Design using the fir1 function
h=fir1(22,[0.3 .6], 'stop');% 22 is even for bandstop IIR
 filters
[H f]=freqz(h,1,100);
plot(f/pi, abs(H)); hold on;
title('Bandstop filter with normalized cutoff frequency of
 [0.3 .6] and N=22');
%Design using the firls function
n=22;
f=[0 .1 .2 .3 .4 .5 .6 .7 .9 1];%pairs of frequencies(even
 number)
m=[1 1 1 0 0 0 0 1 1 1];
h=firls(n,f,m);
[H f]=freqz(h,1,100);
plot(f/pi, abs(H),'*');
%Design using the remez function
```

```
f=[0 .1 .2 .3 .4 .5 .6 .7 .9 1];%pairs of frequencies(even
 number)
m=[1 1 1 0 0 0 0 1 1 1];
h=remez(n,f,m);
[H f]=freqz(h,1,100);
plot(f/pi, abs(H),'d');
legend('fir1 function','firls function','remez function',0);
xlabel('Frequency in pi units');
```

The plots are shown in Figure 11.25. We can see clearly that the `firls` and the `remez` functions produce the best approximation. For these functions the magnitude in the stopband between .3π and .6π is approximately zero.

EOCE 11.13

Design an ideal differentiator using the `remez` function from MATLAB.

Solution

First we will take one pair of frequency points; we take f = [0 1] with the corresponding magnitude vector `m` = [0 π]. We will use `n` = 11 to complete

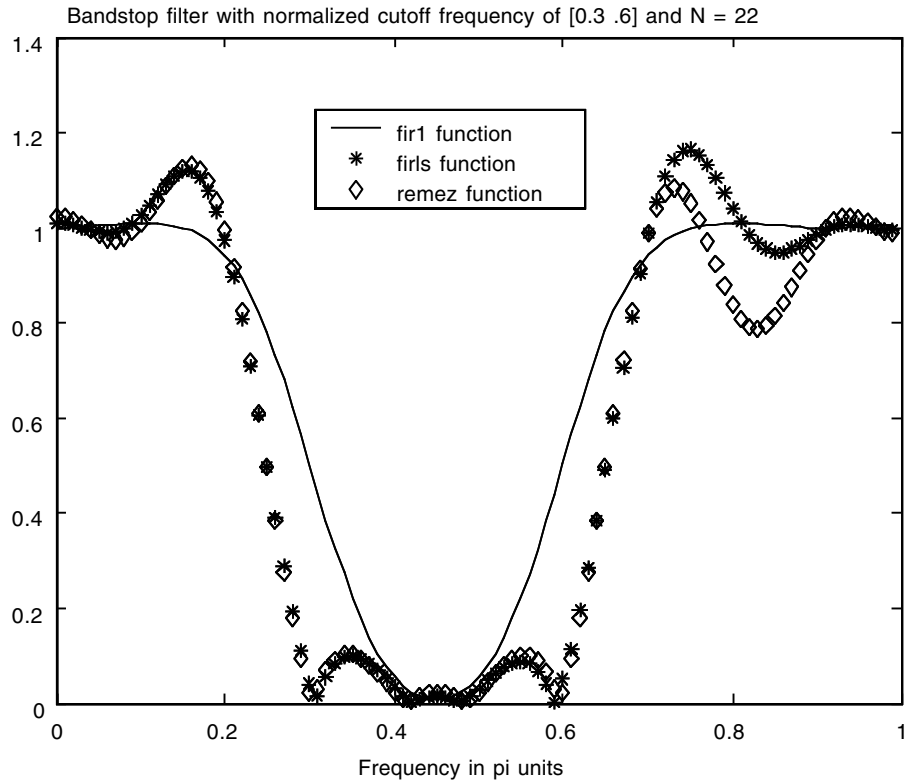

FIGURE 11.25 Plots for EOCE 11.12.

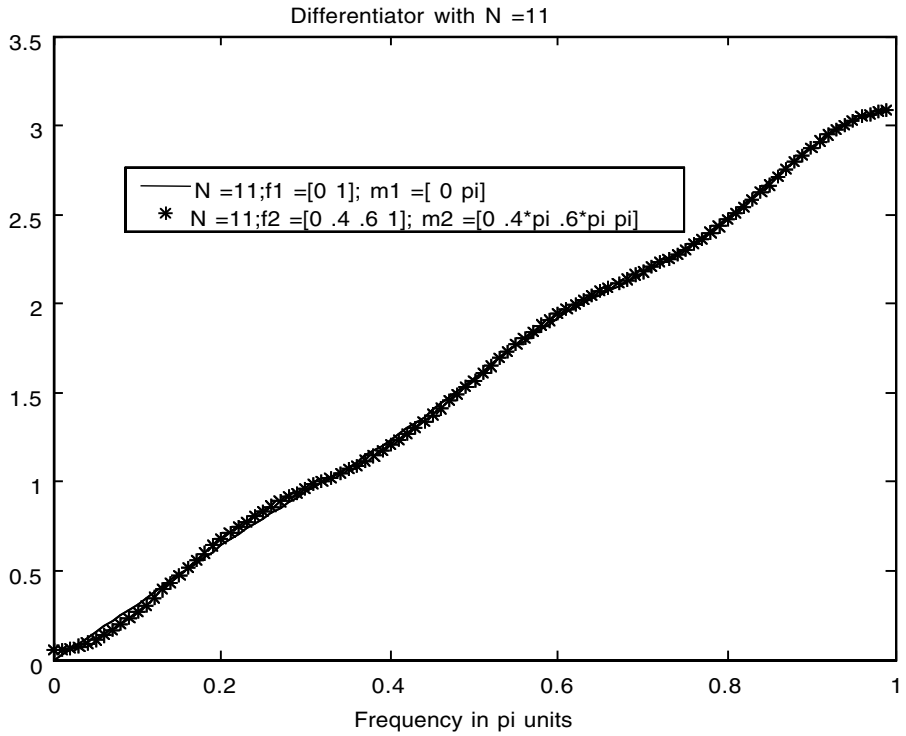

FIGURE 11.26 Plots for EOCE 11.13.

the design. Next we take more frequency pairs; we take `f = [0 .4 .6 1]` with `m = [0 .4π .6π π]`. The MATLAB script is shown next. The plot is shown in Figure 11.26.

```
%Design of a differentiator
n=11
f1=[0 1]; m1=[ 0 pi];
f2=[0 .4 .6 1]; m2=[0 .4*pi .6*pi pi];
h1=remez(n,f1,m1,'d');%d for differentiator
h2=remez(n,f2,m2);
[H1 f]=freqz(h1,1,100); % 100 frequency points in [0 pi]
[H2 f]=freqz(h2,1,100);
plot(f/pi,abs(H1)); hold on;
title('Differentiater with N=11');
xlabel('Frequency in pi units');
plot(f/pi,abs(H2),'*');
legend('N=11;f1=[0 1]; m1=[ 0 pi]','N=11;f2=[0 .4 .6 1]; m2=[0
 .4*pi .6*pi pi]',0);
```

EOCE 11.14

Design a Hilbert transform filter with N = 11, 31 and 51. Plot the magnitude responses.

Solution

We will use the functions `fir1, firls` and `remez` for the design.

```
clf
%we will use N=9 then 21
for N=11:10:31; %filter order
m=(N-1)/2;
n=0:N-1;
mm=n-m+eps;
hilbert=(1-cos(mm*pi))./(mm*pi);
hilbert=hilbert.*hamming(N)';
[Hilbert fh]=freqz(hilbert,1);
subplot(3,1,1);
plot(fh/pi,abs(Hilbert)); hold on;
end
title('Hilbert transform FIR filter magnitude response using
 Eqn. 11.69');
%Design using the MATLAB function firls
for N=12:10:32; %filter order N should be even so magnitude
 at pi is 0
h=firls(N,[0 1],[1 1],'hilbert');
[H f]=freqz(h,1);
subplot(3,1,2);
plot(fh/pi,abs(H)); hold on;
end
title('Hilbert transform FIR filter magnitude response using
 firls');
%Design using the remez function
for N=12:10:32; %filter order N should be even so magnitude
 at pi is 0
h=remez(N,[0 1],[1 1],'hilbert');
[H f]=freqz(h,1);
subplot(3,1,3);
plot(fh/pi,abs(H)); hold on;
end
title('Hilbert transform FIR filter magnitude response using
 remez')
xlabel('Frequency in pi units');
```

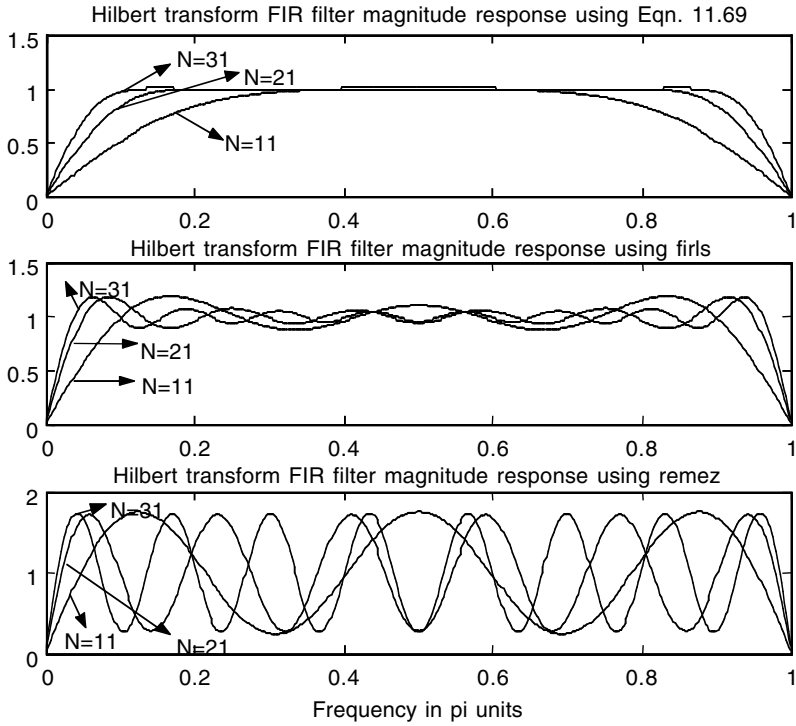

FIGURE 11.27 Plots for EOCE 11.14.

The results are shown in Figure 11.27.

EOCE 11.15

Let θ_c be the cutoff frequency of a lowpass filter. If the same θ_c is also the cutoff frequency of a highpass filter, then in the frequency domain we can write the frequency response of the lowpass filter, $H_{lp}(e^{j\theta})$, and the frequency response of the highpass filter, $H_{hp}(e^{j\theta})$, according to the relation

$$H_{lp}(e^{j\theta}) + H_{hp}\left(e^{j\theta}\right) = 1 \tag{11.79}$$

If we inverse transform Equation (11.79) we get

$$h_{lp}(n) + h_{hp}(n) = \delta(n) \tag{11.80}$$

Similarly for the bandpass and the bandstop FIR filters with the same θ_u and θ_l we have

$$h_{bp}(e^{j\theta}) + H_{sb}\left(e^{j\theta}\right) = 1 \tag{11.81}$$

Taking the inverse transform of Equation (11.81), we get the relation

$$h_{bp}(n) + H_{sb}(n) = \delta(n) \tag{11.82}$$

If the stopband edge frequencies of a stopband filter are at θ_l and θ_u, then the following relation is also true

$$h_{sb}(n) = h_{lp}(n)\Big|_{\theta\lambda} - h_{lp}(n)\Big|_{\theta u} + \delta(n) \tag{11.83}$$

1. Use the MATLAB function `fir1` to plot the highpass filter magnitude response with $\theta_c = .5\ \pi$ derived from the lowpass IIR filter.
2. Use the MATLAB function `fir1` to plot the stopband filter magnitude response derived from the lowpass FIR filter. The stop edges are at $\theta_l = .4\pi$ and $\theta_u = .8\pi$.
3. Use the MATLAB function `fir1` to plot the bandpass filter frequency response derived from the lowpass FIR filter. The pass edges are at $\theta_\gamma = .4\pi$ and $\theta_u = .8\pi$.

Solution

1. We will use `n = 64` to complete the design. The MATLAB script follows. The plots are shown in Figure 11.28.

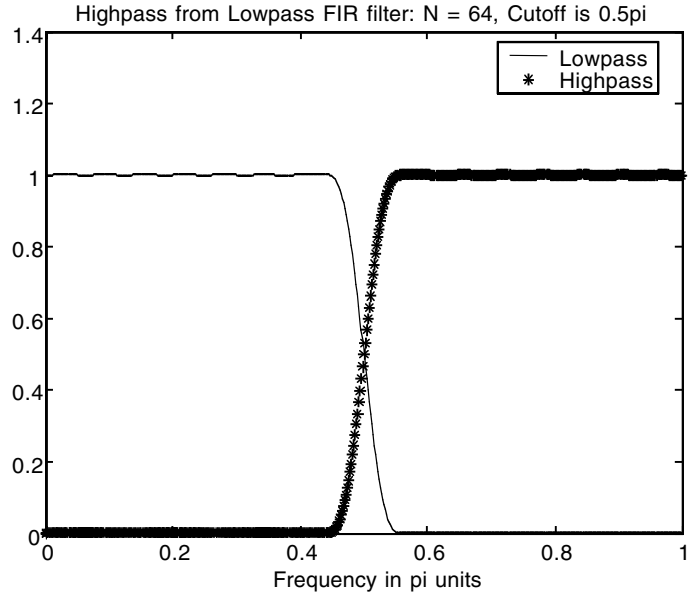

FIGURE 11.28 Plots for EOCE 11.15.

```
n=64;
hlp=fir1(n,.5);
hap=fir1(n,1-eps);%allpass filter. Avoid division by zero
hhp=hap-hlp;
[Hap fap]=freqz(hap,1);
[Hlp flp]=freqz(hlp,1);
[Hhp, fhp]=freqz(hhp,1);
plot(flp/pi,abs(Hlp)); hold on;
plot(fhp/pi, abs(Hhp),'*');
title('Highpass from Lowpass FIR filter: N=64, Cutoff is
 0.5pi');
xlabel('Frequency in pi units');
legend('Lowpass','Highpass',0);
```

2. We will use `n = 64` to complete the design. The MATLAB script follows. The plots are shown in Figure 11.29.

```
wc1=.4; wc2=.8;
n=64;
hlpl=fir1(n,.4);
hlpu=fir1(n,.8);
```

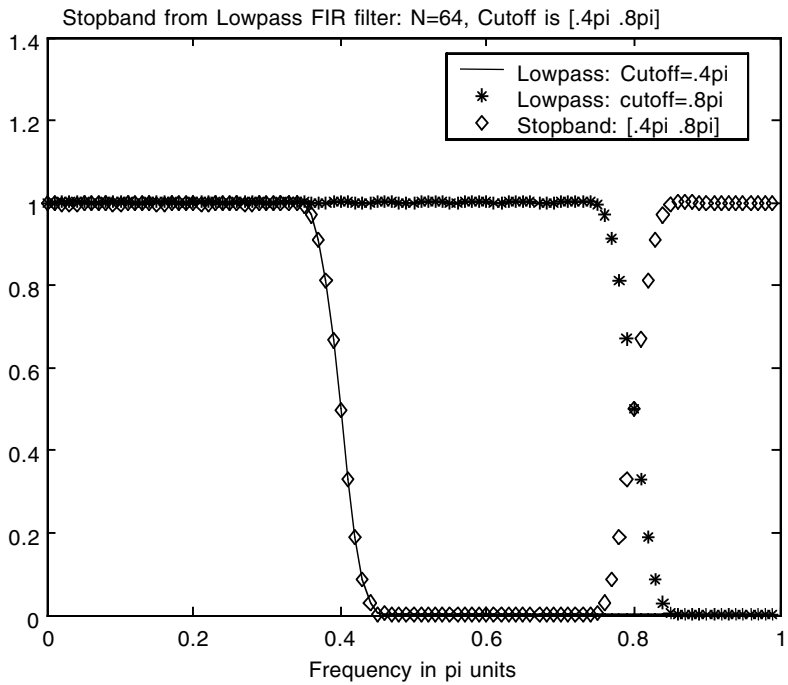

FIGURE 11.29 Plots for EOCE 11.15.

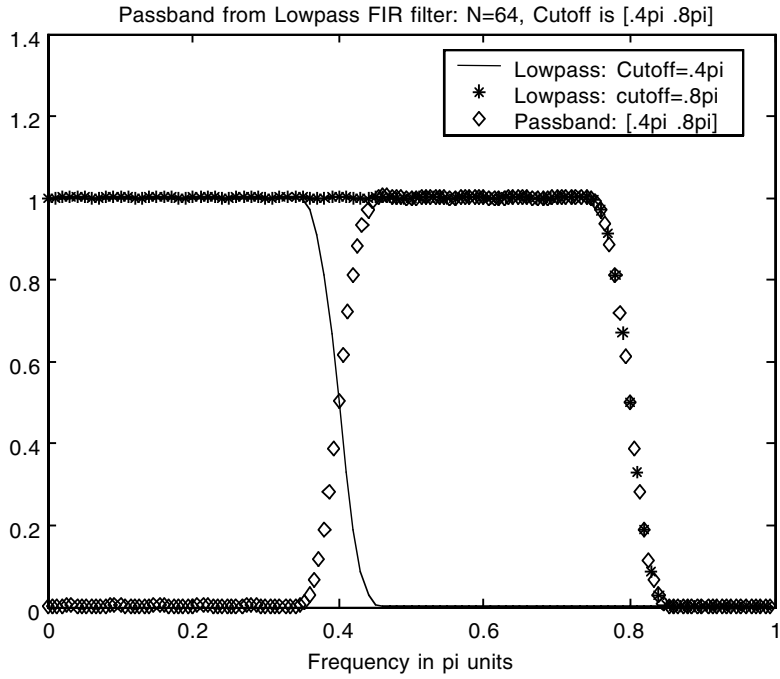

FIGURE 11.30 Plots for EOCE 11.15.

```
hap=fir1(n,1-eps);%avoid division by zero
hsb=hlpl-hlpu + hap;
[Hlpl flpl]=freqz(hlpl,1,100); %for not to condense the
 spectra
[Hlpu flpu]=freqz(hlpu,1,100);
[Hsb, fsb]=freqz(hsb,1,100);
plot(flpl/pi,abs(Hlpl)); hold on;
plot(flpu/pi, abs(Hlpu),'*');
plot(fsb/pi, abs(Hsb),'d');
title('Stopband from Lowpass FIR filter: N=64, Cutoff is
 [.4pi .8pi]');
xlabel('Frequency in pi units');
legend('Lowpass: Cutoff=.4pi','Lowpass: cutoff=.8pi','
 Stopband: [.4pi .8pi]',0);
```

3. We will use n = 64 to complete the design. The MATLAB script follows. The plots are shown in Figure 11.30.

```
wc1=.4; wc2=.8;
n=64;
hlpl=fir1(n,.4);
```

```
hlpu=fir1(n,.8);
hpb=hlpu-hlpl;
[Hlpl flpl]=freqz(hlpl,1,100); %for not to condense the
 spectra
[Hlpu flpu]=freqz(hlpu,1,100);
[Hpb, fpb]=freqz(hpb,1,150);
plot(flpl/pi,abs(Hlpl)); hold on;
plot(flpu/pi, abs(Hlpu),'*');
plot(fpb/pi, abs(Hpb),'d');
title('Passband from Lowpass FIR filter: N=64, Cutoff is
 [.4pi .8pi]');
xlabel('Frequency in pi units');
legend('Lowpass: Cutoff=.4pi','Lowpass: cutoff=.8pi','
 Passband: [.4pi .8pi]',0);
```

11.9 End of Chapter Problems

EOCP 11.1

Consider the following difference equations

1. $y(n) = x(n) + x(n-1) + x(n-2) + x(n-3) + x(n-4)$
2. $y(n) = \frac{1}{2}x(n) + x(n-1) + x(n-2) + x(n-3) + \frac{1}{2}x(n-4)$
3. $y(n) = x(n) + x(n-1) + x(n-2) + \frac{1}{2}x(n-3) + \frac{1}{2}x(n-4)$
4. $y(n) = x(n) + 2x(n-1) + 3x(n-2)$
5. $y(n) = x(n) + x(n-1) + x(n-2) + 2x(n-3) + x(n-4)$

Find the impulse response $h(n)$ for all the filters above and plot their frequency responses using the function `freqz` and the `fft`. Compare the results.

EOCP 11.2

Consider the following continuous signals

1. $x(t) = 2 + \cos(10t) + \sin(100t)$
2. $x(t) = \cos(10t)$
3. $x(t) = \cos(100t) + \sin(1000t)$

Design a lowpass digital filter to filter out the sinusoidal term with the highest frequency in each input signal above. Use different windows in the

design. Comment on your answers and the resulting plots. Do not use MATLAB filter design functions.

EOCP 11.3

Consider the analogue signals

1. $x(t) = \cos(100t) + \sin(1000t) + \sin(500t)$
2. $x(t) = 10 + \cos(150t) + \sin(1050t)$
3. $x(t) = \cos(t) + \sin(10000t) + \sin(400t)$

We are interested in passing the highest frequency in the input signals given. Design a filter that can do that. Use different windows and compare results. Do not use MATLAB filter design functions.

EOCP 11.4

Consider the following analogue signals

1. $x(t) = \cos(5/2\,t) + \sin(10/3\,t) + \sin(5/4t)$
2. $x(t) = 10 + \cos(150t) + \sin(700t)$
3. $x(t) = \cos(200t) + \sin(1020t) + \sin(400t)$

Design a filter that will suppress the intermediate frequency and pass the other two in the input signals above. Again, use different windows and you may increase the order of the filter in some cases to eliminate some frequencies if they are very close. Do not use MATLAB filter design functions.

EOCP 11.5

Consider the following analogue signals

1. $x(t) = \cos(5/7\,t) + \sin(1/3\,t) + \sin(5/4t)$
2. $x(t) = 1 + \cos(140t) + \sin(70t)$
3. $x(t) = \cos(200t) + \sin(3\pi/4\ t) + \sin(400t)$

Design a filter that will suppress the highest and the lowest frequencies and pass the one in between in the input signals above. Again, use different windows and you may increase the order of the filter in some cases to eliminate some frequencies if they are very close. Do not use MATLAB filter design functions.

EOCP 11.6

Repeat EOCP 11.2 by using the MATLAB filter design functions.

EOCP 11.7

Repeat EOCP 11.3 by using the MATLAB filter design functions.

EOCP 11.8

Repeat EOCP 11.4 by using the MATLAB filter design functions.

EOCP 11.9

Repeat EOCP 11.5 by using the MATLAB filter design functions.

EOCP 11.10

We are interested in removing the unwanted frequencies 50 Hz, 100 Hz and 150 Hz from the continuous signal $x(t)$.

1. Design a digital filter that will eliminate these frequencies. Assume that the highest frequency in $x(t)$ is 1000 Hz.
2. Demonstrate by using a sample input that the filter is working.

EOCP 11.11

Consider the following IIR digital filters

1. $H(z) = \dfrac{z-1}{z^2 - \dfrac{1}{8}}$
2. $H(z) = \dfrac{1}{z^2 - \dfrac{3}{4}z + \dfrac{1}{8}}$
3. $H(z) = \dfrac{z^2}{z^2 - 5z + 2}$
4. $H(z) = \dfrac{z^2 - z + 1}{z^2 - \dfrac{3}{4}z + \dfrac{1}{8}}$
5. $H(z) = \dfrac{z^2 - 1}{z^3 + \dfrac{1}{8}}$

Derive an FIR digital filter to approximate the digital IIR filter. Comment on the results.

EOCP 11.12

Consider the following digital filter magnitude responses shown in Figure 11.31 through Figure 11.35. Use the frequency sampling method to design an FIR lowpass filter to approximation the magnitude response of these IIR filters.

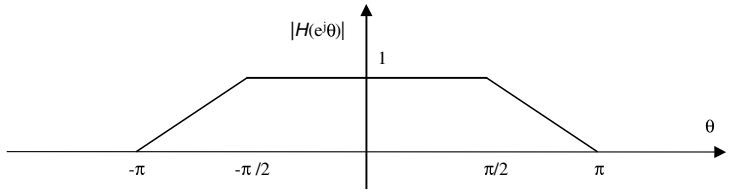

FIGURE 11.31 Filter for EOCP 11.12.

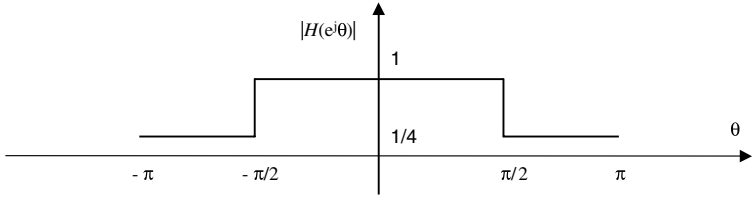

FIGURE 11.32 Filter for EOCP 11.12.

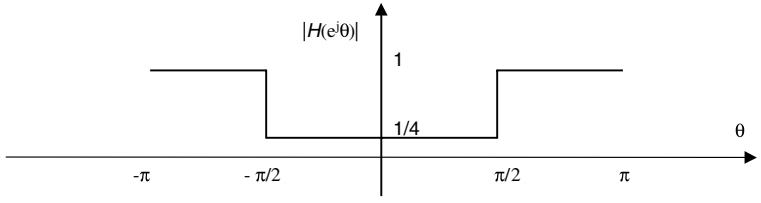

FIGURE 11.33 Filter for EOCP 11.12.

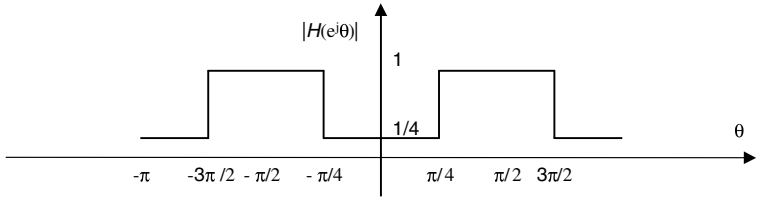

FIGURE 11.34 Filter for EOCP 11.12.

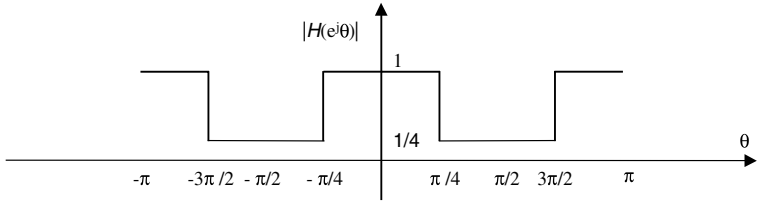

FIGURE 11.35 Filter for EOCP 11.12.

EOCP 11.13

We are interested in removing the frequencies that are higher than 115 kHz from the analogue signal $x(t)$. Design a digital filter to accomplish that. Use a sampling frequency of 500 kHz. Plot the results and use different windows.

EOCP 11.14

We wish to pass the range of frequency $40\,\text{kHz} \leq f \leq 80\,\text{kHz}$ in the analogue signal $x(t)$. Suppose that the highest frequency is $x(t)$ in 100 kHz. Design a bandpass digital filter to do just that. Use different windows and plot results.

EOCP 11.15

Use MATLAB to design a lowpass filter of order 32 that will pass frequencies from 0 to 1 kHz with unity magnitude. We expect zero magnitude for other frequencies. Assume that the highest frequency coming to this filter is 10 kHz. Use the functions `fir1, firls` and `remez` for the design. Use the Blackman and the Hanning windows. Plot the results.

EOCP 11.16

We are interested in designing a bandstop FIR filter with edge frequencies at 0.1π and 0.3π. The gain in the passbands is three and in the stopband is zero. Use the functions `fir1, firls` and `remez` for the design. Use the Hamming and rectangular windows and plot the results.

EOCP 11.17

Design an ideal differentiator using the `remez` function from MATLAB. Compare it with the formula derived in the chapter. Plot the magnitude responses.

EOCP 11.18

1. Use MATLAB to plot the highpass filter magnitude response with $\theta_c = 0.2\pi$ derived from the lowpass IIR filter.
2. Use MATLAB to plot the stopband filter magnitude response derived from the lowpass FIR filter. The stop edges are at $\theta_l = 0.2\pi$ and $\theta_u = 0.6\pi$.
3. Use MATLAB to plot the bandpass filter frequency response derived from the lowpass FIR filter. The pass edges are at $\theta_\ell = 0.2\pi$ and $\theta_u = 0.8\pi$.
4. Use MATLAB to plot the lowpass filter frequency response derived from the highpass FIR filter. Use a cutoff frequency of $\theta_c = 0.5\pi$.
5. Use MATLAB to plot the highpass filter frequency response derived from the lowpass FIR filter. Use a cutoff frequency of $\theta_c = 0.5\pi$.

References

1. ElAli, T.S. and Karim, M., *Continuous Signals and Systems with Matlab*, CRC Press LLC, Boca Raton, FL, 2001.
2. Chen, C., *Digital Signal Processing*, Oxford University Press, New York, 2001.
3. Sherrick, J.D., *Concepts in Systems and Signals*, Prentice Hall, Upper Saddle River, NJ, 2001.
4. Kirk, D.S. and Strum, R.D., *First Principles of Discrete Systems and Digital Signal Processing*, Addison-Wesley, Reading, MA, 1989.
5. Kamen, E.D. and Heck, B.S., *Signals and Systems*, 2nd ed., Prentice Hall, Upper Saddle River, NJ, 2000.
6. Zeimer, E.Z., Trantler, W.H. and Fannin, D.R., *Signals and Systems*, 4th ed., Prentice Hall, Upper Saddle River, NJ, 1998.
7. Oppenheim, A.V. and Schafer, R.W., *Discrete-Time Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1989.
8. Lyons, R.G., *Understanding Digital Signal Processing*, Addison-Wesley, Reading, MA, 1997.
9. Orfanidis, S.J., *Introduction to Signal Processing*, Prentice Hall, Upper Saddle River, NJ, 1996.
10. Ingle, V.K. and Proakis, J.G., *Digital Signal Processing*, PWS, Boston, MA, 1997.
11. Lathi, B.P., *Signal Processing and Linear Systems*, Berkley Cambridge Press, Carmichael, CA, 1998.
12. Mitra, S.K., *Digital Signal Processing*, 2nd ed., McGraw-Hill, New York, 2001.
13. Strum, R.D. and Kirk, D.E., *Contemporary Linear Systems*, PWS, Boston, MA, 1996.
14. Chen, C., *Systems and Signal Analysis*, Oxford University Press, New York, 1994.
15. Denbigh, P., *System Analysis and Signal Processing*, Addison-Wesley Longman, Edinburgh Gate, England, 1998.
16. Phillips, C.L. and Parr, J.M., *Signals, Systems, and Transforms*, 2nd ed., Prentice Hall, Upper Saddle River, NJ, 1999.
17. Vlach, J., *Computerized Approximation and Synthesis of Linear Networks*, John Wiley & Sons, New York, 1969.
18. Daniels, R.W., *Approximation Methods for Electronic Filter Design*, McGraw-Hill, New York, 1974.
19. Temes, G.C. and LaPatra, J.W., *Introduction to Circuit Synthesis and Design*, McGraw-Hill, New York, 1977.
20. Antoniou, A., *Digital Filters: Analysis, Design, and Applications*, 2nd ed., McGraw-Hill, New York, 1993.
21. Parks, T.W. and Burrus, C.S., *Digital Filter Design*, John Wiley & Sons, New York, 1987.
22. Guillemin, E.A., *Theory of Linear Physical Systems*, John Wiley & Sons, New York, 1963.

23. Champers, J.A., Tantaratana, S. and Bomar, B.W., *The Electronics Handbook*, CRC Press LLC, Boca Raton, FL, 1996.
24. Rabiner, L.R. and Gold B., *Theory and Application of Digital Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1975.

Index

A

Active circuit elements, 422
Active filters, 422
AC voltage source, 31
A/D, *see* Analogue-to-digital conversion
Algebra, easy-to-manipulate, 165
Algebraic equations, 115
Aliasing, 529, 531, 532, 534
  prevention of, 617
  sampling without, 493
Amplitude scaling, 20
Analogue-to-digital (A/D) conversion, 488
Analogue filter design, 421–485
  analogue filter design using MATLAB, 438–442
    analogue frequency transformation, 442
    analogue prototype design functions, 440
    complete classical IIR filter design, 440–442
    order estimation functions, 439
  analogue filter specifications, 422–425
  analogue frequency transformation, 437–438
  Bessel filters, 434–437
  Butterworth filter approximation, 425–428
  Chebyshev filters, 428–433
    inverse Chebyshev filter, 431–433
    Type I Chebyshev approximation, 428–431
  comparison between analogue filter types, 447–448
  cut-off frequency, 443–447
  elliptic filter approximation, 433–434
  examples, 449–478
  insights, 448–449
  limitations, 447
  problems, 479–485
Analogue frequency, 159, 437
Analogue prototype
  functions, 440, 548
  IIR digital filter and, 548, 559, 565, 573
Analogue transformation functions, 549
Anti-aircraft gun, 31
Approximation methods
  Bessel filters, 434
  Butterworth, 425
  Chebyshev Type I, 428
  Chebyshev Type II, 428, 431
  elliptic, 433
  equiripple behavior of, 428
  monotonic behavior of, 428
Auto-correlation, 87
  definition of, 89, 394
  important application of, 395
Auxiliary equation, system, 220

B

Backward difference transformation, 512, 514
Band-limited signal, 492, 493
Bandpass filter(s), 439
  approximation
    Fourier series method, 633
    frequency sampling, 633
  bode plot, 470
  fifth-order, 468
  fourth-order, 567
  ideal, 601
  magnitude response for, 601
  transfer functions of, 570
  transforming lowpass filter to, 542
Bandstop cut-off frequencies, 574
Bandstop filter (BSF), 423, 439
  center frequency, 475
  ideal, 601
  magnitude response, 471, 601
  sixth-order, 574
  transforming lowpass filter to, 542
Bessel filter(s), 434
  design functions, 441
  frequency response of, 436
  phase response, 436
  transfer function, 435
BIBO system, *see* Bounded-input bounded-output system
Bilateral *z*-transform, 195
Bilinear transformation, 506, 512, 514
  coefficients, 569

filter design using, 545
impulse invariance transformation vs., 553
magnitude response, 554, 560
pole obtained using, 547
Binary code, 488, 490
Blackman window, 603, 616
Block diagram, 80, 81, 83, 298
difference equation and, 81, 82
drawing of from state and output equation, 342
representation, 112, 113, 115, 210, 334, 335, 337
linear discrete systems, 78
MATLAB scripts, 359
states matrices, 353
state-space representation, 336
Block filtering, 393, 413
Bounded-input bounded-output (BIBO) system, 63
BPF, *see* Passband filter
Broadcast signals, 329
BSF, *see* Bandstop filter
Butterworth filter(s), 423, 460
approximation, 425
characteristics, 453
cut-off frequency, 440, 553
design of, 577, 582
magnitude response of, 454, 463, 551
monotonicity in passband, 446
nonlinear phase, 613
pole zero plot of, 465

C

Capacitor, charging and discharging of, 1
CAT scan operation, 487
Causal filters, difference equation, 594
Causal systems, definition of, 61
Center frequency, bandstop filter, 475
Characteristic equation, 76, 102, 104, 110
characteristic root of, 123
coefficients, 77
difference equation representation, 330
roots of, 75
Chebyshev filter(s), 428, 462
cut-off frequency for, 430
design functions, 441
inverse, 431
magnitude response of, 432, 463, 464
nonlinear phase, 613
pole locations, 433
pole zero plot of, 466
transfer function, 429, 431
Chebyshev Type II approximation, MATLAB function, 469
Circle of unity magnitude radius, 200
Circular convolution, 372, 380, 383
definition of, 382
equation, 385
linear convolution and, 396
Comb filter(s)
design of, 607
transfer function of, 624
Command line prompt, 450
Communication channel interference, 329
Complex number(s)
addition, 145
complex conjugate terms and, 230
definition of, 145
division, 146
multiplication, 145
polar to rectangular, 146
rectangular to polar, 146
review of, 143
subtraction, 145
z-transform, 200
Conditional statements, building of using analogue circuits, 488
Constant coefficients realization, 595
Continuous filter, cut-off frequency of, 509
Continuous Fourier transform, 528
approximated, 376, 385
discrete Fourier transform and, 375
Continuous frequency, 365
Continuous radian frequency, 10
Continuous signal(s), 1, 2, 489
analogue frequency of, 159
binary code representation, 488
discretized, 24
Fourier series approximation, 387
frequency domain, 496
MATLAB simulated, 528
process of discretizing, 488
radar station, 487
sampling and recovery of, 496
Continuous system
differential equation, 522
impulse response, 527
input-output relationship, 507
oscillatory plot of, 523
partial fraction expansion, 525
plots, 519
state-space, 524
transfer functions, 527
Continuous value, 503
Continuous wave, example of, 1
Control systems, 488
Convergence

conditions, 161
z-transform, 200
Convolution, 64
circular, 372, 380, 383
definition of, 382
equation, 385
linear convolution and, 396
equation, 66, 67, 104, 210
frequency response, 600
linear, 381, 412
equation, 382
output using, 413
property, 224
Fourier transform, 164
z-transform, 254
real-time, 225
result, 96, 381
sum(s), 66, 68, 173
equation, 593
evaluation of output using, 335
z-transform, 210
Correlation, 87
radar signal, 394
signals, discrete Fourier transform, 368
Cross-correlation, 87, 407, 408, 409
equation, 88, 394
important application of, 132
Cross multiplication, 348
Cubic-spline interpolation, 533
Current, continuous signal for, 10
Cut-off frequency(ies), 498, 543
analogue filter, 546
bandstop, 574
Butterworth filter, 440, 553
calculation of, 443
continuous filter, 509
digital filter, 546
first-order circuits, 448
highpass filter, 441, 445, 559, 620
lowpass filter, 445, 640
normalized, 557
use of MATLAB to estimate, 451

D

D/A, *see* Digital-to-analogue conversion
Data values, distorted, 329
Decimation in time, 379
Defining equation, discrete Fourier transform, 367
Delay elements, 79, 212
Delta signal, 331
Dense spectrum, 412
Derivative operation, discretizing of, 500
DFT, *see* Discrete Fourier transform
Difference equation, 174
block diagram and, 81, 82
causal, 232, 594
change of coefficients in, 614
delta signal as input, 331
digital FIR filter, 610
equivalent, 502
first-order, 155
general, 106, 123, 155
homogeneous, 69, 70, 101
impulse response, 85
inverse *z*-transform and, 608
linear, 330
model plots, 355
nonhomogeneous, 73, 86
nonrecursive filters, 594
N^{th} order, 236
physical systems, 68
representation, 126, 330, 336, 338, 341, 358
cross multiplication, 348
impulse response, 351
solving of using *z*-transform, 214
state-space representation, 333
step response, 350
systems represented as, 82, 112
unity coefficient, 348
z-transformed, 241, 332
Differentiation property, 223
Differentiator
design of using MATLAB, 637
FIR digital, 612
system, input to, 605
transfer function, 606
Digital-to-analogue (D/A) conversion, 488
Digital filter, 393
analogue to, 549
IIR to FIR, 610
transfer function, 509
Digital frequency, 160, 365, 496
maximum, 630
normalized, 634
Digital passband, normalized, 571
Digital processor, 487
Digital shifter, design of, 609
Discrete Fourier transform (DFT), 366, 507, 508, 604, 628
approximation using, 400
block filtering with, 412
continuous system, 506
correlation signals for, 368
equation, 366, 377
exact approximation using, 401
frequency index for, 366
inverse, 378
properties of, 372

with zero padding, 414
Discrete Fourier transform and discrete systems, 365–419
applications of DFT, 380–395
approximation to coefficients of Fourier series, 387–391
approximation to continuous Fourier transform, 385–387
block filtering, 393–394
circular convolution, 380–384
correlation, 394–395
linear convolution, 384–385
total energy in signal, 391–393
discrete Fourier transform and finite-duration discrete signals, 366–367
exercises, 396–415
fast Fourier transform, 378–380
insights, 395–396
circular convolution and linear convolution, 396
DFT points are samples of Fourier transform of $x(n)$, 395
DFT same as fft, 395
frequency contents of $x(t)$ in DFT, 395–396
frequency leakage and DFT, 396
$|X(w)|$, 396
numerical computation of DFT, 377–378
problems, 415–419
properties of discrete Fourier transform, 367–373
defining equation, 367–368
DFT linearity, 370–371
DFT symmetry, 368–370
magnitude of DFT, 371
meaning of DFT, 372–373
relation of DFT with Fourier transform of discrete signals, 373–377
DFT and continuous Fourier transform of $x(t)$, 375–377
DFT and Fourier transform of $x(n)$, 373–374
DFT and z-transform of $x(n)$, 374–375
Discrete linear systems, ways of representing, 330–333
block diagram representation, 334
difference equation representation, 330–331
impulse response representation, 331–332
state-space representation, 333–334
z-transform representation, 332–333
Discrete matrix, 506
Discrete periodic signals, Fourier series of, 147
Discrete signal(s), 489
average power in, 28
basic operations, 25–28
addition and subtraction, 25
combined operations, 26–28
modulation, 25
scalar multiplication, 25
bounded, 30
complex periodic, 10
conversion of continuous signal to, 487
decaying exponential, 8
decaying sinusoidal, 12
digitized, 3
even, 21
example of, 2
finite-duration, 366
Fourier transform of, 159, 373, 395
growing exponential, 8, 10
impulse, 6
odd, 21
Parseval's relation for, 167
periodic, 3, 4
plot of, 143
ramp, 6, 7
real exponential, 7
representation of, 16, 21
shifted, 15
sinusoidal, 7, 9
time constant, 23, 24
time invariance and, 58
time-scaled, 19
total energy in, 28
unbounded, 30
unit step, 4, 5, 198
z-transform of, 199
Discrete system, 55–141
block diagram representation of linear discrete systems, 78–81
delay element, 79
multiplier, 79–81
summing/subtracting junction, 79
causal systems, 61–62
convolution, 64–68
correlation, 87–89
autocorrelation, 89
cross-correlation, 87–89
definition of system, 55
difference equations of physical systems, 68
difference equations representing, 350–351
examples, 91–134
frequency response of, 154
from block diagram to difference equation, 81–82
from difference equation to block diagram, 82–85
homogeneous difference equation and solution, 69–73
case when roots are all different, 71

case when two roots are complex, 72–73
case when two roots are real and equal, 72
impulse response, 85–87, 151
input and output, 55–56
insights, 90–91
eigenvalues, 90–91
stability and eigenvalues, 91
inverse of system, 62–63
linear discrete systems, 56–58
nonhomogeneous difference equations and solution, 73–75
with periodic inputs, 150
problems, 134–141
stability of linear discrete systems, 75–78
stability depending on values of poles, 75–76
stability from jury test, 76–78
stable system, 63–64
systems with memory, 60–61
time invariance and discrete signals, 58–60
Discrete time
domain, 254
shifting property, Fourier transform, 163
Discrete value, 503
Discretization
formula, 502
interval, 506
methods of, 521
state-space representation, 504
Discrimination parameter, 425, 434
Dynamics matrix, 291

E

Edge frequencies, 635, 641
Eigenvalues, 291, 301
Electrical switch, 31
Electromagnetic signal, 1
Element-by-element multiplication, 28, 41
Elevator system, 55
Elliptic filter(s), 462
approximation, 433
design, 441, 458, 582
magnitude-squared response, 434, 459
nonlinear phase, 613
pole zero plot of, 467
ripples, 562
roll-off characteristics, 448
transfer function of, 433
Ending index, 98
Energy
calculations, 167
discrete signal, 28
finite, 29, 30
signals, cross-correlation equations for, 88
spectrum density, signal, 395
total, 391, 392, 393
use of MATLAB to find total, 34
use of Parseval's theorem to find, 168
Equation(s)
algebraic, 91, 115
analogue filter, 422
auxiliary, 91, 220
characteristic, 69, 76, 102, 104, 110
characteristic root of, 123
coefficients in, 77
difference equation representation, 330
roots of, 75
system, 220
Chebyshev filter, 431
circular convolution, 385
convolution, 66, 67, 104, 210, 593
cross-correlation, 88, 394
defining, discrete Fourier transform, 367
difference, 174
block diagram and, 81, 82
causal filters, 594
change of coefficients in, 614
delta signal as input, 331
digital FIR filter, 610
equivalent, 502
first-order, 155
general, 106, 123, 155
homogeneous, 69, 70, 101
impulse response, 85
inverse z-transform and, 608
linear, 330
model, 355
nonhomogeneous, 73, 86
nonrecursive filters, 594
N^{th} order, 236
physical systems, 68
representation, 126, 330, 336, 338, 341, 345, 348, 358
solving of using z-transform, 214
state-space representation, 333
step response, 350
systems represented as, 82, 112
unity coefficient, 348
z-transformed, 241, 332
discrete Fourier transform, 366, 377
Fourier transform, 164
inverse transform on, 233
linear convolution, 382
matrix state, transfer function calculated from, 340
output, 300, 306, 313
in matrix form, 319
z-domain, 314

simultaneous algebraic, 111
state, 298, 300, 306, 313
discrete state-space approximation, 505
in matrix form, 319, 337
obtaining of by inspection, 339
-space matrix, 299
summation, 604
z-transform, 196, 198, 210
Equiripple linear phase, FIR filter design using, 612
Exponential signal, MATLAB script to simulate, 37

F

Fast Fourier transform (FFT), 149,378
development of, 379
implementation of in MATLAB, 380
FFT, *see* Fast Fourier transform
Filter
active, 422
analogue
prototype functions, 440
specifications, 425
zero-pole plot, 423
average, frequency response, 593
bandpass, 439
bode plot, 470
fifth-order, 468
fourth-order, 567
ideal, 601
magnitude response for, 601
transfer functions of, 570
bandstop, 423, 439
center frequency, 475
ideal, 601
magnitude response, 471, 601
sixth-order, 574
transformed, 471, 542
Bessel, 434
design functions, 441
frequency response of, 436
group delay, 435, 436
phase response, 436
transfer function, 435
Butterworth, 423, 460
approximation, 425
characteristics, 453
cut-off frequency, 440, 553
design of, 577, 582
magnitude response of, 454, 463, 551
monotonicity in passband, 446
nonlinear phase, 613
pole zero plot of, 465
causal, difference equation, 594
Chebyshev, 428, 462
cut-off frequency for, 430
design functions, 441
equation, 431
inverse, 431
magnitude response of, 432, 463, 464
nonlinear phase, 613
pole locations, 433
pole zero plot of, 466
circuit elements, 422
coefficients, 619
comb
digital, 607
transfer function of, 624
continuous, cut-off frequency of, 509
design
direct, 549, 550, 555, 575, 576
functions, 441
IIR, 440, 610
indirect method, 575
limitations in, 447
transfer function, 509
transformation from analogue to, 549
use of windows in, 602
elements used in building of, 422
elliptic, 462
approximation, 433
design, 441, 458, 582
magnitude response of, 434, 459, 464
nonlinear phase, 613
pole zero plot of, 467
ripples, 562
roll-off characteristics, 448
transfer function of, 433
function, initial conditions for, 236
group delay of, 423
high gain, 448
highpass, 423
analogue, cut-off frequency, 441
cut-off frequency, 559, 620
ideal, 601
magnitude response for, 583, 601
Hilbert transform, 609, 612, 639
impulse response, 593
linear phase, 422
low gain, 448
lowpass, 421, 423, 424
analogue Bessel, 435
cutoff frequency of, 445, 640
ideal, 498, 599
impulse response, 499
limitations in design of, 446
maximum gain of unity, 472
peak passband ripple, 449
specifications, 424

transforming, 542
use of MATLAB to design, 634
magnitude response, 431, 469
noncausal, 608
nonrecursive, difference equation, 594
order
estimated, 433
required, 430
output, delay in, 597
parameters specifying, 430
passive, 422
phase shift of, 423
problems of designing, 422
prototype, 438
sixth-order, 554
specifications, 439, 446, 472
types, comparison between analogue, 446
Final value theorem, 219
Finite duration signals, 87, 96
Finite impulse response (FIR) digital filters, 591–648
definition, 591
design based on Fourier series, 598–609
design of comb FIR filters, 607–608
design of digital FIR differentiator, 605–607
design of digital shifter, 609
ideal lowpass FIR filter design, 599–601
other ideal digital FIR filters, 601–602
window giving optimal $h(n)$, 604–605
windows used in design of digital FIR filter, 602–604
examples, 614–644
FIR digital design using MATLAB, 611–613
design using equiripple linear phase, 612
design using least-squared error, 612
design using windows, 611–612
obtaining frequency response, 612–613
FIR filter design, 594–598
linear phase of FIR filters, 597–598
stability of FIR filters, 596–597
frequency sampling and FIR filter design, 610–611
from IIR to FIR digital filters, 610
insights, 613–614
comparison with IIR filters, 613
different method used in FIR filter design, 613–614
motivating example, 591–594
problems, 644–648
FIR digital filters, *see* Finite impulse response digital filters
First-order circuits, cut-off frequency for, 448
First-order difference equation, 155
First-order systems, output for, 90
Forward difference transformation, 512, 514
Fourier, Joseph, 143
Fourier series
approximation, 387
coefficients, 148, 387, 389
approximation to, 391
finding of using MATLAB, 390
filter design based on, 598
magnitude coefficients, 149
Fourier series and Fourier transform of discrete signals, 143–194
convergence conditions, 161
discrete system with periodic inputs, 150–154
examples, 173–188
Fourier series of discrete periodic signals, 147–149
Fourier transform of discrete signals, 159–161
frequency response of discrete systems, 154–159
periodicity property, 157
symmetry property, 157–159
insights, 172–173
ease in analysis and design, 172–173
sinusoidal analysis, 173
numerical evaluation of Fourier transform of discrete signals, 168–172
Parseval's relation and energy calculations, 167–168
problems, 189–194
properties of Fourier transform of discrete signals, 162–167
convolution property, 164–167
discrete-time shifting property, 163
frequency shifting property, 163
linearity property, 162
periodicity property, 162
reflection property, 163–164
review of complex numbers, 143–147
addition, 145
definition, 145
division, 146
from polar to rectangular, 146–147
from rectangular to polar, 146
multiplication, 145–146
subtraction, 145
Fourier transform
approximation to magnitude of, 397
calculation of, 533
continuous, 375, 376, 528
discrete, 159, 366, 373
ease in analysis and design, 172
fast, 378
development of, 379
implementation of in MATLAB, 380
MATLAB simulated, 529

pairs, 166
properties, 166
sampled signal, 507
Frequency(ies)
analogue, 159, 507
axis, 400
center, bandstop filter, 475
components
signal, 149, 592
use of DFT to find, 390
continuous, 365
cut-off, 498, 543
analogue filter, 546
bandstop, 574
Butterworth filter, 440, 553
calculation of, 443
continuous filter, 509
digital filter, 546
highpass filter, 445, 559, 620
lowpass filter, 445, 640
normalized, 557
digital, 160, 365, 496, 511
domain(s)
ambiguity in, 492
continuous signal, 496
convolution in, 164, 165
discrete signal in, 144
representation, 195, 497
sampling operation in, 490
signal in, 365
edge, 635, 641
index, 366, 372
input signal, 572
leakage, 396
noise at high, 621
output, 173
pairs, 638
passband edge, 433
passing, 158
points, 637
radian, 497, 498, 624
rejecting, 158
relationship between analogue and digital, 542
resolution, 395, 407, 415
response, 170, 171, 172
Bessel filter, 436
comparison of, 595
computations, 451, 452
convolution between, 600
defined, 155
discrete systems, 154
equation used to find, 155
function, 157, 158
general, 156
how to obtain, 612
magnitude of, 178, 186
MATLAB, 175, 594, 623
periodicity property, 157
properties of, 157
simplified, 608
symmetry property, 157
sampling, 372, 491, 497, 632
bandpass filter approximation using, 633
FIR filter design and, 610–611
shifting property, Fourier transform, 163
sinusoidal wave of, 492
spacing, 170
spectrum, input, 403
transformation(s), 541, 543
analogue, 437
functions, 442
lowpass filter, 542
value, 154
Fundamental period, 13, 14, 15

G

Gamma function, 435
Geometric series sum, 67, 105, 148, 160, 200
Grid interval, 529
Group delay, Bessel filter, 435, 436

H

Hamming window, 396, 603, 616, 630
Hanning window, 396, 603
approximation using, 402
magnitude frequency response plot of, 616
Highpass filter (HPF), 423
cut-off frequency, 559, 620
ideal, 601
magnitude response for, 583, 601
Hilbert transform filter, 609, 612, 639
Homogeneous solution, 68
complex roots in, 72
distinct roots in, 71
equal roots in, 72
HPF, *see* Highpass filter

I

Identity matrix, 289
IIR filter design, *see* Infinite impulse response filter design
Impulse(s)

discrete signal, 6
representing discrete signal using, 16
shifted, 16
sum of, 18
Impulse invariance
filter design using, 542
transformation, 512, 513, 515
bilinear transformation vs., 553
IIR digital filters based on, 545
magnitude response, 554, 560
numerator and denominator coefficients, 555
Impulse response, 85, 100, 108, 114, 241, 247, 593
calculation of, 119
causal, 216
continuous system, 527
convolution between step input signal and, 341, 345
defined, 174, 405
difference equation representation, 351
discrete system, 124, 177
final, 123
linear system, 403
lowpass filter, 499
model, 350, 355
plotting of actual, 134
representation, 112, 331, 338, 341, 348, 358
samples, FIR filter, 610
solution for, 105
state-space representation, 355–356
use of MATLAB to find, 105, 113, 116, 119, 120, 351
use of *z*-transform to find, 215
Impulse signal, 31, 35, 111
discrete signals using, 18
Fourier transform of, 161
MATLAB-generated, 37
response to, 65
shifted, 65
Infinite impulse response (IIR) filter design, 541–589
design process, 542–548
bilinear transform method, 545–548
impulse invariance method, 542–545
examples, 552–584
IIR filter design using MATLAB, 548–550
direct design, 549–550
from analogue prototype to IIR digital filter, 548–549
insights, 550–552
choice of sampling interval T_s, 552
difficulty in designing IIR digital filters in *z*-domain, 550–552
using impulse invariance method, 552
problems, 584–589
Infinite series, 161
Initial condition vector, 237, 239, 311
Initial index, 41
Initial-value theorem, 219, 220, 248
Input
divided, 178
frequency, 154
frequency response at, 186
magnitude of, 182
spectrum of, 403
magnitude of, 154, 173
matrix, 504
noise, auto-correlation of, 132
–output relation, 56, 60, 68, 508
particular solutions for selected, 75
phase shift of, 178
signal(s), 55, 57
change of magnitude of, 393
frequencies, 172, 572
step, 240
vector, 291, 504
Inspection
obtaining state equations by, 339
state-space representation by, 317
Interchange of summation property, 164
Inverse Chebyshev filter, 431
Inverse transform, 232, 298
calculation of, 404
use of MATLAB to find, 315
Inverse *z*-transform, 203, 335
difference equation and, 608
long division, 206
partial fraction expansion, 203

J

Jacobian elliptic function, 433
Jury test, 76, 78

K

Kaiser window, 603, 604

L

Laplace domain, 472, 506, 510
Laplace transform, 423, 519
inverse, 524
state vector, 520
transition matrix, 520

Lindex, 42, 43
Linear convolution, 381, 412
  equation, 382
  output using, 413
Linear discrete systems, 56
  block diagram representation of, 78
  stability of, 75
Linearity property
  discrete Fourier transform, 371
  Fourier transform, 162
  z-transform, 207, 226
Linear system(s)
  impulse response, 403
  representation, 56
  types of, 365
Linear time-invariant (LTI) model, 438
Linear time-invariant system(s), 90
  analysis and design of, 329
  multiple-inputs multiple-outputs in, 346
  output of, 173
  transfer function of, 220, 329
Linear time-variant system, 153
Load resistor, 31
Long division, 206, 216, 227, 610, 627
Lowpass filter (LPF), 421, 423, 424
  analogue Bessel, 435
  cut-off frequency for, 445, 640
  ideal, 498, 599
  impulse response, 499
  limitations in design of, 446
  maximum gain of unity, 472
  peak passband ripple, 449
  prototype, 438
  specifications, 424
  transforming, 542
  use of MATLAB to design, 634
LPF, *see* Lowpass filter
LTI model, *see* Linear time-invariant model

M

Magnitude
  plot, 158, 159, 160
  requirements, design by satisfying, 541
  response, 158, 427, 429
    Butterworth filter, 454, 551
    characteristic, 434
    Chebyshev filter, 432
    elliptic filter, 434, 459
    highpass filter, 583
MATLAB
  analogue filter design using, 438
  approximation to continuous Fourier transform, 385
  calculation of average power using, 33
  complex poles, 517
  control toolbox, 469
  cut-off frequencies, 451
  data entered as row vectors in, 171
  default scaling, 612
  design of ideal differentiator using, 637
  DFT equation implemented on digital computer using, 377
  digital frequencies using, 569, 576
  direct design method, 557, 563
  filter function, 124
  finding of Fourier series coefficients using, 390
  FIR digital design using, 611
  frequency response, 623
  function, 225, 227
    Chebyshev Type II approximation, 469
    direct design, 550
    general form of, 42
    roots, 76
    use of to find step response, 347, 348
    zeros, 35
  -generated exponential decaying signal, 38
  -generated impulse signal, 37
  -generated sinusoidal signal, 39
  -generated step signal, 35, 36
  IIR filter design using, 548
  implementation of fast Fourier transform in, 380
  impulse invariance transformation, 544
  impulse response, 105, 113, 116, 119, 120
  inverse transform, 315
  limitations in filter design using, 447
  lowpass filter design, 634
  magnitude of frequency response, 175
  partial fraction expansion using, 303
  phase response displayed by, 450
  random signal generation, 409
  reconstruction implementations, 533
  recursion, 303, 308
  script(s), 98, 180, 182
    block diagram representation, 359
    discrete input signal, 580
    exponential signal, 37
    frequency responses, 594
    Hanning windows, 402
    IIR digital bandpass filters, 571–572
    IIR digital bandstop filters, 577
    inverse discrete Fourier transform, 378
    response plotting due to step input, 240
    state-space matrix system, 347
    transfer function representation, 357
  shifting function, 49
  signal processing toolbox, 438, 439

simulated continuous signal, 528
simulated Fourier transform, 529
spectral energy estimate, 410
statement, 561
step response, 311, 320
system identification toolbox, 438
system stability, 311
time-shifting property of Fourier transform, 179
total energy, 392, 393
transfer function, 301
transformation, 565
transition matrix entries, 304
windows implemented in, 396
Matrix state equations, transfer function calculated from, 340
Mechanical systems, modeling of, 329
Medical imaging, 487
Memory, systems with, 60, 61
Mixed system, 488
Model(s)
difference equation, 355, 361
impulse response, 350, 355, 361
linear time-invariant, 438
state-space, 346, 355, 361
transfer function, 350
Modeling and representation of discrete linear systems, 329–364
exercises, 346–361
poles considering different outputs within same system, 346
problems, 361–364
ways of representing discrete linear systems, 330–346
block diagram to other representations, 343–346
difference equation to other representations, 330–334
impulse response to other representations, 335–337
state-space to other representations, 340–342
transfer function to other representations, 337–340
Modulation, 25
Multiplication, term-by-term, 384
Multiplier element, 79

N

Noise components, preventing amplification of, 621
Noncausal filter, 608
Noncausal system, 62
Nonhomogeneous difference equations, 73
Nonlinear system, 57, 58
Nonrecursive system, 591, 614
Nyquist criteria, 400
Nyquist rate, 392, 401, 403, 529, 533, 607

O

Order estimation functions, 439
Orthogonality condition, 148
Output
continuous time system, 506
discrete Fourier transform of, 506
equation, 300, 306, 313
in matrix form, 319
z-domain, 314
frequency, 173
initial condition for, 501
matrix, 504
one-dimensional, 303
values, use of *z*-transform method to find, 303
vector, 334
voltage, at low frequencies, 474

P

Parseval's Theorem, 167, 605
Partial fraction expansion, 231, 242, 253, 303
analytical solutions, 321
continuous system, 524
impulse response representation, 338
inverse *z*-transform, 203
transfer function representation, 228, 318, 357
z-transform representation, 332
Particular solution, 68
Passband
bandwidth, 446
edge frequency, 433
filter (BPF), 423
ripple, 425, 429
Passive circuit elements, 422, 475
Passive filters, 422
Periodicity property
Fourier transform, 162
frequency response, 157
Periodic signal
approximated, 148
average power in, 389
Phase
characteristic, nonlinear, 434
shift, 153
Physical systems, difference equations of, 68
Polar form, complex number written in, 146, 200

Pole(s)
locations, Chebyshev filter, 433
phase of, 247
system, 220, 252
transient shape and, 346
Power
average, periodic signal, 389
calculation of using MATLAB, 33
signals, 30, 89
Proportional-integral-derivative control, design of, 605
Prototype filters, 438
Pulse
definition, 405
signal, 16

Q

Quality factor, 446, 448

R

Radar
antenna, 31
signal correlation, 394
station, continuous signal, 487
Radian frequency, 153, 497, 498, 624
Ramp signal, 6, 7, 31
Random signal
dc component, 577
use of MATLAB to generate, 409
Rational number, 14, 15
Real exponential discrete signal, 7
Real number, 14, 23
Real-time domain, ambiguity in, 490
Record length, 407, 415
Rectangular window, 603, 630
filter design using, 618
magnitude frequency response plot of, 616
Recursive system, 614
Reflected signal, 18, 20
Reflection property, Fourier transform, 163
Region of convergence (ROC), 201, 217, 218, 253
Rindex, 42, 43
RLC series circuit, 448
ROC, *see* Region of convergence
Roots
magnitude of, 314
symmetry in, 552

S

Sample-and-hold process, 489
Sampling
filtering before, 494
frequency, 372, 491, 497, 531, 565, 610–611, 632
ideal, 498
interval, 10, 55, 149, 196, 388, 528, 577
changing of, 521–522
choice of, 515, 552
minimum, 572
operation, 490
period, 2, 147–148, 542, 557, 606
rate, Nyquist, 392
theorem, 493, 495
time, transition matrix at, 506
uniform, 501
Scaling factor, 25, 530
s-domain transfer function, 547
Second-order system, 221
initial conditions, 502
output for, 90
Selectivity parameter, 434
Series connection, 317
Shifting property, *z*-transform, 207
Shift operation, importance of, 15–16
Signal(s)
average value of, 389
band-limited, 492, 493
broadcast, 329
continuous, 1, 2, 489
analogue frequency of, 159
binary code representation, 488
discretized, 24
exponential, 23, 24
Fourier series approximation, 387
frequency domain, 496
MATLAB simulated, 528
process of discretizing, 488
processing of, 2
radar station, 487
sampled, 3
sampling and recovery of, 496
correlation, discrete Fourier transform, 368
delta, 331
discrete, 489
average power in, 28
basic operations, 25–28
bounded, 30
complex periodic, 10
conversion of continuous signal to, 487
decaying exponential, 8
decaying sinusoidal, 12
digitized, 3

even, 21
example of, 2
finite-duration, 366
Fourier transform and, 159, 373, 395
growing exponential, 8, 10
impulse, 6
odd, 21
Parseval's relation for, 167
periodic, Fourier series of, 147
plot of, 143
ramp, 6, 7
real exponential, 7
representation of, 16, 21
shifted, 15
sinusoidal, 7, 9, 144
time constant, 23, 24
time invariance and, 58
time-scaled, 19
total energy in, 28
unbounded, 30
unit step, 4, 5, 198
z-transform of, 199
electromagnetic, 1
energy, 167
cross-correlation equations for, 88
spectrum density of, 395
exponential
decaying, MATLAB-generated, 38
MATLAB script to simulate, 37
finite duration, 87, 96, 374
Fourier series coefficients, 389
Fourier transform of, 365
frequency components in, 149, 592
frequency domain, 365
impulse, 31, 35, 111
discrete signals using, 18
Fourier transform of, 161
MATLAB-generated, 37
response to, 65
shifted, 65
input, 55, 57
change of magnitude of, 393
frequencies, 172, 572
MATLAB script, 580
noise associated with, 494
noncausal, 217
periodic, 39, 147
approximated, 148
average power in, 389
power, 30, 89
processing of in real-life situations, 487
processing toolbox, MATLAB, 438, 439
pulse, 16
radar, correlation, 394
ramp, 31
random
dc component, 577
use of MATLAB to generate, 409
reconstructed, 531
reflected, 18, 20, 163
representation of in real life, 31
sampled, 405, 507
sampling interval for, 196
scaled, 28
shifted step, 16
sinusoidal, 14, 31, 173
decaying, 12
exponentially modulated, 10
growing, 12
irregularly decaying modulated, 12
irregularly growing modulated, 12
MATLAB-generated, 39
MATLAB script to simulate, 37
model, 329
step, 31, 161
input, convolution between impulse response and, 341, 345
MATLAB-generated, 35, 36
total energy in, 391, 392, 393
z-transform of, 195, 221–222
Signal representation, 1–53
amplitude scaling, 20–21
basic operations on discrete signals, 25–28
addition and subtraction, 25
combined operations, 26–28
modulation, 25
scalar multiplication, 25
bounded and unbounded discrete signals, 30
complex periodic discrete signal, 11–15
discrete signal time constant, 23–25
energy and power discrete signals, 28–30
even and off discrete signal, 21–23
examples, 32–50
exponentially modulated sinusoidal signal, 11
impulse discrete signal, 6
periodic and nonperiodic discrete signals, 3–4
problems, 50–53
ramp discrete signal, 6
real exponential discrete signal, 7
reason for discretizing continuous systems, 2–3
reflection operation, 18
representing discrete signal using impulses, 16–18
shifting operation, 15–16
signals in real world, 30–32
impulse signal, 31
other signals, 32
ramp signal, 31–32
sinusoidal signal, 31

step signal, 31
sinusoidal discrete signal, 7–11
time scaling, 19–20
unit step discrete signal, 4–5
Sindex, 42
Sinusoidal discrete signal, 7, 9
Sinusoidal input signal, 173
Sinusoidal response, 108
Sinusoidal signal, 14, 31
decaying, 12
exponentially modulated, 10
growing, 12
irregularly decaying modulated, 12
irregularly growing modulated, 12
MATLAB-generated, 39
MATLAB script to simulate, 37
model, 329
Sixth-order filter, 554
Spectral energy estimate, calculation of, 410
Square-magnitude response expression, 431
Stable system, 63
Starting index, 41, 98
State equation, 298, 300, 306, 313
discrete state-space approximation, 505
in matrix form, 319
obtaining of by inspection, 339
State matrix equations, 337
State-space and discrete systems, 265–328
examples, 292–322
general representation of systems in state-space, 270–283
block diagram to state-space, 273–275
nonrecursive systems, 272–273
recursive systems, 270–272
transfer function $H(z)$ to state-space, 276–283
general solution of state equation in real-time, 284–285
poles and stability, 291–292
problems, 322–328
properties of $\mathbf{A}''$ and evaluation, 285–289
review on matrix algebra, 266–270
adding two matrices, 267
definition, general terms, and notations, 266
determinant of two-by-two matrix, 268
diagonal form of matrix, 269
eigenvalues of matrix, 269
eigenvectors of matrix, 269–270
identity matrix, 266–267
inverse of matrix, 268
matrix multiplication, 269
multiplying matrix by constant, 267
subtracting two matrices, 267
transpose of matrix, 268
solution of state-space equations in z-domain, 283–284
transformations for state-space representations, 289–291
State-space matrix system, MATLAB script, 347
State-space model, 346, 355
State-space representation, 310, 343
discretization of, 504
impulse response, 355–356
transformations for, 289
State-space system matrices, 352
State values, use of z-transform method to find, 303
State vector, 289, 306, 504
Laplace transform, 520
solution, 285, 504, 505
terms of, 302
Steady-state response, 150, 153, 154, 185
Step input, 240
response plot due to, 240
signal, convolution between impulse response and, 341, 345
Step invariance transformation, 512, 513
Step response, 108, 241, 242, 312
difference equation, 350
impulse response model, 350
state-space model, 350
transfer function model, 350
use of MATLAB to find, 311, 320, 347, 348
Step signal, 31, 35, 36, 161
Stopband
attenuation, minimum, 425
ripple, maximum, 424
specifications, 430
Summation equation, 604
Summing junction, 79, 81–82
Superposition, 178
principle, 151
response due to inputs using, 316
Symmetry property, frequency response, 157
System
auxiliary equation of, 220
bounded-input bounded-output, 63
causal, 61, 608
characteristic equation, 69, 70, 123, 220
continuous, 506
differential equation, 522
impulse response, 527
input-output relationship, 507
oscillatory plot of, 523
partial fraction expansion, 525
plots, 519
state-space, 524
transfer functions, 527
control, 488

definition of, 55
discrete
difference equations representing, 350–351
Fourier transform, input to, 371
with periodic inputs, 150
eigenvalues, 91, 221, 290, 301
first-order, 80, 90
frequency response for, 156, 184
function, numerator and denominator coefficients of, 438
highpass, 478
identification toolbox, MATLAB, 438
impulse response for, 67, 85, 155
initial condition vector, 311
inverse of, 62
linear, 75, 78, 173, 365
linear time-variant, 153
analysis and design of, 329
multiple-inputs multiple-outputs in, 346
output of, 173
system, 90, 329
transfer function of, 220
lowpass, 478
magnitude of, 154
matrix, 301, 333, 352
mechanical modeling of, 329
with memory, 60, 61
mixed, 488
noncausal, 62, 218, 235
nonlinear, 57, 58
nonrecursive, 591, 614
output, 130, 150, 167, 333
physical, difference equations of, 68
poles of, 220, 346
recursive, 614
second-order, 221
initial conditions, 502
output for, 90
stability, 90, 91, 221, 292
poles and, 252
use of MATLAB to check, 311
stable, 63, 75, 101, 236, 354
state-space, *see* State-space and discrete systems
steady-state output in, 153, 595
steady-state response of, 220
third-order, output for, 90
time invariant, 59, 65, 66
time variant, 60
transfer function, 212, 230, 290
unknown parameters of, 329
unstable, 78, 235, 292, 314
zeroes of, 221
zero input, 286

T

Thermal interferences, 3
Third-order systems, output for, 90
Third-order transfer function, 210
Time
ambiguity in, 492, 494
constant, discrete, 23, 24, 25
invariance, discrete signals and, 58
invariant system, 59, 65, 66
-scaled discrete signal, 19
scaling, 19, 41
variant system, 60
Time domain, 251
characteristics, 448
no ambiguity in, 495
Total energy
signal, 391, 392, 393
use of MATLAB to calculate, 392, 393
Transfer function(s), 218, 298
additional zeroes introduced for, 516
analogue filter design, 421
Bessel filter, 435
calculation of, 310, 340
Chebyshev filter, 429
comb filter, 624
continuous, 506, 515, 517
denominator coefficient of, 329
differentiator, 606
digital filter, 509
discrete, 518, 519
elliptic filter, 433
IIR filter, 554, 561, 568, 570
magnitude, calculation of, 445
model, 350
numerator coefficient of, 329
partial fraction expansion and, 318
phase angle of, 422
representation, 332, 335, 344, 347
as block diagrams, 210
MATLAB script, 357
partial fraction expansion, 357
roots of denominator in, 332
s-domain, 547
third-order, 210
use of MATLAB to find, 301
z-domain, 216, 230
Transformation, *see also specific types*
functions, 442
matrix, 289, 310
methods, 512
backward difference, 512, 514
bilinear, 512, 514
forward difference, 512, 514
impulse invariance, 512, 513, 515

step invariance, 512, 513
Transformations between continuous and discrete representations, 487–539
bilinear transformation and relationship between Laplace-domain and z-domain representations, 506–511
discretization of derivative operation, 500–503
discretization of state-space representation, 504–506
examples, 517–534
from binary code to continuous signal, 490
from continuous signal to binary code representation, 488–490
insights, 515–516
choice of sampling interval T_s, 515
effect of choosing T_s on dynamics of system, 515–516
introduction of additional zeroes for transfer function $H(z)$, 516
need for converting continuous signal to discrete signal, 487–488
other transformation methods, 5515
backward difference method, 512
bilinear transformation, 512–515
forward difference method, 512
impulse invariance method, 512
step invariance method, 512
problems, 534–539
sampling operation, 490–500
ambiguity in frequency domain, 492–493
ambiguity in real-time domain, 490–492
filtering before sampling, 494–496
sampling and recovery of continuous signal, 496–500
sampling theorem, 493
Transition matrix, 285
continuous system, 525
Laplace transform, 520
sampling time, 505
use of MATLAB to verify entries in, 304
Transmission matrix, 504
Trigonometric identities, 111

U

Unique set, 148
Unit circle, 248, 253, 301
Unit step discrete signal, 4, 5, 198

V

Variable coefficients realization, 596
Voltage
divider, 472
sources, AC, 31
value, 1

W

Warping, 545, 546
Waveform generation, 487
Window(s)
Blackman, 603, 616
default, 634
definition, 600
FIR filter design using, 611
Hamming, 396, 603, 616, 630
Hanning, 396, 603
magnitude frequency response plot of, 616
MATLAB script, 402
Kaiser, 603, 604
rectangular, 603, 630
filter design using, 618
magnitude frequency response plot of, 616
use of in filter design, 602

Z

z-domain
design of IIR filters in, 550
input in, 213
multiplication in, 225
output in, 213, 314
representations, bilinear transformation and, 506
state vector in, 333
Zero input, response to system with, 286
Zero-order hold method, 532
Zero padding, 414, 415
Zero-pole-gain form, 441, 446
z-transform and discrete systems, 195–263
bilateral z-transform, 195–197
convergence, 200–203, 216–218
exercises, 221–255
final value theorem, 219
initial-value theorem, 219–220
inverse z-transform, 203–207
long division, 206–207
partial fraction expansion, 203–206
poles and zeroes, 220–221
poles of system 220
stability of system, 221
zeroes of system, 221
problems, 255–263
properties of z-transform, 207–210

convolution, 210
linearity property, 207
multiplication by e^{-an}, 209
shifting property, 207–209
representation of transfer functions as block diagrams, 210–212
solving difference equation using *z*-transform, 214–216
unilateral *z*-transform, 197–200
$x(n)$, $h(n)$, $y(n)$, and *z*-transform, 212–214