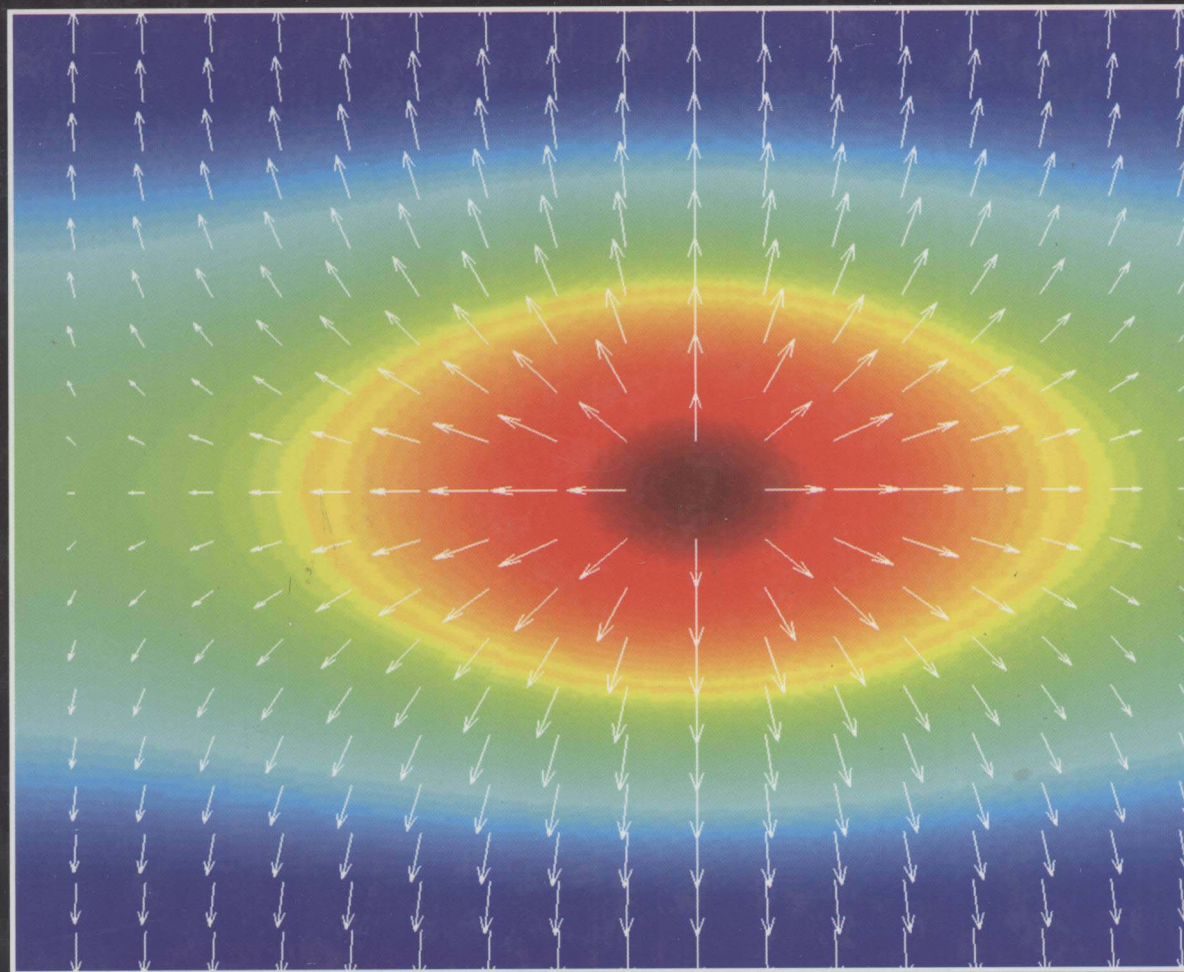


Numerical Methods for Chemical Engineering

Applications in MATLAB®



Kenneth J. Beers

CAMBRIDGE

Numerical Methods for Chemical Engineering

Applications in MATLAB[®]

KENNETH J. BEERS

Massachusetts Institute of Technology



CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS

Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press

The Edinburgh Building, Cambridge CB2 2RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org

Information on this title: www.cambridge.org/9780521859714

© K. J. Beers 2007

This publication is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without
the written permission of Cambridge University Press.

First published 2007

Printed in the United Kingdom at the University Press, Cambridge

A catalog record for this publication is available from the British Library

ISBN-13 978-0-521-85971-4 hardback

ISBN-10 0-521-85971-9 hardback

Numerical Methods for Chemical Engineering

Suitable for a first-year graduate course, this textbook unites the applications of numerical mathematics and scientific computing to the practice of chemical engineering. Written in a pedagogic style, the book describes basic linear and nonlinear algebraic systems all the way through to stochastic methods, Bayesian statistics, and parameter estimation. These subjects are developed at a nominal level of theoretical mathematics suitable for graduate engineers. The implementation of numerical methods in MATLAB[®] is integrated within each chapter and numerous examples in chemical engineering are provided, together with a library of corresponding MATLAB programs. Although the applications focus on chemical engineering, the treatment of the topics should also be of interest to non-chemical engineers and other applied scientists that work with scientific computing. This book will provide the graduate student with the essential tools required by industry and research alike.

Supplementary material includes solutions to homework problems set in the text, MATLAB programs and tutorial, lecture slides, and complicated derivations for the more advanced reader. These are available online at www.cambridge.org/9780521859714.

KENNETH J BEERS has been Assistant Professor at MIT since 2000. He has taught extensively across the engineering discipline at both the undergraduate and graduate level. This book is a result of the successful course the author devised at MIT for numerical methods applied to chemical engineering.

Preface

This text focuses on the application of quantitative analysis to the field of chemical engineering. Modern engineering practice is becoming increasingly more quantitative, as the use of scientific computing becomes ever more closely integrated into the daily activities of all engineers. It is no longer the domain of a small community of specialist practitioners. Whereas in the past, one had to hand-craft a program to solve a particular problem, carefully husbanding the limited memory and CPU cycles available, now we can very quickly solve far more complex problems using powerful, widely-available software. This has introduced the need for research engineers and scientists to become computationally literate – to know the possibilities that exist for applying computation to their problems, to understand the basic ideas behind the most important algorithms so as to make wise choices when selecting and tuning them, and to have the foundational knowledge necessary to navigate independently through the literature.

This text meets this need, and is written at the level of a first-year graduate student in chemical engineering, a consequence of its development for use at MIT for the course 10.34, “Numerical methods applied to chemical engineering.” This course was added in 2001 to the graduate core curriculum to provide all first-year Masters and Ph.D. students with an overview of quantitative methods to augment the existing core courses in transport phenomena, thermodynamics, and chemical reaction engineering. Care has been taken to develop any necessary material specific to chemical engineering, so this text will prove useful to other engineering and scientific fields as well. The reader is assumed to have taken the traditional undergraduate classes in calculus and differential equations, and to have some experience in computer programming, although not necessarily in MATLAB[®].

Even a cursory search of the holdings of most university libraries shows there to be a great number of texts with titles that are variations of “Advanced Engineering Mathematics” or “Numerical Methods.” *So why add yet another?*

I find that there are two broad classes of texts in this area. The first focuses on introducing numerical methods, applied to science and engineering, at the level of a junior or senior undergraduate elective course. The scope is necessarily limited to rather simple techniques and applications. The second class is targeted to research-level workers, either higher graduate-level applied mathematicians or computationally-focused researchers in science and engineering. These may be either advanced treatments of numerical methods for mathematicians, or detailed discussions of scientific computing as applied to a specific subject such as fluid mechanics.

Neither of these classes of text is appropriate for teaching the fundamentals of scientific computing to beginning chemical engineering graduate students. Examples should be typical of those encountered in graduate-level chemical engineering research, and while the students should gain an understanding of the basis of each method and an appreciation of its limitations, they do not need exhaustive theory-proof treatments of convergence, error analysis, etc. It is a challenge for beginning students to identify how their own problems may be mapped into ones amenable to quantitative analysis; therefore, any appropriate text should have an extensive library of worked examples, with code available to serve later as templates. Finally, the text should address the important topics of model development and parameter estimation. This book has been developed with these needs in mind.

This text first presents a fundamental discussion of linear algebra, to provide the necessary foundation to read the applied mathematical literature and progress further on one's own. Next, a broad array of simulation techniques is presented to solve problems involving systems of nonlinear algebraic equations, initial value problems of ordinary differential and differential-algebraic (DAE) systems, optimizations, and boundary value problems of ordinary and partial differential equations. A treatment of matrix eigenvalue analysis is included, as it is fundamental to analyzing these simulation techniques.

Next follows a detailed discussion of probability theory, stochastic simulation, statistics, and parameter estimation. As engineering becomes more focused upon the molecular level, stochastic simulation techniques gain in importance. Particular attention is paid to Brownian dynamics, stochastic calculus, and Monte Carlo simulation. Statistics and parameter estimation are addressed from a Bayesian viewpoint, in which Monte Carlo simulation proves a powerful and general tool for making inferences and testing hypotheses from experimental data.

In each of these areas, topically relevant examples are given, along with **MATLAB** (www.mathworks.com) programs that serve the students as templates when later writing their own code. An accompanying website includes a **MATLAB** tutorial, code listings of all examples, and a supplemental material section containing further detailed proofs and optional topics. Of course, while significant effort has gone into testing and validating these programs, no guarantee is provided and the reader should use them with caution.

The problems are graded by difficulty and length in each chapter. Those of grade A are simple and can be done easily by hand or with minimal programming. Those of grade B require more programming but are still rather straightforward extensions or implementations of the ideas discussed in the text. Those of grade C either involve significant thinking beyond the content presented in the text or programming effort at a level beyond that typical of the examples and grade B problems.

The subjects covered are broad in scope, leading to the considerable (though hopefully not excessive) length of this text. The focus is upon providing a fundamental understanding of the underlying numerical algorithms without necessarily exhaustively treating all of their details, variations, and complexities of use. Mastery of the material in this text should enable first-year graduate students to perform original work in applying scientific computation to their research, and to read the literature to progress independently to the use of more sophisticated techniques.

Writing a book is a lengthy task, and one for which I have enjoyed much help and support. Professor William Green of MIT, with whom I taught this course for one semester, generously shared his opinions of an early draft. The teaching assistants who have worked on the course have also been a great source of feedback and help in problem-development, as have, of course, the students who have wrestled with intermediate drafts and my evolving approach to teaching the subject. My Ph.D. students Jungmee Kang, Kirill Titievskiy, Erik Allen, and Brian Stephenson have shown amazing forbearance and patience as the text became an additional, and sometimes demanding, member of the group. Above all, I must thank my family, and especially my supportive wife Jen, who have been tracking my progress and eagerly awaiting the completion of the book.

Contents

<i>Preface</i>	<i>page ix</i>
1 Linear algebra	1
Linear systems of algebraic equations	1
Review of scalar, vector, and matrix operations	3
Elimination methods for solving linear systems	10
Existence and uniqueness of solutions	23
The determinant	32
Matrix inversion	36
Matrix factorization	38
Matrix norm and rank	44
Submatrices and matrix partitions	44
Example. Modeling a separation system	45
Sparse and banded matrices	46
MATLAB summary	56
Problems	57
2 Nonlinear algebraic systems	61
Existence and uniqueness of solutions to a nonlinear algebraic equation	61
Iterative methods and the use of Taylor series	62
Newton's method for a single equation	63
The secant method	69
Bracketing and bisection methods	70
Finding complex solutions	70
Systems of multiple nonlinear algebraic equations	71
Newton's method for multiple nonlinear equations	72
Estimating the Jacobian and quasi-Newton methods	77
Robust reduced-step Newton method	79
The trust-region Newton method	81
Solving nonlinear algebraic systems in MATLAB	83
Example. 1-D laminar flow of a shear-thinning polymer melt	85
Homotopy	88
Example. Steady-state modeling of a condensation polymerization reactor	89

	Bifurcation analysis	94
	MATLAB summary	98
	Problems	99
3	Matrix eigenvalue analysis	104
	Orthogonal matrices	104
	A specific example of an orthogonal matrix	105
	Eigenvalues and eigenvectors defined	106
	Eigenvalues/eigenvectors of a 2×2 real matrix	107
	Multiplicity and formulas for the trace and determinant	109
	Eigenvalues and the existence/uniqueness properties of linear systems	110
	Estimating eigenvalues; Gershgorin's theorem	111
	Applying Gershgorin's theorem to study the convergence of iterative linear solvers	114
	Eigenvector matrix decomposition and basis sets	117
	Numerical calculation of eigenvalues and eigenvectors in MATLAB	123
	Computing extremal eigenvalues	126
	The QR method for computing all eigenvalues	129
	Normal mode analysis	134
	Relaxing the assumption of equal masses	136
	Eigenvalue problems in quantum mechanics	137
	Single value decomposition SVD	141
	Computing the roots of a polynomial	148
	MATLAB summary	149
	Problems	149
4	Initial value problems	154
	Initial value problems of ordinary differential equations (ODE-IVPs)	155
	Polynomial interpolation	156
	Newton–Cotes integration	162
	Gaussian quadrature	163
	Multidimensional integrals	167
	Linear ODE systems and dynamic stability	169
	Overview of ODE-IVP solvers in MATLAB	176
	Accuracy and stability of single-step methods	185
	Stiff stability of BDF methods	192
	Symplectic methods for classical mechanics	194
	Differential-algebraic equation (DAE) systems	195
	Parametric continuation	203
	MATLAB summary	207
	Problems	208

5	Numerical optimization	212
	Local methods for unconstrained optimization problems	212
	The simplex method	213
	Gradient methods	213
	Newton line search methods	223
	Trust-region Newton method	225
	Newton methods for large problems	227
	Unconstrained minimizer fminunc in MATLAB	228
	Example. Fitting a kinetic rate law to time-dependent data	230
	Lagrangian methods for constrained optimization	231
	Constrained minimizer fmincon in MATLAB	242
	Optimal control	246
	MATLAB summary	252
	Problems	252
6	Boundary value problems	258
	BVPs from conservation principles	258
	Real-space vs. function-space BVP methods	260
	The finite difference method applied to a 2-D BVP	260
	Extending the finite difference method	264
	Chemical reaction and diffusion in a spherical catalyst pellet	265
	Finite differences for a convection/diffusion equation	270
	Modeling a tubular chemical reactor with dispersion; treating multiple fields	279
	Numerical issues for discretized PDEs with more than two spatial dimensions	282
	The MATLAB 1-D parabolic and elliptic solver pdepe	294
	Finite differences in complex geometries	294
	The finite volume method	297
	The finite element method (FEM)	299
	FEM in MATLAB	309
	Further study in the numerical solution of BVPs	311
	MATLAB summary	311
	Problems	312
7	Probability theory and stochastic simulation	317
	The theory of probability	317
	Important probability distributions	325
	Random vectors and multivariate distributions	336
	Brownian dynamics and stochastic differential equations (SDEs)	338
	Markov chains and processes; Monte Carlo methods	353
	Genetic programming	362

	MATLAB summary	364
	Problems	365
8	Bayesian statistics and parameter estimation	372
	General problem formulation	372
	Example. Fitting kinetic parameters of a chemical reaction	373
	Single-response linear regression	377
	Linear least-squares regression	378
	The Bayesian view of statistical inference	381
	The least-squares method reconsidered	388
	Selecting a prior for single-response data	389
	Confidence intervals from the approximate posterior density	395
	MCMC techniques in Bayesian analysis	403
	MCMC computation of posterior predictions	404
	Applying eigenvalue analysis to experimental design	412
	Bayesian multi response regression	414
	Analysis of composite data sets	421
	Bayesian testing and model criticism	426
	Further reading	431
	MATLAB summary	431
	Problems	432
9	Fourier analysis	436
	Fourier series and transforms in one dimension	436
	1-D Fourier transforms in MATLAB	445
	Convolution and correlation	447
	Fourier transforms in multiple dimensions	450
	Scattering theory	452
	MATLAB summary	459
	Problems	459
	References	461
	<i>Index</i>	464

1 Linear algebra

This chapter discusses the solution of sets of linear algebraic equations and defines basic vector/matrix operations. The focus is upon elimination methods such as Gaussian elimination, and the related LU and Cholesky factorizations. Following a discussion of these methods, the existence and uniqueness of solutions are considered. Example applications include the modeling of a separation system and the solution of a fluid mechanics boundary value problem. The latter example introduces the need for sparse-matrix methods and the computational advantages of banded matrices. Because linear algebraic systems have, under well-defined conditions, a unique solution, they serve as fundamental building blocks in more-complex algorithms. Thus, linear systems are treated here at a high level of detail, as they will be used often throughout the remainder of the text.

Linear systems of algebraic equations

We wish to solve a system of N simultaneous linear algebraic equations for the N unknowns x_1, x_2, \dots, x_N , that are expressed in the general form

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N &= b_2 \\ &\vdots \\ a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N &= b_N \end{aligned} \tag{1.1}$$

a_{ij} is the constant coefficient (assumed real) that multiplies the unknown x_j in equation i . b_i is the constant “right-hand-side” coefficient for equation i , also assumed real. As a particular example, consider the system

$$\begin{aligned} x_1 + x_2 + x_3 &= 4 \\ 2x_1 + x_2 + 3x_3 &= 7 \\ 3x_1 + x_2 + 6x_3 &= 2 \end{aligned} \tag{1.2}$$

for which

$$\begin{array}{cccc} a_{11} = 1 & a_{12} = 1 & a_{13} = 1 & b_1 = 4 \\ a_{21} = 2 & a_{22} = 1 & a_{23} = 3 & b_2 = 7 \\ a_{31} = 3 & a_{32} = 1 & a_{33} = 6 & b_3 = 2 \end{array} \tag{1.3}$$

It is common to write linear systems in *matrix/vector form* as

$$A\mathbf{x} = \mathbf{b} \quad (1.4)$$

where

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2N} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} \quad (1.5)$$

Row i of A contains the values $a_{i1}, a_{i2}, \dots, a_{iN}$ that are the coefficients multiplying each unknown x_1, x_2, \dots, x_N in equation i . Column j contains the coefficients $a_{1j}, a_{2j}, \dots, a_{Nj}$ that multiply x_j in each equation $i = 1, 2, \dots, N$. Thus, we have the following associations,

		coefficients multiplying
rows \Leftrightarrow equations	columns \Leftrightarrow	a specific unknown
		in each equation

We often write $A\mathbf{x} = \mathbf{b}$ explicitly as

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} \quad (1.6)$$

For the example system (1.2),

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & 3 \\ 3 & 1 & 6 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 4 \\ 7 \\ 2 \end{bmatrix} \quad (1.7)$$

In MATLAB we solve $A\mathbf{x} = \mathbf{b}$ with the single command, $\mathbf{x} = \mathbf{A} \backslash \mathbf{b}$. For the example (1.2), we compute the solution with the code

```
A = [1 1 1; 2 1 3; 3 1 6];
b = [4; 7; 2];
x = A\b,
x =
    19.0000
   -7.0000
   -8.0000
```

Thus, we are tempted to assume that, as a practical matter, we need to know little about how to solve a linear system, as someone else has figured it out and provided us with this handy linear solver. Actually, we shall need to understand the fundamental properties of linear systems in depth to be able to master methods for solving more complex problems, such as sets of nonlinear algebraic equations, ordinary and partial

differential equations, etc. Also, as we shall see, this solver fails for certain common classes of very large systems of equations, and we need to know enough about linear algebra to diagnose such situations and to propose other methods that do work in such instances. This chapter therefore contains not only an explanation of how the MATLAB solver is implemented, but also a detailed, fundamental discussion of the properties of linear systems.

Our discussion is intended only to provide a foundation in linear algebra for the practice of numerical computing, and is continued in Chapter 3 with a discussion of matrix eigenvalue analysis. For a broader, more detailed, study of linear algebra, consult Strang (2003) or Golub & van Loan (1996).

Review of scalar, vector, and matrix operations

As we use vector notation in our discussion of linear systems, a basic review of the concepts of vectors and matrices is necessary.

Scalars, real and complex

Most often in basic mathematics, we work with *scalars*, i.e., single-valued numbers. These may be real, such as 3, 1.4, $5/7$, $3.14159 \dots$, or they may be complex, $1 + 2i$, $1/2 i$, where $i = \sqrt{-1}$. The *set of all real scalars* is denoted \Re . The *set of all complex scalars* we call C . For a complex number $z \in C$, we write $z = a + ib$, where $a, b \in \Re$ and

$$\begin{aligned} a &= \text{Re}\{z\} = \text{real part of } z \\ b &= \text{Im}\{z\} = \text{imaginary part of } z \end{aligned} \quad (1.8)$$

The *complex conjugate*, $\bar{z} = z^*$, of $z = a + ib$ is

$$\bar{z} = z^* = a - ib \quad (1.9)$$

Note that the product $\bar{z}z$ is always real and nonnegative,

$$\bar{z}z = (a - ib)(a + ib) = a^2 - iab + iab - i^2 b^2 = a^2 + b^2 \geq 0 \quad (1.10)$$

so that we may define the real-valued, nonnegative *modulus* of z , $|z|$, as

$$|z| = \sqrt{\bar{z}z} = \sqrt{a^2 + b^2} \geq 0 \quad (1.11)$$

Often, we write complex numbers in polar notation,

$$z = a + ib = |z|(\cos \theta + i \sin \theta) \quad \theta = \tan^{-1}(b/a) \quad (1.12)$$

Using the important *Euler formula*, a proof of which is found in the supplemental material found at the website that accompanies this book,

$$e^{i\theta} = \cos \theta + i \sin \theta \quad (1.13)$$

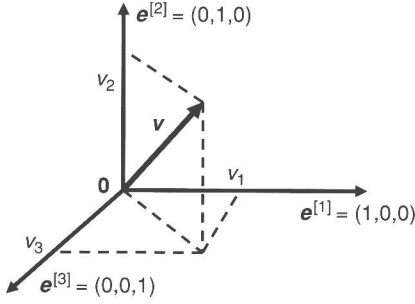


Figure 1.1 Physical interpretation of a 3-D vector.

we can write z as

$$z = |z|e^{i\theta} \quad (1.14)$$

Vector notation and operations

We write a three-dimensional (3-D) vector \mathbf{v} (Figure 1.1) as

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \quad (1.15)$$

\mathbf{v} is *real* if $v_1, v_2, v_3 \in \Re$; we then say $\mathbf{v} \in \Re^3$. We can easily visualize this vector in 3-D space, defining the three coordinate basis vectors in the 1(x), 2(y), and 3(z) directions as

$$\mathbf{e}^{[1]} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{e}^{[2]} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \mathbf{e}^{[3]} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (1.16)$$

to write $\mathbf{v} \in \Re^3$ as

$$\mathbf{v} = v_1 \mathbf{e}^{[1]} + v_2 \mathbf{e}^{[2]} + v_3 \mathbf{e}^{[3]} \quad (1.17)$$

We extend this notation to define \Re^N , the *set of N -dimensional real vectors*,

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{bmatrix} \quad (1.18)$$

where $v_j \in \Re$ for $j = 1, 2, \dots, N$. By writing \mathbf{v} in this manner, we define a *column vector*; however, \mathbf{v} can also be written as a *row vector*,

$$\mathbf{v} = [v_1 \quad v_2 \dots \quad v_N] \quad (1.19)$$

The difference between column and row vectors only becomes significant when we start combining them in equations with matrices.

We write $\mathbf{v} \in \Re^N$ as an expansion in coordinate basis vectors as

$$\mathbf{v} = v_1 \mathbf{e}^{[1]} + v_2 \mathbf{e}^{[2]} + \cdots + v_N \mathbf{e}^{[N]} \quad (1.20)$$

where the components of $\mathbf{e}^{[j]}$ are *Kronecker deltas* δ_{jk} ,

$$\mathbf{e}^{[j]} = \begin{bmatrix} e_1^{[j]} \\ e_2^{[j]} \\ \vdots \\ e_N^{[j]} \end{bmatrix} = \begin{bmatrix} \delta_{j1} \\ \delta_{j2} \\ \vdots \\ \delta_{jN} \end{bmatrix} \quad \delta_{jk} = \begin{cases} 1, & \text{if } j = k \\ 0, & \text{if } j \neq k \end{cases} \quad (1.21)$$

Addition of two real vectors $\mathbf{v} \in \Re^N$, $\mathbf{w} \in \Re^N$ is straightforward,

$$\mathbf{v} + \mathbf{w} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{bmatrix} + \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} v_1 + w_1 \\ v_2 + w_2 \\ \vdots \\ v_N + w_N \end{bmatrix} \quad (1.22)$$

as is multiplication of a vector $\mathbf{v} \in \Re^N$ by a real scalar $c \in \Re$,

$$c\mathbf{v} = c \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{bmatrix} = \begin{bmatrix} cv_1 \\ cv_2 \\ \vdots \\ cv_N \end{bmatrix} \quad (1.23)$$

For all $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Re^N$ and all $c_1, c_2 \in \Re$,

$$\begin{aligned} \mathbf{u} + (\mathbf{v} + \mathbf{w}) &= (\mathbf{u} + \mathbf{v}) + \mathbf{w} & c(\mathbf{v} + \mathbf{u}) &= c\mathbf{v} + c\mathbf{u} \\ \mathbf{u} + \mathbf{v} &= \mathbf{v} + \mathbf{u} & (c_1 + c_2)\mathbf{v} &= c_1\mathbf{v} + c_2\mathbf{v} \\ \mathbf{v} + \mathbf{0} &= \mathbf{v} & (c_1 c_2)\mathbf{v} &= c_1(c_2\mathbf{v}) \\ \mathbf{v} + (-\mathbf{v}) &= \mathbf{0} & 1\mathbf{v} &= \mathbf{v} \end{aligned} \quad (1.24)$$

where the *null vector* $\mathbf{0} \in \Re^N$ is

$$\mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (1.25)$$

We further add to the list of operations associated with the vectors $\mathbf{v}, \mathbf{w} \in \Re^N$ the *dot (inner, scalar) product*,

$$\mathbf{v} \cdot \mathbf{w} = v_1 w_1 + v_2 w_2 + \cdots + v_N w_N = \sum_{k=1}^N v_k w_k \quad (1.26)$$

For example, for the two vectors

$$\mathbf{v} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \quad (1.27)$$

$$\begin{aligned} \mathbf{v} \cdot \mathbf{w} &= v_1 w_1 + v_2 w_2 + v_3 w_3 = (1)(4) + (2)(5) + (3)(6) \\ &= 4 + 10 + 18 = 32 \end{aligned} \quad (1.28)$$

For 3-D vectors, the dot product is proportional to the product of the lengths and the cosine of the angle between the two vectors,

$$\mathbf{v} \cdot \mathbf{w} = |\mathbf{v}| |\mathbf{w}| \cos \theta \quad (1.29)$$

where the *length* of \mathbf{v} is

$$|\mathbf{v}| = \sqrt{\mathbf{v} \cdot \mathbf{v}} \geq 0 \quad (1.30)$$

Therefore, when two vectors are parallel, the magnitude of their dot product is maximal and equals the product of their lengths, and when two vectors are perpendicular, their dot product is zero. These ideas carry completely into N - dimensions. The *length* of a vector $\mathbf{v} \in \Re^N$ is

$$|\mathbf{v}| = \sqrt{\mathbf{v} \cdot \mathbf{v}} = \sqrt{\sum_{k=1}^N v_k^2} \geq 0 \quad (1.31)$$

If $\mathbf{v} \cdot \mathbf{w} = 0$, \mathbf{v} and \mathbf{w} are said to be *orthogonal*, the extension of the adjective “perpendicular” from \Re^3 to \Re^N . If $\mathbf{v} \cdot \mathbf{w} = 0$ and $|\mathbf{v}| = |\mathbf{w}| = 1$, i.e., both vectors are *normalized* to unit length, \mathbf{v} and \mathbf{w} are said to be *orthonormal*.

The formula for the length $|\mathbf{v}|$ of a vector $\mathbf{v} \in \Re^N$ satisfies the more general properties of a *norm* $\|\mathbf{v}\|$ of $\mathbf{v} \in \Re^N$. A norm $\|\mathbf{v}\|$ is a rule that assigns a real scalar, $\|\mathbf{v}\| \in \Re$, to each vector $\mathbf{v} \in \Re^N$ such that for every $\mathbf{v}, \mathbf{w} \in \Re^N$, and for every $c \in \Re$, we have

$$\begin{aligned} \|\mathbf{v}\| &\geq 0 & \|\mathbf{0}\| &= 0 \\ \|\mathbf{v}\| &= 0 & \text{if and only if (iff)} & \mathbf{v} = \mathbf{0} \\ \|c\mathbf{v}\| &= |c| \|\mathbf{v}\| \\ \|\mathbf{v} + \mathbf{w}\| &\leq \|\mathbf{v}\| + \|\mathbf{w}\| \end{aligned} \quad (1.32)$$

Each norm also provides an accompanying *metric*, a measure of how different two vectors are

$$d(\mathbf{v}, \mathbf{w}) = \|\mathbf{v} - \mathbf{w}\| \quad (1.33)$$

In addition to the length, many other possible definitions of norm exist. The *p-norm*, $\|\mathbf{v}\|_p$, of $\mathbf{v} \in \Re^N$ is

$$\|\mathbf{v}\|_p = \left[\sum_{k=1}^N |v_k|^p \right]^{1/p} \quad (1.34)$$