Contents

1 The First Few Steps 1
1.1 What Is a Program? And What Is Programming? 1
1.2 A Matlab Program with Variables 3
1.2.1 The Program 3
1.2.2 Dissection of the Program 4
1.2.3 Why Not Just Use a Pocket Calculator? 5
1.2.4 Why You Must Use a Text Editor to Write Programs 6
1.2.5 Write and Run Your First Program 6
1.3 A Matlab Program with a Library Function 7
1.4 A Matlab Program with Vectorization and Plotting 8
1.5 More Basic Concepts 10
1.5.1 Using Matlab Interactively 10
1.5.2 Arithmetics, Parentheses and Rounding Errors 11
1.5.3 Variables 11
1.5.4 Formatting Text and Numbers 12
1.5.5 Arrays 14
1.5.6 Plotting 15
1.5.7 Error Messages and Warnings 18
1.5.8 Input Data 19
1.5.9 Symbolic Computations 19
1.5.10 Concluding Remarks 21
1.6 Exercises 22

2 Basic Constructions 25
2.1 If Tests 25
2.2 Functions 27
2.3 For Loops 32
2.4 While Loops 35
2.5 Reading from and Writing to Files 36
2.6 Exercises 38

3 Computing Integrals 47
3.1 Basic Ideas of Numerical Integration 48
3.2 The Composite Trapezoidal Rule 49

3.2.1 The General Formula . . . 51
3.2.2 Implementation . . . 52
3.2.3 Alternative Flat Special-Purpose Implementation . . . 54
3.3 The Composite Midpoint Method . . . 57
3.3.1 The General Formula . . . 58
3.3.2 Implementation . . . 58
3.3.3 Comparing the Trapezoidal and the Midpoint Methods . . . 59
3.4 Testing . . . 60
3.4.1 Problems with Brief Testing Procedures . . . 60
3.4.2 Proper Test Procedures . . . 61
3.4.3 Finite Precision of Floating-Point Numbers . . . 62
3.4.4 Constructing Unit Tests and Writing Test Functions . . . 64
3.5 Vectorization . . . 67
3.6 Measuring Computational Speed . . . 69
3.7 Double and Triple Integrals . . . 69
3.7.1 The Midpoint Rule for a Double Integral . . . 69
3.7.2 The Midpoint Rule for a Triple Integral . . . 73
3.7.3 Monte Carlo Integration for Complex-Shaped Domains . . . 76
3.8 Exercises . . . 80

4 Solving Ordinary Differential Equations . . . 87
4.1 Population Growth . . . 88
4.1.1 Derivation of the Model . . . 89
4.1.2 Numerical Solution . . . 91
4.1.3 Programming the Forward Euler Scheme; the Special Case 94
4.1.4 Understanding the Forward Euler Method . . . 97
4.1.5 Programming the Forward Euler Scheme; the General Case 97
4.1.6 Making the Population Growth Model More Realistic . . . 98
4.1.7 Verification: Exact Linear Solution of the Discrete Equations 101
4.2 Spreading of Diseases . . . 102
4.2.1 Spreading of a Flu . . . 102
4.2.2 A Forward Euler Method for the Differential Equation System . . . 105
4.2.3 Programming the Numerical Method; the Special Case . . . 105
4.2.4 Outbreak or Not . . . 106
4.2.5 Abstract Problem and Notation . . . 108
4.2.6 Programming the Numerical Method; the General Case . . . 109
4.2.7 Time-Restricted Immunity . . . 111
4.2.8 Incorporating Vaccination . . . 111
4.2.9 Discontinuous Coefficients: a Vaccination Campaign . . . 114
4.3 Oscillating One-Dimensional Systems . . . 115
4.3.1 Derivation of a Simple Model . . . 115
4.3.2 Numerical Solution . . . 117
4.3.3 Programming the Numerical Method; the Special Case . . . 117
4.3.4 A Magic Fix of the Numerical Method . . . 120
4.3.5 The 2nd-Order Runge-Kutta Method (or Heun's Method) . . . 122
4.3.6 Software for Solving ODEs . . . 123
4.3.7 The 4th-Order Runge-Kutta Method . . . 130

4.3.8 More Effects: Damping, Nonlinearity, and External Forces 133
4.3.9 Illustration of Linear Damping . . . 136
4.3.10 Illustration of Linear Damping with Sinusoidal Excitation . 137
4.3.11 Spring-Mass System with Sliding Friction . . . 138
4.3.12 A Finite Difference Method; Undamped, Linear Case . . . 141
4.3.13 A Finite Difference Method; Linear Damping . . . 143
4.4 Exercises . . . 144

5 Solving Partial Differential Equations . . . 153
5.1 Finite Difference Methods . . . 155
5.1.1 Reduction of a PDE to a System of ODEs . . . 156
5.1.2 Construction of a Test Problem with Known Discrete Solution . . . 158
5.1.3 Implementation: Forward Euler Method . . . 158
5.1.4 Application: Heat Conduction in a Rod . . . 160
5.1.5 Vectorization . . . 165
5.1.6 Using Odespy to Solve the System of ODEs . . . 165
5.1.7 Implicit Methods . . . 166
5.2 Exercises . . . 169

6 Solving Nonlinear Algebraic Equations . . . 177
6.1 Brute Force Methods . . . 178
6.1.1 Brute Force Root Finding . . . 179
6.1.2 Brute Force Optimization . . . 181
6.1.3 Model Problem for Algebraic Equations . . . 182
6.2 Newton's Method . . . 183
6.2.1 Deriving and Implementing Newton's Method . . . 183
6.2.2 Making a More Efficient and Robust Implementation . . . 186
6.3 The Secant Method . . . 189
6.4 The Bisection Method . . . 191
6.5 Rate of Convergence . . . 193
6.6 Solving Multiple Nonlinear Algebraic Equations . . . 196
6.6.1 Abstract Notation . . . 196
6.6.2 Taylor Expansions for Multi-Variable Functions . . . 196
6.6.3 Newton's Method . . . 197
6.6.4 Implementation . . . 198
6.7 Exercises . . . 199

References . . . 203

Index . . . 205

http://www.springer.com/978-3-319-32451-7

Programming for Computations - MATLAB/Octave
A Gentle Introduction to Numerical Simulations with MATLAB/Octave

Linge, S.; Langtangen, H.P.

2016, XVI, 216 p. 43 illus., Hardcover
ISBN: 978-3-319-32451-7