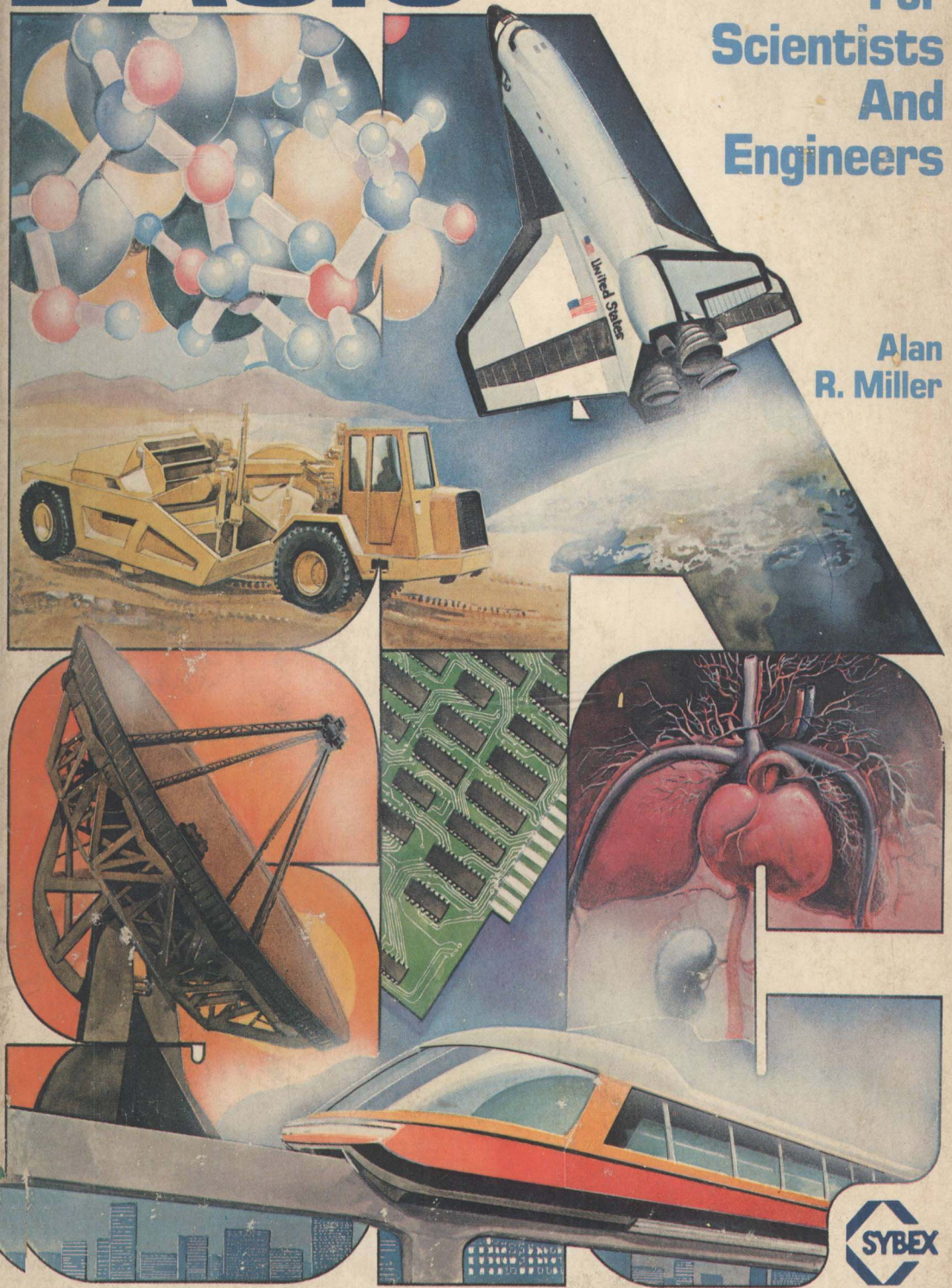


BASIC Programs

For
Scientists
And
Engineers

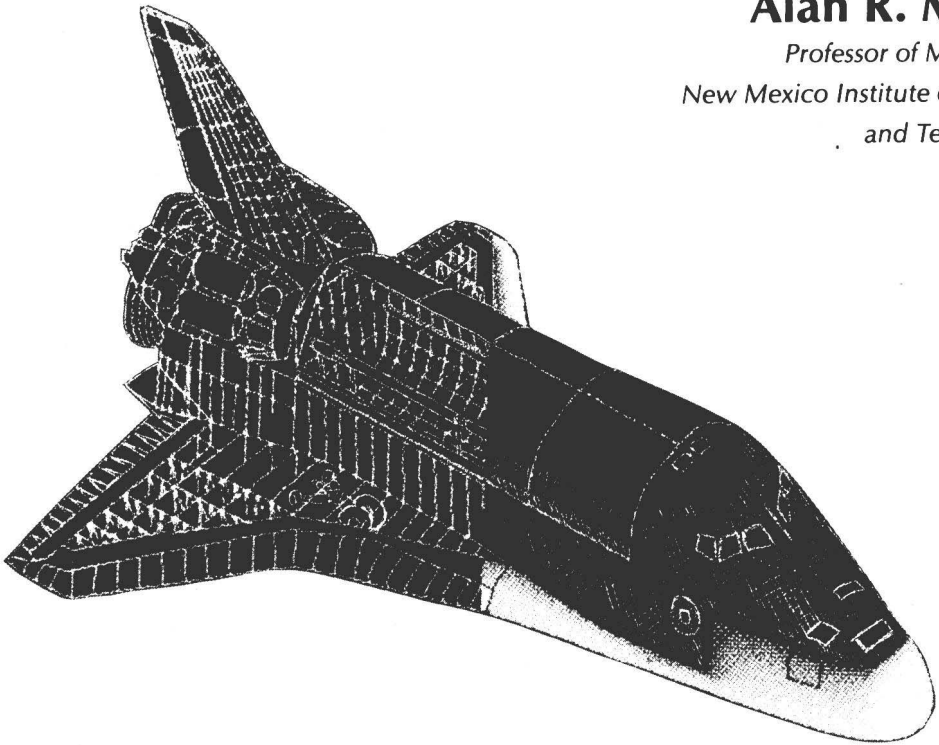
Alan
R. Miller



BASIC PROGRAMS for Scientists and Engineers

Alan R. Miller

*Professor of Metallurgy
New Mexico Institute of Mining
and Technology*



Berkeley • Paris • Düsseldorf

CREDITS

Cover Design by Daniel Le Noury

Technical illustrations by J. Trujillo Smith, Jeanne E. Tennant

NOTICES

Apple is a registered trademark of Apple Computer Corporation.

BASIC-80, Microsoft BASIC, and BASCOM are registered trademarks of Microsoft Consumer Products.

CBASIC is a registered trademark of Compiler Systems, Inc.

DEC-20 is a registered trademark of Digital Equipment Corporation.

Lifeboat 2.2 is a registered trademark of Lifeboat Associates.

North Star BASIC is a registered trademark of North Star Computers, Inc.

PET is a registered trademark of Commodore Business Machines.

TRS-80 is a registered trademark of Tandy Corporation.

Xitan BASIC is a registered trademark of Xitan Systems Ltd.

WordStar is a registered trademark of MicroPro International Corporation.

Z80 is a registered trademark of Zilog, Inc.

SYBEX is not affiliated with any manufacturer.

Every effort has been made to supply complete and accurate information. However, Sybex assumes no responsibility for its use, nor for any infringements of patents or other rights of third parties which would result.

Copyright © 1981 SYBEX Inc. World Rights reserved. No part of this publication may be stored in a retrieval system, transmitted, or reproduced in any way, including but not limited to photocopy, photograph, magnetic or other record, without the prior agreement and written permission of the publisher.

Library of Congress Card Number: 81-84003

ISBN 089588-073-3

First Edition 1981

Printed in the United States of America

10 9 8 7 6 5 4 3 2

BASIC PROGRAMS
for Scientists
and
Engineers

Preface

The ideas and material for this book have been developed during my experience teaching sophomore, junior, and senior engineering students over the past 14 years. I have used FORTRAN, BASIC, and Pascal as the computer languages for these courses.

All of the BASIC programs in this book were developed on a Z80 microcomputer. The operating system was the Lifeboat 2.2 version of CP/M. The source programs were developed and executed with Micro-soft BASIC-80, Version 5. MicroPro's WordStar was frequently used to modify the ASCII form of the source programs. Most of the programs were also run on other BASICs. These included Microsoft's BASCOM (compiling BASIC), CBASIC, Xitan BASIC, North Star BASIC, and BASIC Plus-4 running on a DEC-20.

The manuscript was created and edited with MicroPro's WordStar running on the same Z80 computer. The BASIC source programs have been incorporated directly into the manuscript from the original source files. Computer printouts shown in the figures were also incorporated magnetically into the manuscript. This was accomplished by altering the CP/M operating system so that console output was written into a block of memory. This block was then saved as a disk file. The final manuscript was submitted to SYBEX in a magnetic form compatible with the photocomposer. Consequently, the manuscript and the BASIC source programs have not been retyped.

I am sincerely grateful for the helpful guidance and suggestions of Rudolph Langer and Douglas Hergert during the development of the manuscript. I would also like to thank Ashok Singh for checking the manuscript, especially the mathematical expressions.

Alan R. Miller
Socorro, New Mexico
August 1981

Introduction

The purpose of this book is twofold: to help the reader develop a proficiency in the use of BASIC, and to provide a library of programs useful for solving problems frequently encountered in science and engineering.

The programs in this book, the second in the SYBEX *Programs for Scientists and Engineers* series, are most valuable to the practicing scientist or engineer. The material is also suitable for a junior- or senior-level engineering course in numerical methods. The reader should have a working knowledge of an applications language such as BASIC, FORTRAN or Pascal. Experience with vector operations and differential and integral calculus will also be helpful.

BASIC is currently the most widely available high-level computer language, especially on microcomputers. Because it is usually implemented as an interpreter, it is a particularly easy language to use: program execution can be interrupted at any point, current values can be printed, and then execution can be resumed.

The lowest level of BASIC is extremely limited. Variable names should be no longer than two characters, and looping control is only available with the **FOR /NEXT** and **GOTO** constructions. Fortunately, however, powerful BASICs are becoming more common. These contain features such as long variable names, the **WHILE/WEND** construction and integer typing of variables. The programs in this book were initially developed using a powerful set of BASIC features, and then simplified to the lowest common denominator. Thus, they can be run on all of the common BASICs. In fact, these programs will run on microcomputers such as the Apple, the TRS-80, and the PET, on Microsoft BASIC and on CBASIC. In addition, they will run as well on larger computers such as the DEC-20 with only minor changes to features such as the random number generator and the multiple statement separator. Users are urged to upgrade the programs to the highest possible level. To help with this

conversion, suggested alternate names are given in the source programs, using **REM** lines such as the following:

```

11 REM identifiers
12
13 REM N1%  NROW%  number of rows
14 REM N2%  NCOL%  number of columns
15 REM end of identifiers

```

In lines 13 and 14, the two-character variable names used in the program are followed by longer and more descriptive names suitable for less restrictive BASICs, and then by comments identifying the variables.

The line numbers for the programs in this book have been integrated so that there is a minimum of conflict. It should thus be possible to combine any of the different subroutines into a single program. Duplicate line numbers are only utilized for the same kinds of tasks. The assigned blocks of line numbers are detailed below:

1 - 499	Main program
500 - 880	Input routine
800 - 880	Set up matrix
1000 - 1370	Output routine
2000 - 2410	Numerical integration
2500 - 2810	The error function
3000 - 3550	Sorting routine
3700 - 3790	Conversion to upper case
4000 - 4310	Conversion to square matrix
4400 - 4930	Bessel function of the second kind
5000 - 6150	Matrix solution
7000 - 7990	Plot routine
8000 - 8160	Newton's method
8400 - 8430	Function evaluation
8500 - 8780	Bessel function of the first kind
9000 - 9100	Mean and standard deviation
9200 - 9400	Gamma function
9500 - 9580	Gaussian random numbers
9800 - 9840	Normally distributed random numbers
9999	END statement

The reader who is primarily interested in the BASIC programs developed in this book will have no trouble locating them; the sections that contain programs or subroutines are clearly labeled. It should be

noted, however, that this book is designed to be read from beginning to end. Each chapter discusses and develops tools that will be used again in subsequent chapters. The mathematical algorithms of each program are methodically described before the program itself is implemented, and sample output is supplied for most of the programs. The following brief descriptions summarize the contents of each chapter.

Chapter 1, ***Evaluation of a BASIC Interpreter or Compiler***, identifies weak points in several commercial BASICs, and supplies programs for testing any BASIC. The results will be used to select various constants and operations in later chapters. Also included in Chapter 1 are discussions of the common “NEXT without FOR” bug and the “10 GOSUB 10” problem.

Chapter 2, ***Mean and Standard Deviation***, discusses some simple statistical algorithms and presents a program for implementing them. Routines for generating—and testing—both uniform and Gaussian random numbers are also given.

Chapter 3, ***Vector and Matrix Operations***, summarizes the operations of vector and matrix arithmetic, including dot product, cross product, matrix multiplication and matrix inversion. Two important programs are developed—one for carrying out matrix multiplication, and another for calculating determinants.

Chapter 4, ***Simultaneous Solution of Linear Equations***, presents programs to carry out the algorithms of Cramer’s rule, the Gauss elimination method, the Gauss-Jordan elimination method, and the Gauss-Seidel method—all for solving simultaneous equations. In addition, ill conditioning is studied by observing a program that generates Hilbert matrices, and a program is developed for solving equations with complex coefficients.

Chapter 5, ***Development of a Curve-Fitting Program***, is the first of a series of chapters on curve fitting. In a good illustration of top-down program development, a linear least-squares curve-fitting program is written and discussed. The program includes routines to simulate data, plot curves, compute the fitted curve, and supply the correlation coefficient.

Chapter 6, ***Sorting***, describes and compares several BASIC sorting routines, including two bubble sorts, a Shell sort and a nonrecursive quick sort. A sort routine is incorporated into the curve-fitting program of Chapter 5 to enable the program to handle real experimental data.

Chapter 7, **General Least-Squares Curve Fitting**, extends the curve-fitting program to general polynomial equations, and finds curve fits for three examples: the equations for heat capacity, vapor pressure, and properties of superheated steam.

Chapter 8, **Solution of Equations by Newton's Method**, presents a series of programs that use Newton's algorithm for finding the roots of an equation. This tool will be used again in Chapter 10 for nonlinear curve fitting.

Chapter 9, **Numerical Integration**, develops programs for three different integration methods—the trapezoidal rule, Simpson's rule, and the Romberg method. End correction is also discussed. Simpson's rule will be used in Chapter 11 for evaluating the Gaussian error function.

Chapter 10, **Nonlinear Curve-Fitting Equations**, discusses curve-fitting algorithms for the rational function and the exponential function. Two examples are given—the Clausius factor, and the diffusion equation.

Chapter 11, **Advanced Applications: The Normal Curve, the Gaussian Error Function, the Gamma Function, and the Bessel Functions**, addresses several advanced topics in programming for mathematical applications. This last chapter summarizes and expands upon a number of the concepts presented earlier in the book.

Each chapter also contains exercises designed to extend the reader's comprehension of the material.

For readers who are approaching BASIC for the first time, a summary of the syntax, standard functions, and reserved words of BASIC is included in the appendices. The real educational experience of this book, however, will be gained by carefully working through the programs themselves.

A Note on Typography

Many readers of programming books believe that typeset programs are more likely to contain errors than photographed programs, that errors may be introduced into the programs during the typesetting process. At SYBEX, we have developed a method of typesetting programs (thus enhancing their legibility) without introducing errors. The author submits tested, executable source code on disks. The programs are then run through a formatting program on our in-house computers. The formatting program establishes conventions for spacing, capitalization, etc., so that the programs will have a uniform appearance, without altering their content. Next, the programs (along with the text) are transmitted electronically to our computerized phototypesetter. At no point are the programs actually retyped, so errors that might have been introduced by retyping have been avoided.

The text of this book has been set in the typeface known as Oracle, the programs are in Futura, and the output has been photographed from the actual line-printer output supplied by the author. Reserved words appear in **boldface**. Mathematical expressions (variables, letter constants) appear in *italics*, except for vectors, which appear in boldface roman. For example:

$$A + Bx + Cx^2 = 0$$

$$\mathbf{v} = [1 \ 2 \ 3]$$

Throughout the book an effort has been made to distinguish typographically between mathematical values and program structures. Thus, juxtaposed in a single paragraph, the reader may see references to the variable *x* and the BASIC variable *X*; the vector **v** and the BASIC array *V*; the matrix element v_{ij} and the BASIC array element *V(I,J)*.

Contents

	Preface	xi
	Introduction	xiii
	A Note on Typography	xvii
1	<u>Evaluation of a BASIC Interpreter or Compiler</u>	1
	Introduction	1
	Precision and Range of Floating-Point Operations	2
	Program: A Test of the Floating-Point Operations	2
	BASIC SIN Function	5
	BASIC Program: Testing the SIN Function	5
	BASIC Program: The "NEXT Without FOR" Bug	7
	A GOSUB Without RETURN	9
	Summary	9
	Exercises	10
2	<u>Mean and Standard Deviation</u>	13
	Introduction	13
	The Mean	14
	The Standard Deviation	16
	BASIC Program: Mean and Standard Deviation	18
	Random Numbers	21
	BASIC Function: A Random Number Generator	21
	BASIC Program: Evaluation of a Random Number Generator	22
	BASIC Program: Generating and Testing Gaussian Random Numbers	26
	Summary	29
	Exercises	30
3	<u>Vector and Matrix Operations</u>	33
	Introduction	33
	Scalars and Arrays	34

Vectors	34
Matrices	38
BASIC Program: Matrix Multiplication	43
Determinants	47
BASIC Program: Determinants	48
Inverse Matrices and Matrix Division	50
Summary	51
Exercises	51

4 *Simultaneous Solution of Linear Equations* 53

Introduction	53
Linear Equations and Simultaneous Equations	54
Solution by Cramer's Rule	55
BASIC Program: A More Elegant Use of Cramer's Rule	59
Solution by Gauss Elimination	62
BASIC Program: The Gauss Elimination Method	65
Solution by Gauss-Jordan Elimination	70
BASIC Program: Gauss-Jordan Elimination	72
Multiple Constant Vectors and Matrix Inversion	78
BASIC Program: Gauss-Jordan Elimination, Version Two	80
Ill-Conditioned Equations	88
BASIC Program: Solving Hilbert Matrices	90
A Simultaneous Best Fit	93
BASIC Program: The Best Fit Solution	94
Equations with Complex Coefficients	98
BASIC Program: Simultaneous Equations with Complex Coefficients	101
The Gauss-Seidel Iterative Method	105
BASIC Program: The Gauss-Seidel Method	106
Summary	112
Exercises	113

5 *Development of a Curve-Fitting Program* 115

Introduction	115
The Main Program	116
A Printer Plotter Routine	121
The Curve-Fitting Algorithm	128
The Correlation Coefficient	134

BASIC Program: Least-Squares Curve-Fitting for Simulated Data	137
Summary	140
Exercises	141

6

Sorting

143

Introduction	143
Handling Experimental Data	144
A Bubble Sort	145
BASIC Program: The Bubble Sort	145
BASIC Program: Bubble Sort with SWAP Function	149
A Shell Sort	150
BASIC Program: The Shell-Metzner Sort	150
The Quick Sort	151
BASIC Program: A Nonrecursive Quick Sort	152
Sorting Disk Data	156
Incorporating Sort into the Curve-Fitting Program	159
Summary	159
Exercises	160

7

General Least-Squares Curve Fitting

163

Introduction	163
A Parabolic Curve Fit	164
BASIC Program: Least-Squares Curve Fit for a Parabola	165
Curve Fits for Other Equations	171
BASIC Program: The Matrix Approach to Curve Fitting	173
BASIC Program: Adjusting the Order of the Polynomial	179
BASIC Program: The Heat-Capacity Equation	183
BASIC Program: The Vapor Pressure Equation	186
A Three-Variable Equation	190
BASIC Program: An Equation of State for Steam	191
Summary	198
Exercises	199

8

Solution of Equations by Newton's Method

203

Introduction	203
Formulating Newton's Method	204

BASIC Program: A First Attempt at Newton's Method	209
BASIC Program: Solving Other Equations	219
BASIC Program: The Vapor Pressure Equation	221
Summary	222
Exercises	223

9 *Numerical Integration* 225

Introduction	225
The Definite Integral	226
The Trapezoidal Rule	227
BASIC Program: The Trapezoidal Rule with User Input for the Number of Panels	229
BASIC Program: An Improved Trapezoidal Rule	230
BASIC Program: Trapezoidal Rule with End Correction	234
BASIC Program: Simpson's Integration Method	236
BASIC Program: The Simpson Method with End Correction	240
The Romberg Method	242
BASIC Program: Integration by the Romberg Method	243
Functions that Become Infinite at One Limit	248
BASIC Program: Adjustable Panels for an Infinite Function	248
Summary	251
Exercises	252

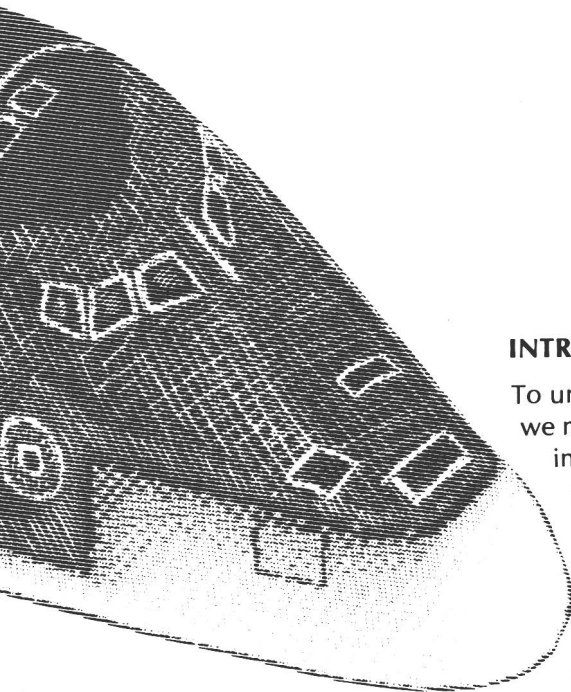
10 *Nonlinear Curve-Fitting Equations* 255

Introduction	255
Linearizing the Rational Function	256
BASIC Program: The Clausing Factor Fitted to the Rational Function	256
Linearizing the Exponential Equation	26
BASIC Program: An Exponential Curve Fit for the Diffusion of Zinc in Copper	261
Direct Solution of the Exponential Equation	265
BASIC Program: A Nonlinearized Exponential Curve Fit	267
Summary	273
Exercises	274

11	<i>Advanced Applications: The Normal Curve, the Gaussian Error Function, The Gamma Function, and the Bessel Function</i>	<i>277</i>
	Introduction	227
	The Normal and Cumulative Distribution Functions	278
	The Gaussian Error Function	280
	BASIC Program: Evaluating the Gaussian Error Function Using Simpson's Rule	282
	BASIC Program: Evaluating the Gaussian Error Function Using an Infinite Series Expansion	285
	The Complement of the Error Function	287
	BASIC Program: Evaluating the Complement of the Error Function	288
	The Gamma Function	291
	BASIC Program: Evaluation of the Gamma Function	292
	Bessel Functions	295
	BASIC Program: Bessel Functions of the First Kind	296
	BASIC Program: Bessel Functions of the Second Kind	298
	Summary	302
	Exercises	303
	<i>Appendix A: Reserved Words and Functions</i>	<i>305</i>
	<i>Appendix B: Summary of BASIC</i>	<i>307</i>
	The BASIC Character Set	307
	Variable Names	309
	Array Variables	308
	Constants	308
	Comments	308
	Operations	309
	Assignment Statements	310
	The Unconditional Branch	310
	Iterative Statements	311
	Input and Output	312
	Subroutines	312
	<i>Bibliography</i>	<i>315</i>
	<i>Index</i>	<i>317</i>

CHAPTER 1

Evaluation of a BASIC Interpreter or Compiler



INTRODUCTION

To understand the results of a BASIC program we must be familiar with the limitations of the interpreter or compiler we are using. This is particularly true with scientific application programs such as the ones given in this book. In this first chapter, then, we will present some tools for evaluating the precision and range of BASIC. In fact, the examples given here were derived from several popular BASICs.

PRECISION AND RANGE OF FLOATING-POINT OPERATIONS

Many of the programs in this work are sensitive to the *precision* and *dynamic range* of the BASIC floating-point operations. For example, in one program an algorithm is terminated when a particular term is smaller than a relative tolerance. The formula in this case is:

$$\text{TERM} < \text{SUM} * \text{TOL}$$

where TERM is the value of the new term, SUM is the current total, and TOL is an arbitrarily small number known as the *tolerance*.

It is important that the value chosen for the tolerance be within the accuracy of the floating-point operations. Otherwise, the summation step will never terminate. Suppose, for example, that the floating-point operations are performed to a precision of six significant figures. Then, the tolerance must be set to a value larger than 10^{-6} .

The dynamic range of the exponent is a separate matter. Typical binary, floating-point operations are performed with 32 bits of precision. BCD (Binary Coded Decimal) floating-point packages, on the other hand, will usually retain more significant figures and have a greater dynamic range.

We will now present a BASIC program for testing the precision and dynamic range. We will investigate output from several common BASICs to illustrate both mantissa and exponent accuracy.

BASIC PROGRAM: A TEST OF THE FLOATING-POINT OPERATIONS

The program given in Figure 1.1 can be used to determine the precision and the dynamic range of BASIC. Type up the program and execute it. The initial value of X is obtained by dividing $1.0\text{E}-4$ by 3. Then, successively smaller and smaller values of X are calculated and displayed

```

10  N% = 18
20  X = 1.0E-4 / 3
30  FOR I% = 1 TO N%
40    X = X/10
50    PRINT I%, X,
60    X = X/10
70    PRINT X
80  NEXT I%
90  END

```

Figure 1.1: A Test of the Floating - Point Operations