AN INTRODUCTION TO NUMERICAL METHODS FOR CHEMICAL ENGINEERS

James B. Riggs

AN INTRODUCTION TO NUMERICAL METHODS FOR CHEMICAL ENGINEERS

James B. Riggs
Department of Chemical Engineering
Texas Tech University

Texas Tech University Press
1988

Preface

Computer-implemented numerical methods are now used extensively in the undergraduate engineering curriculum. There are numerous reasons for this: (1) industry expects engineering graduates to be computer-literate, (2) accreditation agencies require computer exercises to be integrated into core classes for engineering majors, (3) there has been an increase in accessibility of computing capabilities for undergraduate students, (4) numerical methods allow the student to solve more complex problems, and (5) the ease of obtaining an answer allows students to "experiment" with simulators.

In the past, students have largely been expected to teach themselves numerical methods. Although students could take numerical analysis courses from math departments, such courses are typically oriented toward proofs and theorems and are not always useful for engineering students. As a result, undergraduate engineering students often are ill-prepared to apply numerical methods in their upper-division courses. Professors teaching these courses either have to present problems that have analytical solutions, or they must introduce numerical methods themselves. There are some drawbacks to these approaches. Problems which have analytical solutions are usually idealized and are not always the most relevant or interesting examples. In addition, the analytical solution procedures can often be abstract mathematical departures from the subject at hand and can confuse the student instead of providing insight. Finally, when professors are forced to include numerical techniques, they are usually only able to present a "cookbook" perspective of the few numerical methods that they choose to illustrate.

A number of chemical engineering departments have decided to offer courses that represent a unified perspective of the most useful methods for the numerical solution of engineering problems. Considering the number of courses which are designed to teach the fundamentals of chemical engineering, it appears useful to first teach students to solve the equations they will learn to formulate. The goal of this text is to provide students with the knowledge and experience to apply numerical methods efficiently for the solution of engineering problems. In addition, the concepts of convergence, stability, and accuracy are emphasized throughout the text.

The text is explicitly written at the sophomore level, where students usually have difficulty understanding a general presentation of a numerical algorithm. As a result, each such algorithm is presented in explicit detail to aid the understanding of the algorithm. Approximately half of the examples used are chemical engineering examples. Although relatively advanced chemical engineering examples are sometimes used, the final equations are developed from the fundamentals for the student. For example, a student may not have had a course in kinetics, but where kinetic examples are used in the text, students are required only to solve the final equations, not to develop them. As a result, the student is exposed to a wide variety of chemical engineering subjects without having to be knowledgeable in those areas and becomes aware of the usefulness of numerical techniques for the solution of chemical engineering problems.

This text is not intended to be encyclopedic in coverage. Rather, it is designed to present those methods which either are the most useful or illustrate the relevant concepts. The organization of the text is largely based upon problem type: algebraic equations, initial-value problems, boundary-value problems, and optimization. Chapter 1 includes the introduction plus a review of matrix operations, summation notation, and numerical errors. Chapter 2 considers the solution of algebraic equations: linear equations, a single nonlinear equation, and systems of nonlinear equations. Chapter 3 contains the finite difference approximations of derivatives and methods for interpolation and integration. Chapter 4 considers initial value problems: a single ordinary differential equation (ODE), systems of ODE's, and partial differential equations. Chapter 5 covers one-dimensional and two-dimensional boundary value problems. Chapter 6 introduces the student to optimization. Chapter 7 presents linear regression, multiple linear regression, nonlinear regression, and data smoothing. Chapter 8 provides a discussion of modeling and an outline with which to approach modeling problems. In addition, three advanced modeling problems are analyzed using the recommended modeling approach.

An important feature of this book is that all the software presented is available for IBM-compatible PC's, for Apple computers, or on magnetic tape for mainframe computers. The availability of the software allows the student not only to efficiently develop a working knowledge of numerical methods, but also to take the software with him to his upper division classes and even out into the workplace. The software is all written in FORTRAN; however, some of the smaller programs are also available in BASIC. Following is a listing of the library routines used by the text and included as software:

- Linear equation solver
- Multidimensional unconstrained optimizer
- A variable step size/variable order integrator for ODE's
- A finite element program
- A least squares, variable order spline data smoother

In addition, a number of useful codes are developed in the text:

- Cubic spline interpolator
- Thomas method
- Gauss-Seidel method
- N-dimensional Newton method
- Runge-Kutta integrator
- One-dimensional boundary value problem solver
- One-dimensional optimizer
- Linear regression method
- Multiple linear regression method
- Arc length homotopy algorithm

All the software is documented with user instructions. In addition, examples of the use of each of the programs are provided. In summary, the key features of this text are that it is explicitly written for the undergraduate and that a powerful collection of software is available with the book.

Lubbock, Texas
November, 1987

James B. Riggs

Contents

Chapter 1

Introduction

1.1 Background and Objectives

Today's chemical engineer is heavily dependent upon a variety of applications of computing technology. These include scientific computing, word processing, data management, and data acquisition. These tools have greatly enhanced his capabilities and productivity. This book is concerned with the development of numerical methods skills required for scientific computing.

Scientific computing is a generic term that applies to a variety of computational applications ranging from the use of computer-aided design packages to solving for the root of a single equation to developing from scratch a computer-implemented model for a novel, complex process. Numerical methods are an integral part of each of these problems. That is, numerical methods are the means by which model equations are solved on the computer. Moreover, the understanding of numerical methods as well as the understanding of the concepts of convergence, stability, and accuracy of an approximation are necessary for the successful solution of scientific computing problems.

Before the widespread availability and use of computers, classroom instruction of chemical engineering relied almost solely upon problems that lend themselves to analytical solutions, especially problems involving differential equations. But today much more interesting, relevant problems can be studied because we are not nearly as restricted in the type of problem that can be efficiently solved. In fact, even when an analytical solution exists for a particular problem, it is usually easier to solve the problem numerically. In many cases, the numerical solution procedures actually provide a better physical illustration of the dominant factors of a problem.

Each field of chemical engineering (i.e., kinetics, thermodynamics, control, mass transfer, etc.) has its own requirements for numerical methods. **Table 1.1** lists the most common types of problems encountered for each area. Note the prominence of initial value problems, systems of nonlinear algebraic equations, boundary value problems, systems of linear equations, and optimization in the table. Methods directed at the solution of these classes of problems are the major emphasis of this book.

Table 1.1
Types of Chemical Engineering Problems Listed by Area

AREA	MOST COMMON PROBLEM TYPE
MATERIAL AND ENERGY BALANCES	Systems of linear equations; sometimes systems of nonlinear algebraic equations.
HEAT TRANSFER	Boundary value problems, initial value problems.
MASS TRANSFER	Boundary value problems, initial value problems.
KINETICS	Systems of nonlinear algebraic equations, initial value problems.
THERMODYNAMICS	System of nonlinear algebraic equations, initial value problems, integration, interpolation.
CONTROL	Initial value problems.
DESIGN	Optimization.

It is very important to be able to identify the type of problem you are trying to solve since the problem type determines the numerical method required. For example, it makes very little difference whether an initial value problem comes from a reaction kinetics system or a control system. The point is that the problem is an initial value problem and the particular characteristics of the initial value problem will determine which numerical solution procedure is appropriate, not necessarily what type of physical problem it represents.

Therefore, the objective of this book is to enable the student to do the following:

1. Identify the general type of numerical problem he has undertaken

2. Based upon the characteristics of the equations, choose the proper numerical technique

3. Effectively implement the numerical solution

We begin this study with a review of matrix operations, summation notation, and an analysis of numerical errors.

1.2 Matrix Operations

A matrix is a rectangular array of numbers, symbols, or functions. For example,

$$\begin{pmatrix} 1 & 2 & 4 \\ -1 & 3 & 5 \\ 4 & 2 & -3 \end{pmatrix}$$

$$\begin{pmatrix} x^2 & y-2 & 2 \\ 1-z & z^{\frac{1}{2}} & 2 \end{pmatrix}$$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

are all matrices. Most of the matrices that we will encounter will be square matrices; i.e., they have the same number of rows as columns like the first and third cases in the previous example.

Consider a generalized rectangular matrix $\mathbf{A}$ with m rows and n columns

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13}\cdots & a_{1n} \\ a_{21} & a_{22} & a_{23}\cdots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{m2} & a_{m3}\cdots & a_{mn} \end{pmatrix}$$

If $n = m$, the matrix is a square matrix of order n which is also called an $n \times n$ matrix. The numbers that comprise this matrix (a_{ij}) are the elements of the matrix in which i is the row number and j is the column number. A row is a horizontal set of elements and a column is a vertical set. Elements of the type a_{ii} (e.g., a_{11}, a_{22}, a_{nn}, etc.) are called diagonal elements.

A special type of matrix is an identity matrix ($\mathbf{I}$). An identity matrix is a square matrix in which all elements are zeroes except the diagonal elements, each of which has a value of one. That is, a fourth order identity matrix is

$$\mathbf{I} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

A special class of square matrices is called "banded". A banded matrix has a triangle of zeros in the upper right and lower left corners, i.e.,

$$\begin{pmatrix} 2 & 5 & 0 & 0 & 0 \\ 1 & 3 & 2 & 0 & 0 \\ 0 & 2 & 5 & 1 & 0 \\ 0 & 0 & 2 & 5 & 1 \\ 0 & 0 & 0 & 2 & 5 \end{pmatrix}$$

is a banded matrix with a band width of 3. A banded matrix with a band width of 3 is also called a tridiagonal matrix. The band width is usually the maximum number of non-zero elements in any row. More precisely, the band width is the maximum number of non-zero elements to the right of the diagonal plus the maximum number of non-zero elements to the left of the diagonal plus one. For example, an identity matrix is a banded matrix with a band width of one.

A sparse matrix contains a large number of zero elements. The locations of the zero elements are not restricted for a sparse matrix. For example, banded matrices are a subset of the general class of sparse matrices. An example of a sparse matrix is

$$\begin{pmatrix} 1 & 0 & 2 & 0 & 2 \\ 0 & 0 & 0 & 1 & 3 \\ 4 & 0 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 & 0 \\ 5 & 3 & 0 & 0 & 0 \end{pmatrix}$$

The transpose of a matrix is the matrix that results when the columns are written as rows (or likewise if the rows are written as columns). That is, the transpose of a matrix with elements a_{ij} would have elements a_{ji}; for example,

$$\mathbf{A} = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \\ 0 & 3 & 7 \end{pmatrix}$$

$$\mathbf{A}^T = \begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 3 \\ 5 & 6 & 7 \end{pmatrix}$$

A vector is a special case of a matrix: a column vector has a single column with a number of rows, e.g.,

$$\begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$$

A row vector has a single row with a number of columns, e.g., $(2\ 1\ 3)$. Unless otherwise specified, a vector is usually considered a column vector.

Two matrices are said to be equal, i.e.,

$$\mathbf{A} = \mathbf{B}$$

when corresponding elements are equal, i.e., when $a_{ij} = b_{ij}$, for all values of i and j.

Matrix addition is applicable only to matrices that have the same number of rows and the same number of columns. For example, for

$$\mathbf{A} + \mathbf{B} = \mathbf{C}$$

A, **B**, and **C** must have the same number of rows and columns. Also, the equation indicates that

$$a_{ij} + b_{ij} = c_{ij}$$

for all values of i and j. Therefore, matrix addition provides a concise means of performing a number of parallel addition operations.

Matrix multiplication involves row and column multiplication, whereas matrix addition involves only element addition. Consider the following example:

$$\mathbf{AB} = \mathbf{C}$$

where

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{pmatrix}$$

By matrix multiplication, the first element of $\mathbf{C}$, c_{11}, is given by the scalar product of the first row of $\mathbf{A}$ and the column of $\mathbf{B}$; i.e.,

$$c_{11} = (a_{11}\ a_{12}) \cdot \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix} = a_{11}\, b_{11} + a_{12}\, b_{21}$$

Likewise, c_{21} is formed by the scalar product of the second row of $\mathbf{A}$ and the first column of $\mathbf{B}$. In general, c_{ij} is formed by the product of the ith row of $\mathbf{A}$ and the jth column of $\mathbf{B}$. Therefore, the number of columns of the first matrix of the product must be equal to the number of rows of the second matrix. It can easily be shown that in general

$$\mathbf{A}\,\mathbf{B} \neq \mathbf{B}\,\mathbf{A}$$

The inverse of a matrix ($\mathbf{A}^{-1}$) is defined such that

$$\mathbf{A}\,\mathbf{A}^{-1} = \mathbf{A}^{-1}\,\mathbf{A} = \mathbf{I}$$

Matrix multiplication provides a very simple, concise means of representing a system of linear equations. Consider the following matrix equation

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}$$

where

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Performing the matrix multiplication yields the following set of linear equations:

$$\begin{aligned} a_{11}\, x_1 + a_{12}\, x_2 + a_{13}\, x_3 &= b_1 \\ a_{21}\, x_1 + a_{22}\, x_2 + a_{23}\, x_3 &= b_2 \\ a_{31}\, x_1 + a_{32}\, x_2 + a_{33}\, x_3 &= b_3 \end{aligned}$$

The $\mathbf{A}$ matrix is called the coefficient matrix. Therefore, a large system of linear equations can be compactly expressed using a coefficient matrix $\mathbf{A}$

and a constant vector **b**. In addition, this form lends itself quite readily to a generalized computer-based-solution algorithm.

Consider a function of two independent variables, $f(x,y)$. Then the gradient of f is a vector given as

$$\textbf{grad } f = \frac{\partial f}{\partial x}\mathbf{i} + \frac{\partial f}{\partial y}\mathbf{j}$$

where **i** is the unit vector in the x direction and **j** is the unit vector in the y direction.

The Jacobian matrix is generated for a system of n functions containing n unknowns; i.e.,

$$f_i(\mathbf{x}) \qquad i = 1, 2, \ldots, n$$

where

$$\mathbf{x} = (x_1, x_2, \ldots, x_n)$$

A generalized element of the Jacobian is given as

$$a_{ij} = \frac{\partial f_i(\mathbf{x})}{\partial x_j}$$

As an example, consider the following two functions:

$$f_1(\mathbf{x}) = x_1^2 e^{x_2}$$
$$f_2(\mathbf{x}) = \sin(x_2) \cdot (1 - x_1^2)$$

Then the Jacobian matrix for these functions would be

$$\begin{pmatrix} 2x_1\ e^{x_2} & x_1^2\ e^{x_2} \\ \\ -2x_1\ \sin(x_2) & (1 - x_1^2)\ \cos(x_2) \end{pmatrix}$$

Eigenvalues ($\lambda's$) of matrix **A** are defined by the nontrivial solution of the following equation:

$$(\mathbf{A} - \lambda\, \mathbf{I})\, \mathbf{x} = 0$$