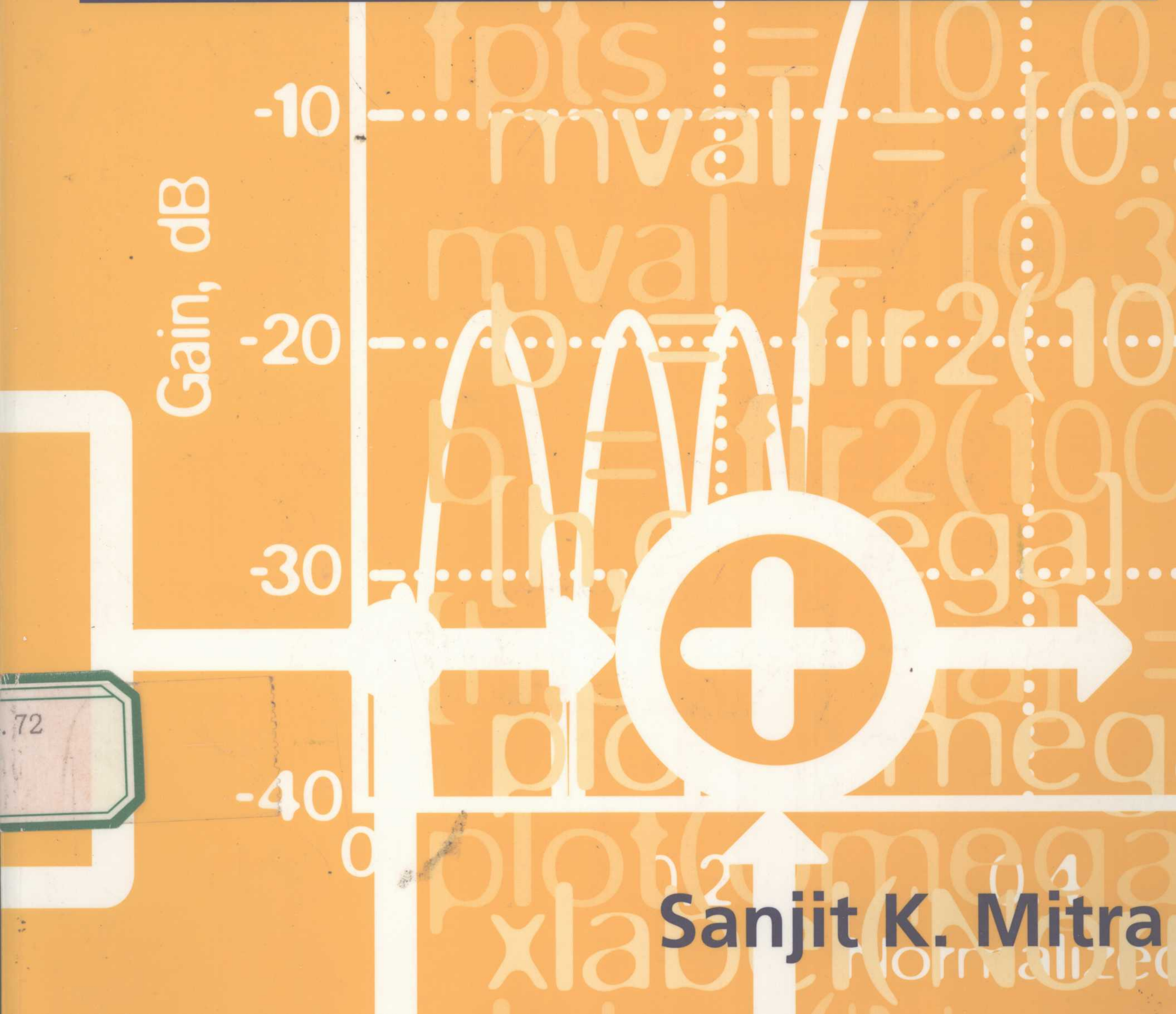
Digital Signal Processing Laboratory Using MATLAB®



Sanjit K. Mitra

Digital Signal Processing Laboratory using MATLAB®

Sanjit K. Mitra
University of California at Santa Barbara

WCB McGraw-Hill

Boston Burr Ridge, IL Dubuque, IA Madison, WI New York San Francisco St. Louis
Bangkok Bogotá Caracas Lisbon London Madrid
Mexico City Milan New Delhi Seoul Singapore Sydney Taipei Toronto

WCB/McGraw-Hill

A Division of The **McGraw-Hill** *Companies*

DIGITAL SIGNAL PROCESSING LABORATORY USING MATLAB®

The composition of the text was prepared by Sanjit K. Mitra.
The cover design was prepared by Goutam K. Mitra.

6 7 8 9 BKM BKM 0 9 8 7 6 5 4 3 2

Manual ISBN 0-07-232246-2
Package ISBN 0-07-232876-2

http://www.mhhe.com

Preface

Digital signal processing (DSP) is concerned with the representation of signals as a sequence of numbers and the algorithmic operations carried out on the signals to extract specific information contained in them. In barely 30 years the field of digital signal processing has matured considerably due to the phenomenal growth in both research and applications, and almost every university is now offering at least one or more courses at the upper division and/or first-year graduate level on this subject. With the increasing availability of powerful personal computers and workstations at affordable prices, it has become easier to provide the student with a practical environment to verify the concepts and the algorithms learnt in a lecture course.

This book is for a computer-based DSP laboratory course that supplements a lecture course on the subject. It includes 11 laboratory exercises with each exercise containing a number of projects to be carried out on a computer. The total number of projects may be more than what can be completed in a quarter- or a semester-long course assuming a three-hour per week laboratory. It is suggested that the instructor select pertinent projects that are more relevant to the lecture course he/she is teaching. If the computer laboratory is open for longer hours, it is recommended that the student be encouraged to come to the laboratory for longer periods of time to enable him/her to complete all projects.

The programming language used in this book is MATLAB,[1] widely used for high-performance numerical computation and visualization. The book assumes that the reader has no background in MATLAB and teaches him/her through tested programs in the first half of the book the basics of this powerful language in solving important problems in signal processing. In the second half of the book the student is asked to write the necessary MATLAB programs to carry out the projects. I believe students learn the intricacies of problem solving with MATLAB faster by using tested, complete programs and later writing simple programs to solve specific problems. A short review of the key concepts and features of MATLAB is provided in Appendix A.

Altogether there are 75 MATLAB programs in the text that have been tested under version 5.2 of MATLAB and version 4.2 of the Signal Processing Toolbox. The programs listed in this book are not necessarily the fastest with regard to their execution speeds, nor are they the shortest. They have been written for maximum clarity without detailed explanations. This book includes a diskette containing all MATLAB programs for the PC

[1]MATLAB is a registered trademark of The Mathworks, Inc., 24 Prime Park Way, Natick, MA 01760-1500, phone: 508-647-7000, **http://www.mathworks.com**.

running Windows 95/98, the Macintosh PowerPC running Mac OS 7 or higher and UNIX workstations. All programs are also available via anonymous FTP from the Internet site **iplserv.ece.ucsb.edu** in the directory **/pub/mitra/Labs**.

Each laboratory exercise contains a number of projects for the student to implement on their computers. Each project is followed by a series of questions the students must answer before embarking on the following project. These questions are designed to teach the student fundamentals of MATLAB and also the key concepts of DSP. For the latter part, each exercise includes a section summarizing the materials necessary for a quick review of DSP materials necessary to carry out the projects included in the exercise. For further details and explanations, each exercise includes at the end a list of DSP texts with specific chapter and/or section numbers. Each exercise also includes a section summarizing the MATLAB commands used to enable the student to find out more about one or more of these commands, if necessary, through the `help` command. A brief explanation of all MATLAB functions used in this book is given in Appendix B.

A novel feature of this book is the inclusion of partially written report documents for each of the first 10 laboratory exercises in the diskette provided. These reports are written in Microsoft Word. The students fill in the space provided answers to the questions as they proceed through the projects. This feature permits the students to complete more work in a specified amount of time than that would have been possible without it. The answers of the students should appear in a different font to make it easier for the laboratory instructor to evaluate the student's work. The completed reports also can serve as a guide for writing reports in other laboratory courses.

This book has evolved from teaching a laboratory component to an upper- division course on digital signal processing at the University of California, Santa Barbara, for the last 10 years. I would like to thank Drs. Stefan Thurnhofer and Ing-Song Lin for their assistance in developing the preliminary version of the laboratory course materials. I also would like to thank the students who took the upper division course and provided valuable comments that have improved the contents and style of the laboratory portion of the course. The complete manuscript of this book has been reviewed by Professor Hrvoje Babic of the University of Zagreb, Zagreb, Croatia; Professor Tamal Bose of the University of Colorado, Denver; Professor Ulrich Heute of the University of Kiel, Kiel, Germany; Professor Ottar Johnsen of the Ecole d'Ingénieurs de Friboug, Fribourg, Switzerland; Professor Abul N. Khondker of the Clarkson University, Potsdam, New York; Professor V. John Mathews of the Utah State University and Professor Yao Wang of the Polytechnic University, Brooklyn, New York. I thank them for their valuable comments. I thank my students, Rajeev Gandhi, Michael S. Moore and Debargha Mukherjee, for their assistance in proofreading of the manuscript and checking all the programs included in this book. I also acknowledge with gratitude the support of the Office of Instructional Development at the University of California, Santa Barbara, for providing me with two instructional improvement grants to develop the laboratory course. Finally, I thank my son Goutam for the cover design of my book.

Every attempt has been made to ensure the accuracy of all materials in this book, including the MATLAB programs. I would, however, appreciate readers bringing to my attention any errors that may have appeared in the printed version for reasons beyond my control and that of the publisher. These errors and any other comments can be communicated to me by e-mail addressed to: **mitra@ece.ucsb.edu**.

Santa Barbara

Sanjit K. Mitra

Contents

Discrete-Time Signals in the Time-Domain 1

1.1 Introduction

Digital signal processing is concerned with the processing of a discrete-time signal, called the *input signal*, to develop another discrete-time signal, called the *output signal*, with more desirable properties. In certain applications, it may be necessary to extract some key properties of the original signal using specific digital signal processing algorithms. It is also possible to investigate the properties of a discrete-time system by observing the output signals for specific input signals. It is thus important to learn first how to generate in the time-domain some basic discrete-time signals in MATLAB and perform elementary operations on them, which are the main objectives of this first exercise. A secondary objective is to learn the application of some basic MATLAB commands and how to apply them in simple digital signal processing problems.

1.2 Getting Started

The diskette provided with this book contains all of the MATLAB programs and the partially written reports for both the PC and the Macintosh PowerPC. In particular, it includes both PC and Macintosh versions of the MATLAB M-files of the first 10 exercises in folders grouped by chapters and report documents written in Microsoft Word in folders also grouped by chapters. After the completion of a project of a laboratory exercise, you record in the report of that exercise the answers to each question referring to this project at their designated locations.

Installation Instructions for a PC

To copy the program and the report folders onto the hard disk of a PC running Windows 95/98 follow the steps given below:

1. Insert the floppy disk.
2. Open the **My Computer** window by double-clicking on its icon displayed on the Desktop.
3. Open the window of the floppy drive by double-clicking on its icon.
4. Open the window of the desired hard drive by double-clicking on its icon. Depending on your setup, it may be necessary to open another window by double-

clicking on **My Computer** icon before you can select the destination hard drive.

5. In the floppy drive window, select the folder marked **PC** and drag it to the directory displayed in the hard drive window where you would like to copy the files.

Installation Instructions for a Macintosh PowerPC

To copy the program and the report folders on the hard disk of a PowerPC running Mac OS or higher follow the steps given below:

1. Insert the floppy disk.
2. Open the hard drive window by double-clicking on its icon displayed on the Desktop.
3. Open the window of the floppy drive by double-clicking on its icon.
4. In the floppy drive window, select the folder marked **MAC** and drag it to the directory displayed in the hard drive window where you would like to copy the files.

Downloading via FTP and the World Wide Web

The FTP site for downloading the files to a computer is **iplserv.ece.ucsb.edu**. The directories containing the files for the PC, Macintosh PowerPC, and UNIX workstations are as follows:

pub/mitra/Labs/pc
pub/mitra/Labs/mac
pub/mitra/Labs/unix (M-files only)

To download the files from the FTP site to your PC or UNIX computer, follow the steps given below:

1. Open the FTP program for your computer.
2. Type **ftp iplserv.ece.ucsb.edu**.
3. At the **Login:** prompt, type **anonymous**.
4. At the **Password** prompt, type **Your E-mail address**.
5. To retrieve the PC files type **cd pub/mitra/Labs/pc**. To retrieve the UNIX files type **cd pub/mitra/Labs/unix**. These commands will get you to the directory where the M-files and the report files can be found.
6. Type **cd directory name** to change to the directory of the desired M-files or report files.
7. To download ASCII files such as the M-files, type **asc**. To download binary files such as the Word report documents (for the PC), type **bin**. To download a desired file type **get file name**. The last command will place the desired file in the current directory on your local system.

8. You can download another file using the **get** command or you can change back to the previous directory by typing **cd** followed by two periods.
9. When finished downloading all of the desired files, type **bye**.

To download the files from the FTP site to your Macintosh computer, a variety of FTP programs are available. The steps given below are for the **Fetch** program from Dartmouth College.

1. Open the FTP program.
2. Enter the following information in the **New Connection** dialog box. If you do not see this dialog box, open it by choosing **New Connection** from the **File** menu.

   `Host:` **iplserv.ece.ucsb.edu**
   `User ID:` **anonymous**
   `Password:` **Your E-mail address**
   `Directory` **/pub/mitra/Labs/mac**

3. Choose **OK**. You should see a window with the contents of the directory of the FTP site. You can change directories simply by double-clicking on the folder you want.
4. To transfer a file, select the file you want by double-clicking on it. A dialog box will pop up. Select the place on your computer where you want to store the file and choose **Save**. In recent versions of the **Fetch** program, entire directories can be downloaded at once by selecting the directory name and choosing **Get**.

For downloading via world wide web, follow the steps given below:

1. Open the available web browser.
2. Type **ftp://iplserv.ece.ucsb.edu** in the URL window.
3. Double-click on the desired directory (the directory for the PC and Macintosh versions are shown above).
4. Double-click on the desired file for downloading. You will get a dialog box asking where you would like to save the file.

1.3 Background Review

R1.1 A discrete-time signal is represented as a sequence of numbers, called *samples* . A sample value of a typical discrete-time signal or sequence $\{x[n]\}$ is denoted as $x[n]$ with the argument n being an integer in the range $-\infty$ and ∞. For convenience, the sequence $\{x[n]\}$ is often denoted without the curly brackets.

R1.2 The discrete-time signal may be a finite length or an infinite length sequence. A finite length (also called *finite duration* or *finite extent*) sequence is defined only for a finite time interval:

$$N_1 \leq n \leq N_2, \tag{1.1}$$

where $-\infty < N_1$ and $N_2 < \infty$ with $N_2 \geq N_1$. The length or duration N of the finite length sequence is

$$N = N_2 - N_1 + 1. \tag{1.2}$$

R1.3 A sequence $\tilde{x}[n]$ satisfying

$$\tilde{x}[n] = \tilde{x}[n + kN] \qquad \text{for all } n, \tag{1.3}$$

is called a *periodic sequence* with a period N where N is a positive integer and k is any integer.

R1.4 The *energy* of a sequence $x[n]$ is defined by

$$\mathcal{E} = \sum_{n=-\infty}^{\infty} |x[n]|^2. \tag{1.4}$$

The energy of a sequence over a finite interval $-K \leq n \leq K$ is defined by

$$\mathcal{E}_K = \sum_{n=-K}^{K} |x[n]|^2. \tag{1.5}$$

R1.5 The *average power* of an aperiodic sequence $x[n]$ is defined by

$$\mathcal{P}_{av} = \lim_{K\to\infty} \frac{1}{2K+1}\mathcal{E}_K = \lim_{K\to\infty} \frac{1}{2K+1} \sum_{n=-K}^{K} |x[n]|^2. \tag{1.6}$$

The average power of a periodic sequence $\tilde{x}[n]$ with a period N is given by

$$\mathcal{P}_{av} = \frac{1}{N} \sum_{n=0}^{N-1} |\tilde{x}[n]|^2. \tag{1.7}$$

R1.6 The *unit sample sequence* , often called the *discrete-time impulse* or the *unit impulse*, denoted by $\delta[n]$, is defined by

$$\delta[n] = \begin{cases} 1, & \text{for } n = 0, \\ 0, & \text{for } n \neq 0. \end{cases} \tag{1.8}$$

The *unit step sequence* , denoted by $\mu[n]$, is defined by

$$\mu[n] = \begin{cases} 1, & \text{for } n \geq 0, \\ 0, & \text{for } n < 0. \end{cases} \tag{1.9}$$

R1.7 The *exponential sequence* is given by

$$x[n] = A\alpha^n, \tag{1.10}$$

where A and α are real or complex numbers. By expressing

$$\alpha = e^{(\sigma_o + j\omega_o)}, \quad \text{and} \quad A = |A|e^{j\phi},$$

we can rewrite Eq. (1.10) as

$$x[n] = |A|e^{\sigma_o n + j(\omega_o n + \phi)} = |A|e^{\sigma_o n}\cos(\omega_o n + \phi) + j|A|e^{\sigma_o n}\sin(\omega_o n + \phi). \tag{1.11}$$

R1.8 The *real sinusoidal sequence* with a constant amplitude is of the form

$$x[n] = A\cos(\omega_o n + \phi), \tag{1.12}$$

where A, ω_o, and ϕ are real numbers. The parameters A, ω_o, and ϕ in Eqs. (1.11) and (1.12) are called, respectively, the *amplitude* , the *angular frequency* , and the *initial phase* of the sinusoidal sequence $x[n]$. $f_o = \omega_o/2\pi$ is the *frequency*.

R1.9 The complex exponential sequence of Eq. (1.11) with $\sigma_o = 0$ and the sinusoidal sequence of Eq. (1.12) are periodic sequences if $\omega_o N$ is an integer multiple of 2π, that is,

$$\omega_o N = 2\pi r, \tag{1.13}$$

where N is a positive integer and r is any integer. The smallest possible N satisfying this condition is the *period* of the sequence.

R1.10 The *product* of two sequences $x[n]$ and $h[n]$ of length N yields a sequence $y[n]$, also of length N, as given by

$$y[n] = x[n] \cdot h[n]. \tag{1.14}$$

The *addition* of two sequences $x[n]$ and $h[n]$ of length N yields a sequence $y[n]$, also of length N, as given by

$$y[n] = x[n] + h[n]. \tag{1.15}$$

The *multiplication* of a sequence $x[n]$ of length N by a scalar A results in a sequence $y[n]$ of length N as given by

$$y[n] = A \cdot x[n]. \tag{1.16}$$

The *time-reversal* of a sequence $x[n]$ of infinite length results in a sequence $y[n]$ of infinite length as defined by

$$y[n] = x[-n]. \tag{1.17}$$

The *delay* of a sequence $x[n]$ of infinite length by a positive integer M results in a sequence $y[n]$ of infinite length given by

$$y[n] = x[n - M]. \tag{1.18}$$

If M is a negative integer, the operation indicated in Eq. (1.18) results in an *advance* of the sequence $x[n]$.

A sequence $x[n]$ of length N can be *appended* by another sequence $g[n]$ of length M resulting in a longer sequence $y[n]$ of length $N + M$ given by

$$\{y[n]\} = \{\{x[n]\}, \{g[n]\}\}. \tag{1.19}$$

1.4 MATLAB Commands Used

The MATLAB commands you will encounter in this exercise are as follows:

Operators and Special Characters

```
:    .    +    -    *    /    ;
%
```

Elementary Matrices and Matrix Manipulation

```
i    ones    pi    rand    randn    zeros
```

Elementary Functions

```
cos    exp    imag    real
```

Data Analysis

```
sum
```

Two-Dimensional Graphics

```
axis    grid    legend    plot    stairs
stem    title   xlabel    ylabel
```

General Purpose Graphics Functions

```
clf    subplot
```

Signal Processing Toolbox

```
sawtooth    square
```

For additional information on these commands, see the *MATLAB Reference Guide* [Mat94] and the *Signal Processing Toolbox User's Guide* [Mat96] or type `help commandname` in the Command window. A brief explanation of the MATLAB functions used here can be found in Appendix B.

1.5 Generation of Sequences

The purpose of this section is to familiarize you with the basic commands in MATLAB for signal generation and for plotting the generated signal. MATLAB has been designed to operate on data stored as vectors or matrices. For our purposes, sequences will be stored as vectors. Therefore, all signals are limited to being causal and of finite length . The steps to follow to execute the programs listed in this book depend on the platform being used to run the MATLAB.

MATLAB on the Windows PC

The program can be executed by typing the name of the program without `.m` in the Command window and hitting the carriage return. Alternately, choose **Open** from the **File** menu in the Command window and choose the desired M-file. This opens the M-file in the Editor/Debugger window in which an M-file can be executed using the **Run** command under the **Tools** menu.

MATLAB on the Macintosh

The program can be executed by typing the name of the program without `.m` in the Command window and hitting the carriage return. Alternately, it can be copied into the Editor Window by using the **Open M-File** command on your screen and then choosing the **Save and Execute** command on your screen.

Project 1.1 Unit Sample and Unit Step Sequences

Two basic discrete-time sequences are the unit sample sequence and the unit step sequence of Eqs. (1.8) and (1.9), respectively. A unit sample sequence `u[n]` of length `N` can be generated using the MATLAB command

$$\texttt{u} = [\texttt{1} \quad \texttt{zeros(1,N}-\texttt{1)}];$$

A unit sample sequence `ud[n]` of length `N` and delayed by `M` samples, where `M < N`, can be generated using the MATLAB command

$$\texttt{ud} = [\texttt{zeros(1,M)} \quad \texttt{1} \quad \texttt{zeros(1,N}-\texttt{M}-\texttt{1)}];$$

Likewise, a unit step sequence `s[n]` of length `N` can be generated using the MATLAB command

$$\texttt{s} = [\texttt{ones(1,N)}];$$

A delayed unit step sequence can be generated in a manner similar to that used in the generation of a delayed unit sample sequence.

Program P1_1 can be used to generate and plot a unit sample sequence.

```
% Program P1_1
% Generation of a Unit Sample Sequence
clf;
% Generate a vector from -10 to 20
n = -10:20;
% Generate the unit sample sequence
u = [zeros(1,10) 1 zeros(1,20)];
% Plot the unit sample sequence
stem(n,u);
xlabel('Time index n');ylabel('Amplitude');
title('Unit Sample Sequence');
axis([-10 20 0 1.2]);
```