Contents

Preface

This book is designed to be the primary text for an undergraduate course on numerical analysis. It is appropriate for students who have studied calculus and linear algebra and who have some exposure to differential equations and mathematical proofs. No prior experience with computer programming is expected.

To the student

Numerical analysis is a subject that combines mathematical theory and computer code to solve problems in mathematics, science, and engineering. The focus is on problems that are impossible to solve by hand. For example, even simple and important integrals like $\int e^{-x^2}\,\mathrm{d}x$, which comes from the "bell curve" of statistics, may not be expressible as simple formulas. Although exact values may be impossible to obtain, we can often compute accurate approximations to extremely high precision using mathematical ingenuity and a bit of computer code.

If you have never studied numerical analysis, then this book is for you. Prior experience with computer programming or numerical computation is not expected; the MATLAB® computing environment is introduced early, and you will gradually write more sophisticated computer code as your reading progresses. You should be proficient with multivariable calculus and matrices, and some exposure to differential equations is beneficial. Prior experience with mathematical proofs is helpful, but most error bounds follow from a small handful of fundamental theorems.

By working through this text from beginning to end, you will achieve a number of objectives. You will understand why numerical computation is necessary. You will implement and apply algorithms for interpolation, integration, linear systems, zero finding, and differential equations. You will measure error, recognize numerical convergence, and infer rate of convergence from experimental data. You will prove error bounds and explain how the performance of a numerical method is determined by characteristics of the supplied data or the desired solution.

The first part of the book begins to teach you to think like a numerical analyst. You will perform quick computations on the computer, write your first computer code, and evaluate a few methods for accuracy and efficiency. The rest of the book is organized around a few areas of mathematics that benefit from numerical analysis: interpolation, integration, linear systems, zero finding, and differential equations. Within each of these areas, you will develop methods for solving some of the most important problems and evaluate the effectiveness of those methods.

As you begin to compute the impossible, my greatest recommendation is to be active. While reading the text, run the demonstrations on your own computer, conduct additional experiments of your own design, and check the proofs by hand. While working the exercises, if you are not sure whether an approach will succeed, try it on the computer and see. Be guided first and foremost by what works. Theory will follow.

To the instructor

This textbook is based on an undergraduate numerical analysis course that I have taught since 2007. In the years since the first offering, I have made significant changes to better reflect my understanding of how numerical analysts do their work and to include profound developments in the practice of numerical computation. As a result, my students are now solving problems that I considered prohibitively difficult a few years ago, and they have a better understanding of how their methods work. This textbook is intended to bring these developments to a wider audience.

A few distinctive features of the book are described below.

Theory and experimentation. This book is full of numerical experiments in addition to proved theorems. During my own education, I developed an experimental approach to mathematics out of necessity: when encountering an unfamiliar subject, numerical experiments were the quickest and most reliable way to gain insight. From other researchers I learned that experiments can also be quite convincing, even when proofs are available. A kind of graphic that has become very common in research articles is the rate-of-convergence graph, which plots error versus computational resources. For numerical methods that work well, the error steadily falls toward zero as more work is performed, and the rate of convergence is easily inferred. Many examples in this text involve theoretical prediction followed by numerical experimentation, with the rate-of-convergence graph providing a direct comparison.

Local and global methods. A significant motivation for this book is Chebyshev technology, which has seen an explosion of activity in recent years thanks in large part to the Chebfun project initiated by Zachary Battles and Lloyd N. Trefethen [3]. While many older methods work locally—think of Newton's method targeting a single zero or Newton–Cotes quadrature chopping a function into small pieces—a Chebyshev method models a function globally by a single polynomial. This can deliver extremely high accuracy at low cost. I was introduced to this approach during an eye-opening presentation by Trefethen in 2008, and I could never teach numerical analysis the old way again. This book introduces global Chebyshev methods alongside more traditional local methods, in many cases deriving both types of methods from a common core.

Analysis and linear algebra. In this text, I deliberately pursue a marriage between analysis (calculus, differential equations, real and complex analysis, etc.) and linear algebra. Although the table of contents suggests a heavier emphasis on problems from analysis, linear algebra is equally important because it is the foundation of so many solution methods. A common tactic is to project a continuous function onto a finite-dimensional vector space and simultaneously reduce the analysis problem to a linear algebra problem. In particular, definite integration reduces to an inner product; interpolation, differentiation, and antidifferentiation reduce to matrix-vector products; and differential equations reduce to systems of linear (algebraic) equations. The unifying question is, Can we design a matrix to solve the problem for us? Through this pursuit, the student goes beyond the mechanics of matrix computations to recognize and exploit linearity.

In my experience, students enjoy studying numerical analysis. Numerical methods allow them to solve problems that would otherwise be impossible, and the feedback from computer code is immediate and compelling. At the same time, the mathematical theory underlying numerical methods is deep and broad. By completing this course, students will experience the practical implications of theory that may otherwise seem abstract.

Software

The numerical methods in this book are implemented in the MATLAB programming language and are stored in computer files called *M-files*. These files and directions for their installation are available on the web at www.siam.org/books/ot161. Briefly, you will download a package, expand it to produce a folder of M-files, and add the folder to the *search path* in the MATLAB environment. As long as the example code in Chapter 2 executes without error messages, you will know that the software is installed correctly.

Acknowledgments

My collaborator in all things is my wife Megan, who has taught me much about the English language and who has provided essential support and encouragement throughout the writing of this book.

I recognize my MATH 442 students for working through drafts, some rougher than others, of this book.

I thank everyone at SIAM for their improvements to this book and for being so great to work with.

Two conferences, *Chebfun and Beyond* and *New Directions in Numerical Computation*, both located at Oxford University, gave me the opportunity to learn firsthand from many innovators in the area of Chebyshev technology.

The writing of this book was supported by sabbatical leave from Randolph-Macon College and by grants from the Rashkind Family Foundation and the Walter Williams Craigie Teaching Endowment.

List of Notation

$\hat{\cdot}$, approximation, 6
$\cdot_2$, binary notation, 52
$\cdot^T$, transpose, 17
$\cdot|_{\mathbf{x}}$, sample, 70
$\|\cdot\|_\infty$
  function infinity norm, 44
  matrix infinity norm, 238
  vector infinity norm, 167
$\approx$, approximately equal, 7
$\oplus$, floating-point addition, 57
$\ominus$, floating-point subtraction, 57
$\otimes$, floating-point multiplication, 57
$\oslash$, floating-point division, 57
$\vee$, maximum, 89
$\mathbf{0}$, vector or matrix of zeros, 163
$\mathbf{1}$, vector of ones, 161

$\alpha(t)$, coefficient function in a linear DE, 251
β, floating-point exponent, 54
$\beta(t)$, coefficient function in a linear DE, 251
γ, contour, 409
$\gamma(t)$, coefficient function in a linear DE, 253
$\Gamma(x)$, gamma function, 329
δ
  bound on sampling error, 75
  representation error, 56
δ_{jk}, Kronecker delta, 136
Δp, Newton step in collocation, 364
Δx, Newton step, 348
ε, error tolerance, 92
$\boldsymbol{\varepsilon}$, discrete collocation residual, 274
ζ
  zero of a function, 73
  variable of integration in a contour integral, 410
η, point of unknown location, 71
θ, angle in elliptical coordinates, 111
κ_{abs}, absolute condition number, 239
κ_{rel}, relative condition number, 239
$\boldsymbol{\lambda}$, definite integration weights, 159
Λ, Lebesgue constant, 74
$\lambda(x)$, Lebesgue function, 74
ξ, dummy variable of integration, 148
ρ, ellipse parameter, 121
ω, sign, 75

a, left endpoint, 71
$\mathbf{A}$
  companion matrix, 335
  matrix in a matrix-vector equation, 213
  multiplication operator, 262
Ai, Airy function, 329

b
  right endpoint, 71
  right-hand side of an equation, 36
$\mathbf{b}$, right-hand side of a matrix-vector equation, 213
$\mathbf{B}$
  companion matrix, 335
  multiplication operator, 262

C
  arbitrary constant in barycentric weights, 68
  arbitrary constant of integration, 148
  coefficient in an error bound, 89
$\mathbf{C}$, multiplication operator, 266
c_i
  coefficient in initial condition, 261
  polynomial coefficient, 24
$C^k[a, b]$, continuously differentiable functions, 71

d, initial value in an IVP, 261
$\mathbf{D}_{\mathbf{x}',\mathbf{x}}$, differentiation matrix, 148
diag($\cdot$), diagonal matrix, 266

e, shifted exponent in IEEE 754 representation, 55
$\mathbf{e}_1$, first column of an identity matrix, 282
$E_\rho(a, b)$, Bernstein ellipse, 121
$\mathbf{E}_{lm}$
  resampling matrix for Chebyshev grids, 140
  resampling matrix for uniform grids, 138
$\mathbf{E}_{\mathbf{x}',\mathbf{x}}$, evaluation matrix, 135

f, significand, 54

$\mathbb{F}$, floating-point-representable numbers, 55
$f(t, u)$, right-hand side of a nonlinear DE, 363
$f(t, u, v)$, right-hand side of a nonlinear DE, 363
$f(x)$, function, 41
$\mathrm{fl}(x)$, floating-point representative, 56

$g(t)$, right-hand side of a linear DE, 251
$g(x)$, integrand, 126

h, distance between nodes, 79

i, index, 41
i, imaginary unit, 401
$\mathbf{I}$, identity matrix, 163

j, index, 79
J_ν, Bessel function, 255
$\mathbf{J}_m$
  net change operator for a Chebyshev grid, 197
  net change operator for a uniform grid, 179
$\mathbf{J}_{mn}$, net change operator for a piecewise-uniform grid, 181
$\mathbf{J}_\mathbf{x}$, net change operator, 161

k
  index, 65
  multiplicity of a zero or node, 397
$\mathbf{K}_m$
  integration matrix for a Chebyshev grid, 195
  integration matrix for a uniform grid, 171
$\hat{\mathbf{K}}_m$
  truncated integration matrix for a Chebyshev grid, 197
  truncated integration matrix for a uniform grid, 179
$\hat{\mathbf{K}}_{mn}$ truncated integration matrix for a piecewise-uniform grid, 181
$\mathbf{K}_{\mathbf{x},\mathbf{x}'}$ integration matrix, 158
$\hat{\mathbf{K}}_{\mathbf{x},\mathbf{x}'}$, truncated integration matrix, 161

l
  degree of a grid, 135
  width of an interval, 79
$\mathbf{L}$
  collocation matrix, 274
  unit lower-triangular matrix, 227
$l(x)$, monic polynomial associated with a grid, 110
$l_i(x)$, Lagrange basis polynomial, 65
$\mathbf{L}_j$, elimination matrix, 218

m
  degree of a grid, 65
  degree of a polynomial, 41
M, function bound, 122
$\mathbf{M}$, invertible matrix, 215

n
  index in a sequence, 29
  number of subintervals, 79
N, approximate sampling cost, 89

$\mathbf{p}$, polynomial sample, 148
$\mathbf{P}$, permutation matrix, 231
$p(x)$, polynomial or piecewise-polynomial function, 24
$\hat{p}(x)$, perturbed polynomial, 74
$\tilde{p}(x)$, polynomial interpolating the true solution to a DE, 274
$P_{mn}[a, b]$, piecewise-polynomial functions, 82

$\mathbf{q}$, polynomial sample, 148
$q(x)$, polynomial or piecewise-polynomial integrand, 148
$\tilde{q}(x)$, polynomial interpolating the true solution to a DE, 274

r
  multiplicity of a zero, 342
  number of derivatives, 126
$\mathbf{r}$, polynomial sample, 266
$R(t)$, residual for a DE, 261

s, sign bit, 54
s_j, collocation node, 261
$\mathbf{S}_j$, transposition matrix, 222

t
  domain variable in $[-1, 1]$, 122
  domain variable in $[0, 1]$, 172
  independent variable in a DE, 251
$\mathbf{t}$, grid, 138
$T_m(x)$, Chebyshev polynomial of the first kind, 113

$\mathbf{u}$, sample of a solution to a DE, 273
$\mathbf{U}$, upper-triangular matrix, 213
$u(t)$, solution to a DE, 251
u_a, initial value in an IVP, 252
$U_m(x)$, Chebyshev polynomial of the second kind, 111

$\mathbf{v}$, sample of the derivative of a solution to a DE, 273
v_a, initial value for a derivative in an IVP, 253

$\mathbf{w}$
  sample of the second derivative of a solution to a DE, 277
  vector of barycentric weights, 336
w_i, barycentric weight, 68

x
  desired value in a numerical computation, 6
  domain variable, 20

$\mathbf{x}$
  grid, 70
  solution to a matrix-vector equation, 147
$\mathbf{X}$, diagonal matrix of nodes, 336
$\hat{x}$, approximation to x, 6
$\hat{\mathbf{x}}$, approximation to $\mathbf{x}$, 237
x_i, interpolation node, 65
x_{ij}, node in a piecewise grid, 79
$x^{(k)}$, Newton iterate, 347
x_n, term in a sequence, 31

y_i, sample point, 65
$\hat{y}_i$, perturbed sample point, 75

z
  approximate zero, 341
  complex variable, 401

Index