

Contents

Preface	xiii
Author	xvii
I Solving Linear and Nonlinear Systems of Equations	1
1 Solving Linear Systems Using Direct Methods	3
1.1 Testing the Existence of the Solution	3
1.2 Methods for Solving Linear Systems	5
1.2.1 Special Linear Systems	5
1.2.2 Gauss and Gauss-Jordan Elimination	9
1.2.3 Solving the System with the rref Function	10
1.3 Matrix Factorization Techniques	12
1.3.1 The <i>LU</i> Factorization	12
1.3.2 The <i>QR</i> Factorization	16
1.3.3 The Singular Value Decomposition (SVD)	19
2 Solving Linear Systems with Iterative and Least Squares Methods	23
2.1 Mathematical Backgrounds	23
2.1.1 Convergent Sequences and Cauchi's Convergence	23
2.1.2 Vector Norm	25
2.1.3 Convergent Sequences of Vectors	25
2.2 The Iterative Methods	25
2.2.1 The General Idea	26
2.2.2 The Jacobi Method	28
2.2.3 The Jacobi Method in the Matrix Form	31
2.2.3.1 The Gauss-Seidel Iterative Method	33
2.2.4 The Gauss-Seidel Method in the Vector Form	35
2.2.5 The Relaxation Methods	36
2.3 The Least Squares Solutions	39
2.3.1 Some Applications of Least Squares Solutions	43

3	Ill-Conditioning and Regularization Techniques in Solutions of Linear Systems	57
3.1	Ill-Conditioning in Solutions of Linear Systems	57
3.1.1	More Examples of Ill-Posed System	63
3.1.2	Condition Numbers and Ill-Conditioned Matrices	67
3.1.3	Linking the Condition Numbers to Matrix Related Eigenvalues	71
3.1.4	Further Analysis on Ill-Posed Systems	75
3.2	Regularization of Solutions in Linear Systems	78
3.2.1	The Truncated SVD (TSVD) Method	78
3.2.2	Tikhonov Regularization Method	82
3.2.3	The L-curve Method	87
3.2.4	The Discrepancy Principle	87
4	Solving a System of Nonlinear Equations	89
4.1	Solving a Single Nonlinear Equation	89
4.1.1	The Bisection Method	89
4.1.2	The Newton-Raphson Method	91
4.1.3	The Secant Method	93
4.1.4	The Iterative Method Towards a Fixed Point	94
4.1.5	Using the MATLAB and Python solve Function	96
4.2	Solving a System of Nonlinear Equations	97
II	Data Interpolation and Solutions of Differential Equations	103
5	Data Interpolation	105
5.1	Lagrange Interpolation	105
5.1.1	Construction of Lagrange Interpolating Polynomial	105
5.1.2	Uniqueness of Lagrange Interpolation Polynomial	107
5.1.3	Lagrange Interpolation Error	108
5.2	Newton's Interpolation	109
5.2.1	Description of the Method	109
5.2.2	Newton's Divided Differences	113
5.3	MATLAB's Interpolation Tools	116
5.3.1	Interpolation with the interp1 Function	116
5.3.2	Interpolation with the Spline Function	117
5.3.3	Interpolation with the Function pchip	119
5.3.4	Calling the Functions spline and pchip from interp1	120
5.4	Data Interpolation in Python	120
5.4.1	The Function interp1d	121
5.4.2	The Functions pchip_interpolate and CubicSpline	122
5.4.3	The Function lagrange	123

6	Numerical Differentiation and Integration	125
6.1	Numerical Differentiation	125
6.1.1	Approximating Derivatives with Finite Differences	125
6.2	Numerical Integration	133
6.2.1	Newton-Cotes Methods	134
6.2.2	The Gauss Integration Method	143
7	Solving Systems of Nonlinear Ordinary Differential Equations	165
7.1	Runge-Kutta Methods	165
7.2	Explicit Runge-Kutta Methods	167
7.2.1	Euler's Method	168
7.2.2	Heun's Method	171
7.2.3	The Fourth-Order Runge-Kutta Method	174
7.3	Implicit Runge-Kutta Methods	176
7.3.1	The Backward Euler Method	176
7.3.2	Collocation Runge-Kutta Methods	180
7.3.2.1	Legendre-Gauss Methods	180
7.3.2.2	Lobatto Methods	184
7.4	MATLAB ODE Solvers	191
7.4.1	MATLAB ODE Solvers	191
7.4.2	Solving a Single IVP	191
7.4.3	Solving a System of IVPs	193
7.4.4	Solving Stiff Systems of IVPs	195
7.5	Python Solvers for IVPs	197
7.5.1	Solving ODEs with odeint	197
7.5.2	Solving ODEs with Gekko	201
8	Nonstandard Finite Difference Methods for Solving ODEs	207
8.1	Deficiencies with Standard Finite Difference Schemes	207
8.2	Construction Rules of Nonstandard Finite Difference Schemes	213
8.3	Exact Finite Difference Schemes	217
8.3.1	Exact Finite Difference Schemes for Homogeneous Linear ODEs	218
8.3.1.1	Exact Finite Difference Schemes for a Linear Homogeneous First-Order ODE	218
8.3.1.2	Exact Finite Difference Scheme for Linear Homogeneous Second Order ODE	220
8.3.1.3	Exact Finite Difference Scheme for a System of Two Linear ODEs	223
8.3.2	Exact Difference Schemes for Nonlinear Equations	230
8.3.3	Exact Finite Difference Schemes for Differential Equations with Linear and Power Terms	234
8.4	Other Nonstandard Finite Difference Schemes	236

III Solving Linear, Nonlinear and Dynamic Optimization Problems	241
9 Solving Optimization Problems: Linear and Quadratic Programming	243
9.1 Form of a Linear Programming Problem	243
9.2 Solving Linear Programming Problems with <code>linprog</code>	246
9.3 Solving Linear Programming Problems with <code>fmincon</code> MATLAB's Functions	249
9.4 Solving Linear Programming Problems with <code>pulp</code> Python	250
9.5 Solving Linear Programming Problems with <code>pyomo</code>	252
9.6 Solving Linear Programming Problems with <code>gekko</code>	254
9.7 Solving Quadratic Programming Problems	255
10 Solving Optimization Problems: Nonlinear Programming	261
10.1 Solving Unconstrained Problems	261
10.1.1 Line Search Algorithm	263
10.1.2 The Steepest Descent Method	264
10.1.3 Newton's Method	266
10.1.4 Quasi Newton's Methods	269
10.1.4.1 The Broyden-Fletcher-Goldfarb-Shanno (BFGS) Method	269
10.1.4.2 The Davidon-Fletcher-Powell (DFP) Algorithm	272
10.1.5 Solving Unconstrained Optimization Problems with MATLAB	274
10.1.6 Solving an Unconstrained Problem with Python	275
10.1.7 Solving Unconstrained Optimization Problems with Gekko	276
10.2 Solving Constrained Optimization Problems	278
10.2.1 Solving Constrained Optimization Problems with MATLAB <code>fmincon</code> Function	280
10.2.2 Solving Constrained Minimization Problems in Python	284
10.2.3 Solving Constrained Optimization with Gekko Python	286
11 Solving Optimal Control Problems	289
11.1 Introduction	289
11.2 The First-Order Optimality Conditions and Existence of Optimal Control	290
11.3 Necessary Conditions of the Discretized System	293
11.4 Numerical Solution of Optimal Control	294
11.5 Solving Optimal Control Problems Using Indirect Methods	295
11.5.1 Numerical Solution Using Indirect Transcription Method	295

11.6 Solving Optimal Control Problems Using Direct Methods . . . 306

 11.6.1 Statement of the Problem 307

 11.6.2 The Control Parameterization Technique 307

 11.6.2.1 Examples 309

 11.6.3 The Gekko Python Solver 313

Bibliography **321**

Index **327**

The past few decades have witnessed tremendous development in the manufacture of computers and software, and scientific computing has become an important tool for finding solutions to scientific problems that arise from various branches of science and engineering. Nowadays, scientific computing has become one of the most important means of research and learning in the fields of science and engineering, which are indispensable to any researcher, teacher, or student in the fields of science and engineering.

One of the most important branches of scientific computing is numerical analysis which deals with the issue of finding approximate numerical solutions to such problems and analyzing errors related to such approximate methods. Both the MATLAB® and Python programming languages provide many libraries that can be used to find solutions of scientific problems visualizing them. The ease of use of these two languages became the most languages that most scientists who use computers to solve scientific problems care about.

The idea of this book came after I taught courses of scientific computing for physics students, introductory and advanced courses in mathematical software and mathematical computer applications in many Universities in Africa and the gulf area. I also conducted some workshops for mathematics and science students who are interested in computational mathematics in some European Universities. In these courses and workshops, MATLAB and Python were used for the implementation of the numerical approximation algorithms. Hence, the purpose of introducing this book is to provide the student with a practical guide to solve mathematical problems using MATLAB and Python software without the need for third-party assistance. Since numerical analysis is concerned with the problems of approximation and analysis of errors of numerical methods associated with approximation methods, this book is more concerned with how these two aspects are applied in practice by software, where illustrations and tables are used to clarify approximate solutions, errors and speed of convergence, and its relation to some of the numerical method parameters, such as stepsize and tolerance. MATLAB and Python are the most popular programming languages for mathematicians, scientists, and engineers. Both the two programming languages possess various libraries for numerical and symbolic computations and data representation and visualization. Proficiency with the computer programs contained in this book requires that the student have prior knowledge of the basics of the programming languages MATLAB and Python, such as branching, loops, symbolic packages, and the graphical