APPENDIX A
MATLAB BUILT-IN FUNCTIONS

abs, 40
acos, 40
ascii, 64
axis, 51
axis square, 45
beep, 76
besselj, 451
ceil, 40
chol, 285
clabel, 212
clear, 64
cond, 296, 297
contour, 212, 566
conv, 186
cumtrapz, 510, 511
deconv, 185
det, 253
diag, 338
diff, 560, 561, 562
disp, 61
double, 61
eig, 336
elfun, 40
eps, 112
erf, 544
error, 66
event, 622
exp, 40
eye, 336
factorial, 51, 73n3
fft, 420
fix, 459
floor, 40, 212
fminbnd, 137, 210, 211
fminsearch, 213, 215, 199
format bank, 30
format compact, 28n1
format long, 30, 102
format long e, 30, 300
format long eng, 30
format long g, 30
format loose, 28n1
format short, 30
format short e, 30, 300
format short eng, 30
format short g, 30
fplot, 82
fprintf, 62, 63
fzero, 137, 181–183
getframe, 77, 78
gradient, 563, 564
grid, 42
help, 39, 46
help elfun, 40
hist, 353
hold off, 44
hold on, 44
humps, 94, 540
inline, 82
input, 61
interp1, 470
interp2, 475
interp3, 475
inv, 236, 238
isempty, 105
legend, 398, 511
length, 37
LineWidth, 44
linspace, 84
load, 64, 65
log, 39
log10, 371
log2, 150n2
loglog, 51
logspace, 34
lookfor, 46, 55
lu, 282
MarkerEdgeColor, 44
MarkerFaceColor, 44
MarkerSize, 44
max, 41, 262
mean, 57, 352
median, 352
mesh, 76
meshgrid, 212, 566
min, 41, 352
mode, 352
movie, 77–78
nargin, 71–72
norm, 296
ode113, 618
ode15s, 632
ode23, 617
ode23s, 632
ode23t, 632
ode23tb, 632

ode45, 618, 633
odeset, 620
ones, 32
optimset, 182, 183, 189
pause, 76
pchip, 468, 471
peaks, 572
pi, 30
plot, 42
plot3, 44, 608
poly, 184, 185
polyfit, 373, 433, 446
polyval, 373, 433, 446
prod, 41
quiver, 565
rand, 353–356
randn, 353, 356
realmax, 112
realmin, 112
roots, 184–187
round, 40
save, 63
semilogy, 50, 51
set, 479
sign, 69
sin, 40
size, 237
sort, 41
spline, 468
sqrt, 40
sqrtm, 40
std, 352
stem, 423
subplot, 44
sum, 40, 285
surfc, 212
tanh, 7, 35
tic, 76
title, 42
toc, 76
trapz, 510, 518
var, 352
varargin, 86
who, 32
whos, 32
xlabel, 42
ylabel, 42
ylim, 424
zeros, 32
zlabel, 212

APPENDIX B

MATLAB M-FILE FUNCTIONS

M-file Name	Description	Page
`bisect`	Root location with bisection	151
`eulode`	Integration of a single ordinary differential equation with Euler's method	586
`fzerosimp`	Brent's method for root location	180
`GaussNaive`	Solving linear systems with Gauss elimination without pivoting	258
`GaussPivot`	Solving linear systems with Gauss elimination with partial pivoting	263
`GaussSeidel`	Solving linear systems with the Gauss-Seidel method	310
`goldmin`	Minimum of one-dimensional function with golden-section search	208
`incsearch`	Root location with an incremental search	144
`IterMeth`	General algorithm for iterative calculation	105
`Lagrange`	Interpolation with the Lagrange polynomial	443
`linregr`	Fitting a straight line with linear regression	372
`natspline`	Cubic spline with natural end conditions	477
`Newtint`	Interpolation with the Newton polynomial	440
`newtmult`	Root location for nonlinear systems of equations	318
`newtraph`	Root location with the Newton-Raphson method	174
`quadadapt`	Adaptive quadrature	539
`rk4sys`	Integration of system of ODEs with 4th-order RK method	602
`romberg`	Integration of a function with Romberg integration	530
`TableLook`	Table lookup with linear interpolation	458
`trap`	Integration of a function with the composite trapezoidal rule	500
`trapuneq`	Integration of unequispaced data with the trapezoidal rule	509
`Tridiag`	Solving tridiagonal linear systems	266

APPENDIX C
INTRODUCTION TO SIMULINK

Simulink® is a graphical programming environment for modeling, simulating, and analyzing dynamic systems. In short, it allows engineers and scientists to build process models by interconnecting blocks with communication lines. Thus, it provides an easy-to-use computing framework to quickly develop dynamic process models of physical systems. Along with offering a variety of numerical integration options for solving differential equations, Simulink includes built-in features for graphical output which significantly enhance visualization of a system's behavior.

As a historical footnote, back in the Pleistocene days of analog computers (aka the 1950s), you had to design information flow diagrams that showed graphically how multiple ODEs in models were interconnected with themselves and with algebraic relationships. The diagrams also showed flaws in modeling where there was information lacking or structural defects. One of the nice features of Simulink is that it also does that. It's often beneficial to see that aspect separate from numerical methods and then merge the two in MATLAB.

As was done with Chap. 2, most of this appendix has been written as a hands-on exercise. That is, you should read it while sitting in front of your computer. The most efficient way to start learning Simulink is to actually implement it on MATLAB as you proceed through the following material.

So let's get started by setting up a simple Simulink application to solve an initial value problem for a single ODE. A nice candidate is the differential equation we developed for the velocity of the free-falling bungee jumper in Chap. 1,

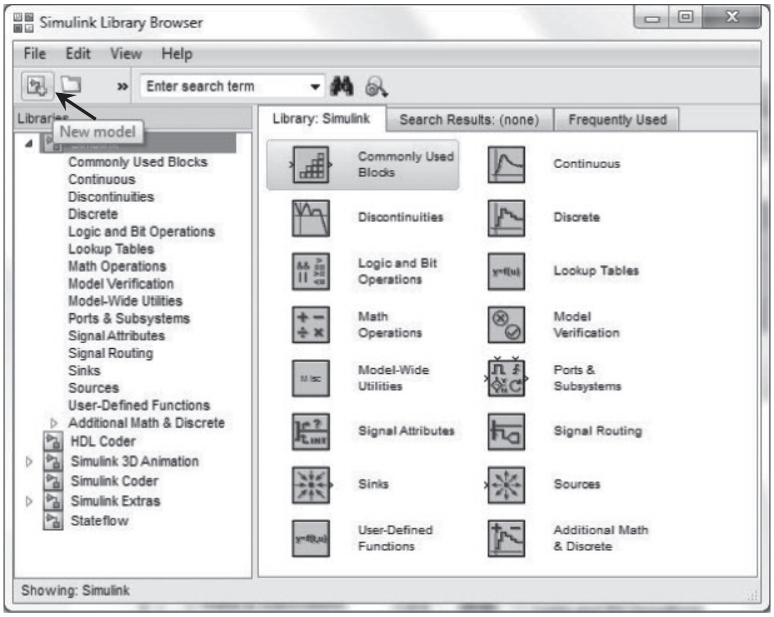
$$\frac{dv}{dt} = g - \frac{c_d}{m}v^2 \tag{C.1}$$

where v = velocity (m/s), t = time (s), $g = 9.81$ m/s^2, c_d = drag coefficient (kg/m), and m = mass (kg). As in Chap. 1 use $c_d = 0.25$ kg/m, $m = 68.1$ kg, and integrate from 0 to 12 s with an initial condition of $v = 0$.

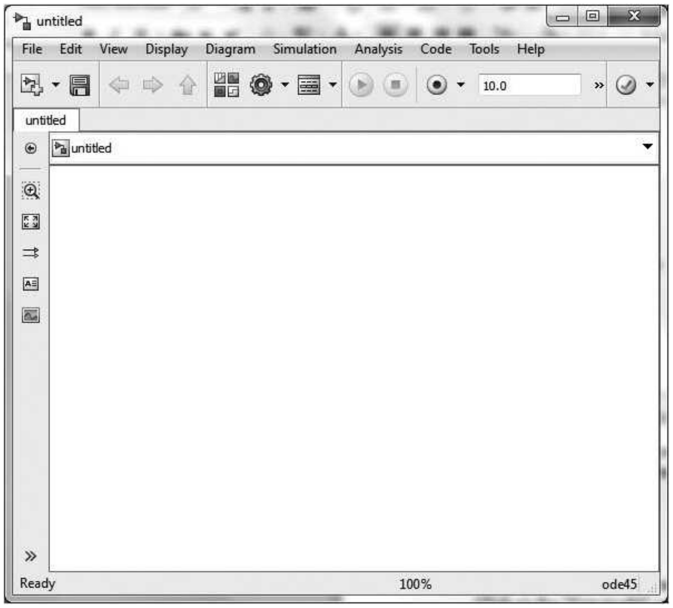
To generate the solution with Simulink, first launch MATLAB. You should eventually see the MATLAB window with an entry prompt, ≫, in the Command Window. After changing MATLAB's default directory, open the Simulink Library Browser using one of the following approaches:

- On the MATLAB toolbar, click the Simulink button (▦).
- At the MATLAB prompt, enter the `simulink` command.

The Simulink Library Browser window should appear displaying the Simulink block libraries installed on your system. Note, to keep the Library Browser above all other windows on your desktop, in the Library Browser select **View, Stay on Top.**

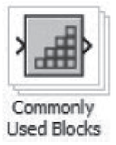


Click on the **New model** command button on the left of the toolbar, and an untitled Simulink model window should appear.

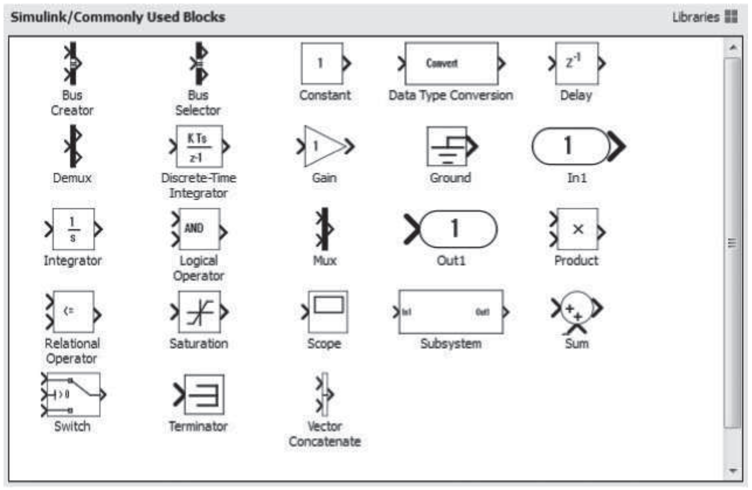

untitled
File Edit View Display Diagram Simulation Analysis Code Tools Help
10.0
untitled
Ready
100%
ode45

You will build your simulation model in the untitled window by choosing items from the Library Browser and then dragging and dropping them onto the untitled window. First, select the untitled window to activate it (clicking on the title bar is a sure way to do this) and select **Save As** from the File menu. Save the window as **Freefall** in the default directory. This file is saved automatically with an **.slx** extension. As you build your model in this window, it is a good idea to save it frequently. You can do this in three ways: the **Save** button, the **Ctrl+s** keyboard shortcut, and the menu selections, **File, Save.**

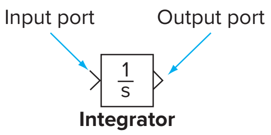
We will start by placing an integrator element (for the model's differential equation) in the **Freefall** window. To do this, you need to activate the **Library Browser** and double click on the **Commonly Used Blocks** item.



The Browser window should show something like



Examine the icon window until you see the Integrator icon.

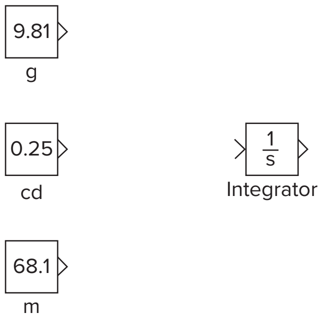


The block symbol is what will appear in your model window. Notice how the icon has both input and output ports which are used to feed values in and out of the block. The 1/s symbol represents integration in the Laplace domain. Use the mouse to drag an **Integrator** icon onto your **Freefall** window. This icon will be used to integrate the differential equations. Its input will be the differential equation [the right hand side of Eq. (C.1)] and its output will be the solution (in our example, velocity).

Next you have to build an information flow diagram that describes the differential equation and "feeds" it into the integrator. The first order of business will be to set up constant blocks to assign values to the model parameters. Drag a **Constant** icon from the **Commonly Used Blocks**[1] branch in the **Browser** window to the **Freefall** model window and place it above and to the left of the integrator.

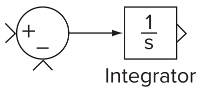
Next, click on the **Constant Label** and change it to g. Then, double click the Constant block and the Constant Block Dialogue box will be opened. Change the default value in the Constant field to 9.81 and click OK.

Now set up Constant blocks for both the drag coefficient (cd = 0.25) and mass (m = 68.1) positioned below the g block. The result should look like



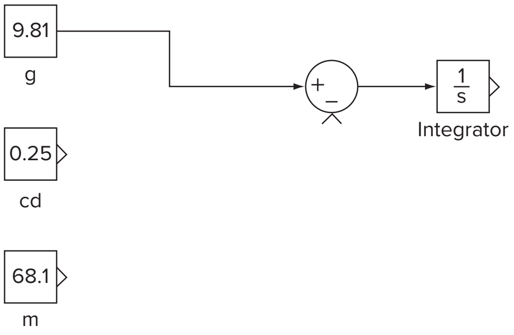
From the Math Operations Library Browser, select a Sum icon and drag it just to the left of the Integrator block. Place the mouse pointer on the output port of the sum block. Notice that the mouse pointer will change to a crosshair shape when it's on the output port. Then drag a connecting line from the output port to the Integrator block's input port. As you drag, the mouse pointer retains its crosshair shape until it is on the input port whereupon it changes to a double-lined crosshair. We have now "wired" the two blocks together with the output of the sum block feeding into the Integrator.

Notice that the sum block has two input ports into which can be fed two quantities that will be added as specified by the two positive signs inside the circular block. Recall that our differential equation consists of the difference between two quantities: $g - (c_d/m)v^2$. We therefore, have to change one of the input ports to a negative. To do this, double click on the Sum Block in order to open the Sum Block Dialogue box. Notice that the List of Signs has two plus signs (++). By changing the second to a minus sign (+−), the value entering the second input port will be subtracted from the first. After closing the Sum Block Dialogue box, the result should look like

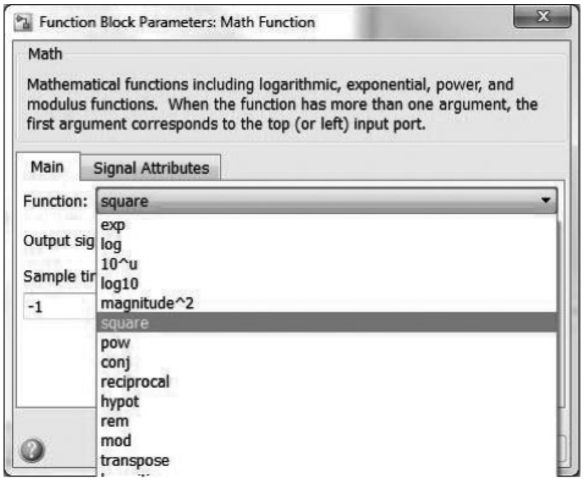


[1] All the icons in the **Commonly Used Blocks** group are available in other groups. For example, the **Constant** icon is located in the **Sources** group.

Because the first term in the differential equation is g, wire the output of the g block to the positive input port of the sum block. The system should now look like



In order to construct the second term that will be subtracted from g, we must first square the velocity. This can be done by dragging a Math Functions block from the Math Operations Library Browser and positioning it to the right and below the integrator block. Double click the Math Functions icon to open the Math Functions Dialogue box and use the pull down menu to change the Math Function to square.

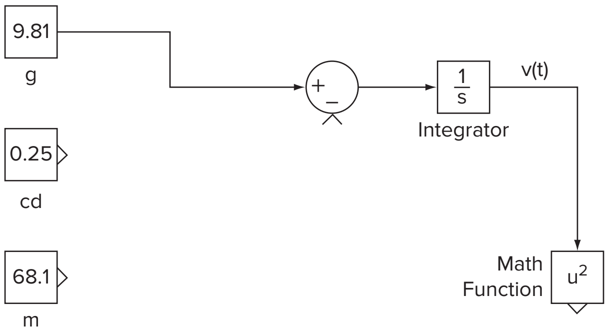

Function Block Parameters: Math Function

Math

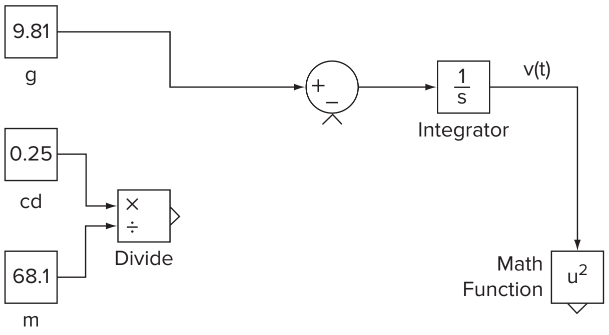
Mathematical functions including logarithmic, exponential, power, and modulus functions. When the function has more than one argument, the first argument corresponds to the top (or left) input port.

Function: square

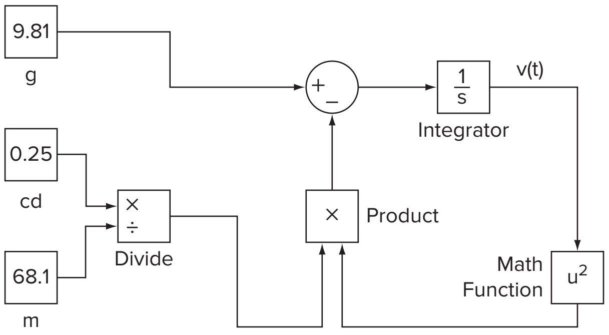
Before connecting the Integrator and Square blocks, it would be nice to rotate the latter so its input port is on top. To do this, select the Square block and then hit ctrl-r once. Then, we can wire the output port of the Integrator block to the input port of the square block. Because the output of the Integrator block is the solution of the differential equation, the output of the square block will be v^2. To make this clearer, double click the arrow connecting the Integrator and Square blocks and a text box will appear. Add the label, v(t), in this text box to indicate that the output of the Integrator block is the velocity. Note that you can label all the connecting wires in this way in order to better document the system diagram. At this point, it should look like



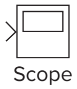
Next, drag a Divide block from the Math Operations Library Browser and position it to the right of the cd and m blocks. Notice that the Divide block has two input ports: one for the dividend (×) and one for the divisor (÷). Note that these can be switched by double clicking the Divide block to open the Divide Block Dialogue box and switching the order in the "number of inputs:" field. Wire the cd block output port to the × input port and the m block output port to the ÷ input port of the Divide block. The output port of the divide block will now carry the ratio, c_d/m.



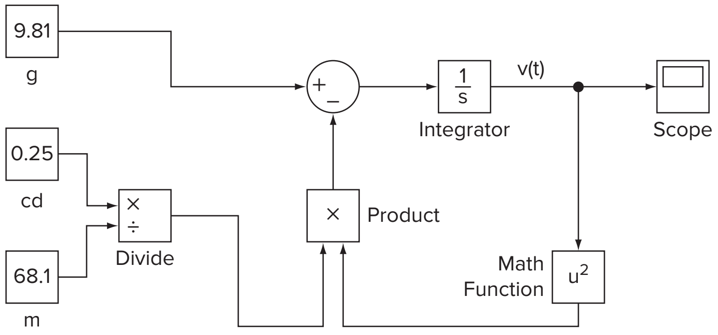
Drag a Product block from the Math Operations Library Browser and position it to the right of the divide block and just below the sum block. Use ctrl-r to rotate the product block until its output port points upward toward the sum block. Wire the divide block output port to the nearest product input port and the square Math Function block output port to the other product input port. Finally, wire the product output port to the remaining sum input port.



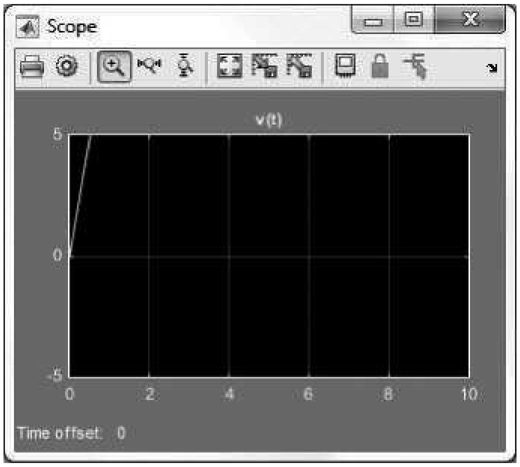
As displayed above, we have now successfully developed a Simulink program to generate the solution to this problem. At this point, we could run the program but we have not yet set up a way to display the output. For the present case, a simple way to do this employs a Scope Block



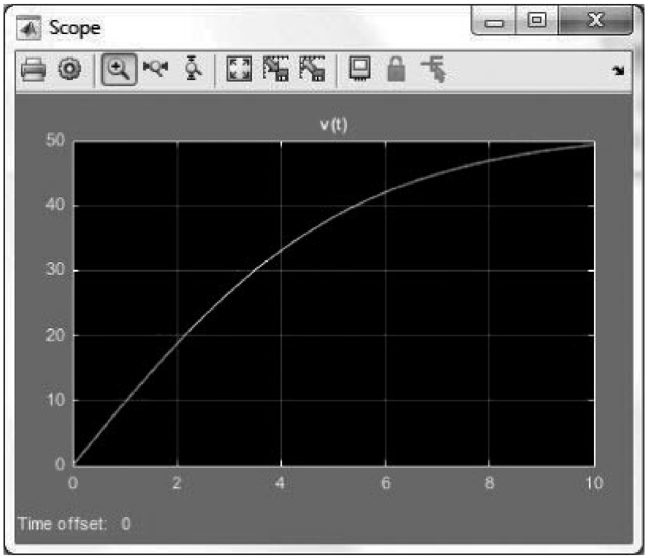
The Scope block displays signals with respect to simulation time. If the input signal is continuous, the Scope draws a point-to-point plot between major time step values. Drag a Scope block from the **Commonly Used Blocks** browser and position it to the right of the Integrator block. Position the mouse pointer on the Integrator's output wire (for the present case a nice position would be at the corner). Simultaneously hold down the control key and another line across to the Scope block's input port.



We are now ready to generate results. Before doing that, it's a good idea to save the model. Double click on the Scope block. Then click the run button, ▶. If there are any mistakes, you will have to correct them. Once your have successfully made corrections, the program should execute and the scope display should look something like



Click on the Autoscale button, ⛶, and the plot will resize to fit the entire range of results



Note that a Scope window can display multiple *y*-axes (graphs) with one graph per input port. All of the *y*-axes have a common time range on the *x*-axis. By selecting the parameter button on the graph window (⚙), you can use scope parameters to change graph features such as figure color and style and axis settings.

BIBLIOGRAPHY

Anscombe, F. J., “Graphs in Statistical Analysis,” *Am. Stat., 27*(1):17–21, 1973.

Attaway, S., *MATLAB: A Practical Introduction to Programming and Problem Solving,* Elsevier Science, Burlington, MA, 2009.

Bogacki, P. and L. F. Shampine, “A 3(2) Pair of Runge-Kutta Formulas,” *Appl. Math. Letters, 2*(1989):1–9, 1989.

Brent, R. P., *Algorithms for Minimization Without Derivatives,* Prentice Hall, Englewood Cliffs, NJ, 1973.

Butcher, J. C., “On Runge-Kutta Processes of Higher Order,” *J. Austral. Math. Soc., 4*:179, 1964.

Carnahan, B., H. A. Luther, and J. O. Wilkes, *Applied Numerical Methods,* Wiley, New York, 1969.

Chapra, S. C. and R. P. Canale, *Numerical Methods for Engineers,* 6th ed., McGraw-Hill, New York, 2010.

Cooley, J. W. and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Math. Comput., 19*:297–301, 1965.

Dekker, T. J., “Finding a Zero by Means of Successive Linear Interpolation.” In B. Dejon and P. Henrici (editors), *Constructive Aspects of the Fundamental Theorem of Algebra,* Wiley-Interscience, New York, 1969, pp. 37–48.

Devaney, R. L. Chaos, Fractals, and Dynamics: Computer Experiments in Mathematics, Menlo Park, CA: Addison-Wesley, 1990.

Dormand, J. R. and P. J. Prince, “A Family of Embedded Runge-Kutta Formulae,” *J. Comp. Appl. Math., 6*:19–26, 1980.

Draper, N. R. and H. Smith, *Applied Regression Analysis,* 2nd ed., Wiley, New York, 1981.

Fadeev, D. K. and V. N. Fadeeva, *Computational Methods of Linear Algebra,* Freeman, San Francisco, CA, 1963.

Forsythe, G. E., M. A. Malcolm, and C. B. Moler, *Computer Methods for Mathematical Computation,* Prentice Hall, Englewood Cliffs, NJ, 1977.

Gabel, R. A. and R. A. Roberts, *Signals and Linear Systems,* Wiley, New York, 1987.

Gander, W. and W. Gautschi, *Adaptive Quadrature–Revisited, BIT Num. Math., 40*:84–101, 2000.

Gerald, C. F. and P. O. Wheatley, *Applied Numerical Analysis,* 3rd ed., Addison-Wesley, Reading, MA, 1989.

Hanselman, D. and B. Littlefield, *Mastering MATLAB 7,* Prentice Hall, Upper Saddle River, NJ, 2005.

Hayt, W. H. and J. E. Kemmerly, *Engineering Circuit Analysis,* McGraw-Hill, New York, 1986.

Heideman, M. T., D. H. Johnson, and C. S. Burrus, “Gauss and the History of the Fast Fourier Transform,” *IEEE ASSP Mag., 1*(4):14–21, 1984.

Hornbeck, R. W., *Numerical Methods,* Quantum, New York, 1975.

James, M. L., G. M. Smith, and J. C. Wolford, *Applied Numerical Methods for Digital Computations with FORTRAN and CSMP,* 3rd ed., Harper & Row, New York, 1985.

Moler, C. B., *Numerical Computing with MATLAB,* SIAM, Philadelphia, PA, 2004.

Moore, H., *MATLAB for Engineers,* 2nd ed., Prentice Hall, Upper Saddle River, NJ, 2008.

Munson, B. R., D. F. Young, T. H. Okiishi, and W. D. Huebsch, *Fundamentals of Fluid Mechanics*, 6th ed., Wiley, Hoboken, NJ, 2009.

Ortega, J. M., *Numerical Analysis–A Second Course,* Academic Press, New York, 1972.

Palm, W. J. III, *A Concise Introduction to MATLAB,* McGraw-Hill, New York, 2007.

Ralston, A., “Runge-Kutta Methods with Minimum Error Bounds,” *Match. Comp., 16*:431, 1962.

Ralston, A. and P. Rabinowitz, *A First Course in Numerical Analysis,* 2nd ed., McGraw-Hill, New York, 1978.

Ramirez, R. W., *The FFT, Fundamentals and Concepts,* Prentice Hall, Englewood Cliffs, NJ, 1985.

Recktenwald, G., *Numerical Methods with MATLAB,* Prentice Hall, Englewood Cliffs, NJ, 2000.

Scarborough, I. B., *Numerical Mathematical Analysis,* 6th ed., Johns Hopkins Press, Baltimore, MD, 1966.

Shampine, L. F., *Numerical Solution of Ordinary Differential Equations,* Chapman & Hall, New York, 1994.

Van Valkenburg, M. E., *Network Analysis*, Prentice Hall, Englewood Cliffs, NJ, 1974.

White, F. M., *Fluid Mechanics.* McGraw-Hill, New York, 1999.

INDEX

A

Absolute error, 101
Absolute tolerance (AbsTol), 618
Accuracy, 100–101
Adaptive integration, 487
Adaptive methods
  adaptive Runge-Kutta methods, 615–623
  MATLAB application, 634–635
  multistep methods, 624–628
  Pliny's intermittent fountain, 635–639
Adaptive quadrature, 525, 537. *See also* Gauss quadrature; Romberg integration
  example, 540
  integral function, 539
  quadadapt function, 537–539
Adaptive Runge-Kutta methods, 615. *See also* Runge-Kutta methods (RK methods)
  adaptive step-size control, 615–616
  events, 621–623
  MATLAB functions for nonstiff systems, 617–621
  solution of ODE, 616
Adaptive step-size control, 615–616
Allosteric enzymes, 374
Alphanumeric information, 34
Amplification factor, 585
Amplitude, 406
Analytical solution, 9
  for heated rod, 650–651
Angular frequency, 332, 406, 407
Animation, 77
  of projectile motion, 77–78
Anonymous functions, 81–82
Areal integral, 492
Arithmetic manipulations of computer numbers, 112
  adding large and small number, 114
  inner products, 114
  large computations, 113
  smearing, 114
Arithmetic mean, 348
Arrays, 31–32
  operations, 39
Ascent methods, 213
ASCII files in MATLAB, 64–65
Assignment function, 29
  arrays, 31–32
  character strings, 34–35
  colon operator, 33
  linspace functions, 33–34
  logspace functions, 33–34
  matrices, 31–32
  scalars, 29–30
  vectors, 31–32
Associative equation, 232
Aug function, 258
Augmentation, 234

B

Backslash operator, 394
Backward method, 630
Banded matrix, 231, 264
Base-2. *See* Binary
Base-8. *See* Octal
Bias. *See* Inaccuracy
Bilinear function, 474
Binary, 107
  digits, 106
  search, 458–459
bisect function, 151–152
Bisection method, 146, 147. *See also* Numerical methods
  bisect function, 151–152
  error estimation for, 148–151
  preferable to false position, 154–155
Bits, 106
Blunders, 130
Boole's rule, 538
Boolean variables, 636
Boundary-value problems, 577, 647
  in engineering and science, 648–651
  finite-difference methods, 658–665
  initial-value problem *vs.*, 647, 648
  MATLAB function, 665–667
  ordinary differential equation, 647
  shooting method, 651–658
  single ODE, 646
Bracketing methods. *See also* Numerical methods; Roots
  and initial guesses, 141
    incremental search, 143–146
Brent's method, 176. *See also* Secant methods
  algorithm, 179–181
  inverse quadratic interpolation, 177–179
Brent's root-finding method, 179
Brent's root-location method, 176
Built-in functions, 39–42
Bungee jumper
  with cord, 634–635
  problem
    analysis, 291–292
    analytical solution to, 7–10
    free-falling, 573
    numerical solution to, 10–12
    velocity, 87–90

Butterfly effect, 607
bvp4c function, 665–667

C

Calculator mode, 2, 28
Calculus, 549
Cartesian coordinates, 492
ceil function, 40
Centered finite difference, 123
Chaos, 604–609
Character Strings, 34–35
Characteristic polynomial, 328
Chemical reactions, 320–322
chol function, 285–286
Cholesky decomposition. *See also* Cholesky factorization
Cholesky factorization, 283
    example, 284
    MATLAB function, 285–286
Circuits, currents and voltages in, 240–243
Clamped end condition, 467
Classical fourth-order RK method, 595–598
Closed integration formulas, 487, 494
Closed-form solution, 9
Coefficient matrix, 251
Colebrook equation, 186
Colon operator, 33
Column, 229
    column-sum norm, 294
    vectors, 230
Command mode, 2
Commutative equation, 231, 232
Companion matrix, 184
Complete pivoting, 261
Composite Simpson's 1/3 rule, 503–505, 524–525
Composite trapezoidal rule, 497–499
Computer algorithm for iterative calculations, 104–106
Computer applications, 370
    MATLAB functions, polyfit and polyval, 373
    MATLAB M-file, linregr, 371–372
Computer mathematics, 1
Computer number representation, 106
    floating-point representation, 108–111
    integer representation, 107–108
    precision, 111–112
    range, 111
Computing work with numerical integration, 515–518
Concatenation, 32, 34
Conditionally stable method, 585
Conservation
    of charge, 240
    of energy, 241
    laws, 223
        in engineering and science, 12–13
Constant of integration, 575–576
Constitutive laws, 551, 552
Continuity condition, 460
Continuous Fourier series, 411
    approximation, 411–412
    Euler's formula, 413
    square wave, 412–413
Control codes, 62–63
Convergence, 164, 308
Cooley-Tukey algorithm, 420
Corrector equation, 588
Correlation coefficient, 364
Cosine, 406
    function, 407
Cramer's rule, 249, 250–253, 315
Creating and accessing files, 63–65
Cubic interpolation, 439
Cubic splines, 453, 462. *See also* Linear splines
    derivation, 463–465
    end conditions, 467–468
    material on, 455
    natural, 466–467
cumtrapz functions, 510
Currents in circuits, 240–243
Curvature, 550
Curve fitting
    with sinusoidal functions, 405, 407
        alternative formulation, 408
        least-squares fit of sinusoid, 408–411
        plot of sinusoidal function, 406
    techniques, 343
        through data points, 344
        engineering and science, 343–345
        linear regression, 345

D

Darcy's law, 552
Data uncertainty, 131
Decisions, 65
    error function, 66
    if structure, 65–66, 69–70
    if... else structure, 69
    if... elseif structure, 69
    logical conditions, 66–68
    switch structure, 70–71
    variable argument list, 71–72
Default value, 71
Definite integrals equation, 575
Deflation, 336
Degrees of freedom, 349
Dependent variable, 5, 573–574
Derivative, 485, 549, 550
    boundary conditions
        finite-difference methods, 660–662
        shooting method with, 654–656
    for data with errors, 558–559
    mean-value theorem, 119
    of unequally spaced data, 557–558
Descent methods, 213
Descriptive statistics, 348
    location measurement, 348–349
    in MATLAB, 352–353
    spread measurement, 349
    statistics of sample, 350–351
det function, 253
Determinants, 250–253
    evaluation with gauss elimination, 263–264
DFT. *See* Discrete Fourier transform (DFT)
Diagonal dominance, 308
Diagonal matrix, 230
diff function, 560–562
Differential equations, 7, 573, 574
Differentiation, 485–487, 549–551
    in engineering and science, 551–552
    one-dimensional forms of constitutive laws, 552
Direct methods, 213
Dirichlet boundary condition, 654
Discrete Fourier transform (DFT), 418
    FFT, 419–420
    MATLAB function, 420–423
    of simple sinusoid with MATLAB, 421–422
Discretization errors, 583
disp function, 61

Distributed variable problems, 225
Distributive, 232
Dot product. *See* Inner product of two vectors
Double integral, 512–513
  to determine average temperature, 513–514
Dummy variable, 89

E

Earthquakes, eigenvalues and, 337–339
Echo printing, 29
Eigenvalues, 327, 328
  and earthquakes, 337–339
  eig function, 336
  mass–three spring system, 331
  with MATLAB, 336–337
  physical background, 331
  physical interpretation of, 332–333
  polynomial method, 329–331
  positions and velocities *vs.* time, 327
  power method, 333–336
Eigenvectors, 226, 329
  with MATLAB, 336
  physical interpretation of, 332–333
Electromotive force (emf), 241
Element-by-element operations, 39
Elimination of unknowns, 253–254
Ellipsis, 34
Embedded RK methods, 616, 617
emf. *See* Electromotive force (emf)
End conditions, 467–468
Enzyme kinetics, 373–378
Equilibrium, 213–215
error function, 66
Error(s), 100
  absolute, 101
  accuracy, 100–101
  amplification, 559
  analysis for Euler's method, 583–585
  blunders, 130
  computer algorithm for iterative calculations, 104–106
  data uncertainty, 131
  estimation, 627–628
    for iterative methods, 103–104
  model errors, 130–131
  numerical, 101
  percent relative, 102
  precision, 100–101
  roundoff, 106–114
  total numerical error, 125–130
  truncation, 114–125
Euclid's definition, 203
Euclidean norm, 293
Euler-Cauchy method. *See* Euler's method
Euler's formula, 413
Euler's method, 10, 13, 577, 581, 582, 598, 629–630. *See also* Runge-Kutta methods (RK methods)
  error analysis for, 583–585
  example, 582–583
  fundamental source of error in, 587
  Heun's method, 588–592
  MATLAB M-file function, 586–587
  midpoint method, 592–593
  solving systems of ODEs with, 598–600
  stability of, 585–586
eulode function, 586–587
Events, 621–623
Explicit Euler method, 630–632
Exploratory data analysis, 46–48
Exponential model, 366
Extrapolation, 445–447. *See also* Interpolation
"Eyeball" approaches, 358

F

Factorial, loops to computing, 73–74
False position, 152, 177, 433–435, 470
  bisection method preferable to, 154–155
  method, 153–154
False-position formula, 152
Fast Fourier transform (FFT), 345, 419–420
fft function, 420–423
Fick's law, 552
50th percentile. *See* Median
Finite difference, 121
  approximations
    of derivatives, 124–125
    of higher derivatives, 125
  methods, 658
    of boundary-value problems, 659–660
    derivative boundary conditions, 660–662
    implementation, 659
    incorporating derivative boundary conditions, 662–663
    for nonlinear ODEs, 663–665
First derivative
  backward difference approximation, 123
  centered difference approximation, 123
First-order
  approximation, 116
  equation, 574
  method, 585
  splines, 456–457
Fitting experimentatal data, 397–399
Floating-point operations (flops), 258
Floating-point representation, 108–111
floor function, 40
flops. *See* Floating-point operations (flops)
fmin corresponds, 207
fminbnd function, 210–211
fminsearch function, 213
fminsearch MATLAB function, 395
for... end structure, 72–73
Forcing functions, 5
Format codes, 62
Forward elimination of unknowns, 255–256
Fourier analysis, 345, 405. *See also* Gauss elimination
  continuous Fourier series, 411–413
  curve fitting with sinusoidal functions, 405–411
  DFT, 418–423
  Fourier integral and transform, 415–418
  frequency and time domains, 414–415
  power spectrum, 423–424
  sunspots, 425–426
Fourier coefficients, 423
Fourier integral, 415
  amplitude and phase line spectra, 417
  aperiodic signal, 418
  various phases of sinusoid, 416
Fourier series, 415
  continuous, 411–413

Fourier transform, 415–418
 pair, 417
Fourier's law, 552, 655
 of heat conduction, 551
Fourth-order RK method, 600–601
fprintf function, 62
Free-falling bungee jumper problem, 573
Frequency domain, 414–415
Frequency plane, 415
Friction factor, 186
Frobenius norm, 294
fsolve function, 319–320
Function files, 55–57
Function functions, 81, 82–83
 building and implementing, 84–85
Fundamental frequency, 411
fval function, 213
fzero function, 181–183
fzerosimp function, 179

G

Gauss elimination, 225, 249. *See also* Fourier analysis
 determinants and Cramer's rule, 250–253
 elimination of unknowns, 253–254
 graphical method, 249–250
 as *LU* factorization, 276
  example, 278
  MATLAB function, 282–283
  matrix, 277
  with pivoting, 280–282
  substitution steps, 279–280
 model of heated rod, 266–269
 naive Gauss elimination, 254–261
 pivoting, 261–264
 solving small numbers of equations, 249
 tridiagonal systems, 264–266
Gauss quadrature, 487, 508, 525, 530–531. *See also* Adaptive quadrature; Romberg integration
 higher-point formulas, 536–537
 three-point Gauss-Legendre formula, 536–537
 two-point Gauss-Legendre formula, 533–536, 543
 undetermined coefficients method, 531–533
Gauss-Newton method, 395
Gauss-Seidel method, 226, 305–312, 664
 with relaxation, 311–312
GaussNaive function, 258
GaussPivot function, 262–263
Gear backward differentiation formulas, 632
General linear least-squares regression, 391–392. *See also* Nonlinear regression
 model, 345
 multiple linear regression, 389–391
 polynomial regression, 385–389
  with MATLAB, 392–393
 QR factorization and backslash operator, 394
Global optimum, 202
Global truncation error, 583–584
Global variables, 58–60
Golden ratio, 203
Golden-section search, 203–208
Gradien methods, 213
gradient function, 563–564
Graphical methods, 140–141, 249–250
Graphics, 42–45

H

Harmonics, 411
Heated rod model, 266–269
Hertz (Hz), 332
Heun's method, 588
 example, 589–592
 without iteration, 595
 predictor and corrector, 588
 predictor equation, 588
 predictor-corrector approach, 589
High-accuracy differentiation formulas, 552
 example, 553–555
 Taylor series, 552–553
Higher-order corrections, 527
Higher-order differential equations, 574
Higher-order polynomial interpolation, dangers of, 447–449
Higher-point formulas, 536–537
Hilbert matrix, 295
Histogram, 351–352
Homogeneous linear algebraic system, 328
Hooke's law, 213, 552, 580
humps function, 540
Hypothesis testing, 344

I

Identity matrix, 230
IEEE double-precision format, 111
if structure, 65–66, 69–70
if... else structure, 69
if... elseif structure, 69
Ill-conditioned ODEs, 585
Ill-conditioned systems, 249
Implicit Euler method, 630–632
Implicit method, 630
Implicit Runge-Kutta formula, 632
Imprecision, 100
Inaccuracy, 100
Increment function, 581, 593
Incremental search method, 143–146
Indefinite integral equation, 575
Indentation, 79–81
Independent variables, 5, 573–574
Indoor air pollution, 297–300
Infinite loop, 75
Initial-value problems, 577, 579, 647
 Euler's method, 581–593
 MATLAB M-file function, rk4sys, 602–603
 predator-prey models and chaos, 604–609
 RK methods, 593–598
 signum function, 580
 solving ODE, 581
 systems of equations, 598–604
Inner product of two vectors, 37
input function, 61
Input-output, 61
 creating and accessing files, 63–65
 interactive M-file function, 62–63
Integer representation, 107–108
integral function, 539, 543
 integral2 functions, 514
 integral3 functions, 514
Integrals, 492
 for data with errors, 558–559

Integration, 485–487, 489–490
  in engineering and science, 490–492
  Newton-Cotes open integration formulas, 512
  numerical integration to computing distance, 510–511
  trapezoidal rule with unequal segments, 508
  trapuneq function, 509
  trapz and cumtrapz, 510
  with unequal segments, 508
Interactive M-file function, 62–63
interp1 function, 468, 470
  options, 472
  trade-offs using, 471–473
Interpolation, 16, 343, 345, 430. *See also* Polynomial interpolation
  determining polynomial coefficients, 431–432
  polyfit functions, 433
  polyval functions, 433
Inverse Fourier transform, 417, 418
Inverse interpolation, 444–445. *See also* Interpolation
Inverse quadratic interpolation, 177–179
Iterative methods
  linear systems, 305–312
  nonlinear systems, 312–322

J

Jacobi iteration, 307
Joule's law, 541
Jumpers motion, 326

K

Kirchhoff's current, 240
Kirchhoff's voltage rule, 241
Knots, 457

L

Lagging phase angle, 407
Lagrange interpolating polynomial, 345, 441. *See also* Newton interpolating polynomial
  Lagrange function, 443–444
  rationale, 442
Lagrange polynomial, 178
Leading phase angle, 407
Least squares
  criterion, 360
  fit of sinusoid, 408–411
  regression, 343
Left division, polynomial regression implemention with, 394
Line spectra, 415
Linear algebraic equations, 223, 227, 248
  bungee cords, 228
  currents and voltages in circuits, 240–243
  in engineering and science, 223–225
  free-body diagrams, 228
  linear algebraic equations with MATLAB, 238–240
  matrix algebra, 229–238
Linear convergence, 166
Linear interpolation method. *See* False position
Linear Lagrange interpolating polynomial, 441
Linear least-squares regression, 358
  "criteria for best" fit, 358–360
  error quantification of linear regression, 362
  errors estimation for linear least-squares fit, 364–366
  least-squares fit of straight line, 360
  linear regression, 360–362
  linear regression with residual errors, 364
  regression data, 363
  residual in linear regression, 362–363
Linear ODE, shooting method for, 652–654
Linear regression, 345, 360–362, 370
  computer applications, 370–373
  descriptive statistics, 348–351
    in MATLAB, 352–353
  enzyme kinetics, 373–378
  experimental data for force and velocity, 347–348
  fluid mechanics, 346–347
  linear least-squares regression, 358–366
  linearization of nonlinear relationships, 366–370
  normal distribution, 351–352
  random numbers and simulation, 353–358
  statistics review, 348
  wind tunnel experiment, 347
Linear splines, 455. *See also* Cubic splines
  first-order splines, 456–457
  notation for, 456
  table lookup, 458–459
Linear systems, 305. *See also* Nonlinear systems
  chemical reactions, 320–322
  convergence and diagonal dominance, 308
  fsolve function, 319–320
  Gauss-Seidel method, 306–308
    GaussSeidel function, 309
  linear algebraic equations, 305–306
  relaxation, 309–312
Linearization of nonlinear relationships, 366
  comments on linear regression, 370
  fitting data with power equation, 368–370
  nonlinear regression techniques, 367–368
  power equation, 366–367
linregr MATLAB M-file function, 371
linspace functions, 33–34
Local optimum, 202
Local truncation error, 583–584
Local variables, 57, 58
Logical conditions, 66–68
Logical variables, 636
logspace functions, 33–34
Loop(s), 65, 72
  to computing factorial, 73–74
  for... end structure, 72–73
  pause command, 76
  preallocation of memory, 74–75
  rule, 241
  vectorization, 74
  while structure, 75
  while... break structure, 75–76
Lorenz equations, 604, 606
Lotka-Volterra equations, 604, 607
Lotka-Volterra model, 606
Lower triangular matrix, 231

LU factorization, 225, 275
  Cholesky factorization, 283–286
  with Gauss elimination, 278
  Gauss elimination as, 276–283
  LU function, 282–283
  MATLAB left division, 286
  n-dimensional systems, 275
  two-step strategy, 276
Lumped drag coefficient, 7
Lumped variable problems, 224

M

M-files, 54, 143, 637
  bisect function, 151–152
  eulode function, 586–587
  function files, 55–57
  GaussNaive, 258
  GaussPivot, 262–263
  GaussSeidel, 309
  global variables, 58–60
  to implementing Lagrange interpolation, 443
  to implementing Newton interpolation, 440
  to implementing Romberg integration, 530
  Lagrange function, 443–444
  linregr M-file function, 371–372
  Newtint function, 440
  newtraph function, 173–174
  passing functions to, 81
    anonymous functions, 81–82
    building and implementing function function, 84–85
    function functions, 82–83
    passing parameters, 85–87
  quadadapt function, 537–539
  rk4sys, 602–603
  script files, 54–55
  subfunctions, 60–61
  trap function, 499–501
  trapuneq function, 509
  Tridiag function, 266
  variable scope, 57–58
Machine epsilon, 110
Machine precision. *See* Machine epsilon
Maclaurin series expansion, 103
Main function, 61
Mantissa, 110
Mathematical model, 2, 5
  analytical solution to bungee jumper problem, 7–10
  numerical solution to bungee jumper problem, 10–12
  real drag, 17–19
Mathematical operations, 36–39
MATLAB
  assignment function, 29–35
  bisect function, 151–152
  built-in functions, 674–675
  bvp4c function, 665–667
  chol function, 285–286
  Cholesky factorization with, 285–286
  cumtrapz functions, 510
  descriptive statistics in, 352–353
  DFT of simple sinusoid with, 421–422
  eig function, 336
  eigenvalues, 336–337
  eigenvectors, 336–337
  environment, 28–29
  eulode, 586–587
  exploratory data analysis, 46–48
  fft function, 420–423
  fminsearch, 395
  fsolve function, 319–320
  fzero function, 181–183
  GaussNaive, 258
  GaussPivot, 262–263
  GaussSeidel, 309
  graphics, 42–45
  to implement Gauss-Seidel method, 310
  integral function, 539
  integral2 and integral3 functions, 514
  Lagrange function, 443–444
  left division, 286
  linear algebraic equations with, 238–240
  linregr function, 371–372
  LU factorization, 282–283
  M-file functions, 676
  mathematical operations, 36–39
  matrix manipulations, 234–237
  multidimensional interpolation in, 475
  Newtint function, 440
  newtraph function, 173–174
  numerical differentiation with, 560–564
  piecewise interpolation in, 468
    interp1 function, 470–473
    spline function, 468–470
  polyfit functions, 373, 433
  polyval functions, 373, 433
  power spectrum with, 424
  programming mode, 3
  quadadapt function, 537–539
  rand function, 354–356
  randn function, 356–358
  resources, 46
  rk4sys, 602–603
  roots function, 184–186
  software environment, 2
  for stiffness, 632–634
  trap, 499–501
  trap function, 499–501
  trapuneq function, 509
  trapz functions, 510
  Tridiag function, 266
  use of built-in functions, 39–42
  variables, 57
  workspace, 57
Matrix, 31–32, 229
  algebra, 225, 229
  condition number, 294–296
    in MATLAB, 296–297
  form, 409
  linear algebraic equations in matrix form, 237–238
  matrix algebra, 229–238
  multiplication, 231–232
  norms, 293–294
  notation, 229–231
  operating rules, 231
    higher-dimensional matrices, 233
    MATLAB matrix manipulations, 234–237
    visual depiction, 232
Matrix inverse, 226, 288
  bungee jumper problem analysis, 291–292
  calculating inverse, 288–289
  error analysis and system condition, 292
    matrix condition number, 294–296

methods, 292–293
vector and matrix norms, 293–294
example, 289–290
indoor air pollution, 297–300
norms and condition number in MATLAB, 296–297
stimulus-response computations, 290–291
max function, 262
Maximum likelihood principle, 363
Mean value, 406
Measurement errors, 131
Median, 348
Michaelis-Menten equation, 373–374
Midpoint method, 592–593, 595
Midtest loop, 75
Minimax criterion, 359
Minimum potential energy, 213–215
Minors, 251
Mixed partial derivative, 560
Mode, 348
Model errors, 130–131
Modified secant method, 175–176
Multidimensional interpolation, 473. *See also* Piecewise interpolation; Polynomial interpolation
bilinear interpolation, 473–475
in MATLAB, 475
Multidimensional optimization, 211–213
Multimodal cases, 202
Multiple integrals, 512
double integral, 512–514
MATLAB functions: integral2 and integral3 functions, 514
Multiple linear regression, 345, 389–391
Multistep methods, 581, 624
error estimation, 627–628
non-self-starting Heun method, 624–627

N

Naive Gauss elimination, 254, 261
back substitution, 256
example, 257
forward elimination of unknowns, 255–256
MATLAB M-file, 258
operation counting, 258–261
Natural cubic splines, 466–467
Nearest neighbor interpolation, 470
Nesting, 79–81
Neumann boundary condition, 654
Newtint function, 440–441
Newton interpolating polynomial, 433. *See also* Lagrange interpolating polynomial
form of, 437–439
linear interpolation, 433–435
Newtint function, 440–441
quadratic interpolation, 435–437
Newton linear-interpolation formula, 433
Newton-Cotes
closed integration formulas, 507
formulas, 486, 492–494, 512, 525, 541–542
integration formulas, 593
open integration formulas, 512
Newton-Raphson bungee jumper problem, 173–174
Newton-Raphson method, 169, 170, 314
MATLAB M-file, 173–174, 318–319
multivariable Taylor series, 314–315
near-zero slope, 171
for nonlinear system, 315–318
poor convergence, 172
slowly converging function, 170–171
Newton’s interpolating polynomial, 345
Newton’s second law, 12, 228, 326, 405, 551, 573
Newton’s viscosity law, 552
newtraph function, 173–174
Non-self-starting Heun method, 624
example, 626–627
fundamental difference, 624, 625
Nongradient methods, 213
Nonhomogeneous system, 328
Nonlinear algebraic equation, 183
Nonlinear ODEs
finite-difference methods for, 663–665
shooting method for, 656–658
Nonlinear regression, 345, 395. *See also* General linear least-squares regression
fitting experimentatal data, 397–399
with MATLAB, 395–396
techniques, 367
Nonlinear relationships, linearization of, 366
comments on linear regression, 370
fitting data with power equation, 368–370
nonlinear regression techniques, 367–368
power equation, 366–367
Nonlinear simultaneous equations, 226
Nonlinear systems, 312. *See also* Linear systems
Newton-Raphson method, 314–319
simultaneous nonlinear equations, 312–313
successive substitution, 313–314
Nonperiodic function, 416
Nonstiff systems, MATLAB functions for, 617
MATLAB to solving system of ODEs, 618–620
ode23 function, 617
odeset to control integration options, 620–621
predator-prey model, 619, 620
Norm, 293
in MATLAB, 296–297
Normal distribution, 351–352
Normalization, 256
“Not-a-knot” end condition, 467
Number system, 106
Numerical differentiation, 16, 121, 487, 549–551
backward difference approximation of first derivative, 123
centered difference approximation of first derivative, 123
derivatives
and integrals for data with errors, 558–559
partial, 559–560
of unequally spaced data, 557–558
diff function, 560–562
in engineering and science, 551–552
error analysis, 126–129
finite-difference approximations of derivatives, 124–125
of higher derivatives, 125

Numerical differentiation (*Continued*)
- free-falling bungee jumper, 548
- gradient function, 563–564
- high-accuracy differentiation formulas, 552–555
- with MATLAB, 560
- Richardson extrapolation, 555–557
- roundoff errors in, 127–129
- truncation errors in, 127–129
- visualizations of vector fields, 565–567

Numerical errors, 101
- control, 129–130

Numerical integration, 16
- formulas
  - computing work with, 515–518
  - free-falling bungee jumper, 488–489
  - higher-order Newton-Cotes formulas, 507–508
  - multiple integrals, 512–514
  - Newton-Cotes formulas, 492–494
  - open methods, 512
  - Simpson's rules, 501–507
  - trapezoidal rule, 494–501
- of functions
  - adaptive quadrature, 537–540
  - Gauss quadrature, 530–537
  - Romberg integration, 525–530
  - root-mean-square current, 540–543

Numerical methods, 1, 2, 9, 13, 104, 136. *See also* Bisection method
- conservation laws in engineering and science, 12–13
- devices and types of balances, 14
- real drag, 17–19

Nyquist frequency, 419

O

Octal, 107
ODE. *See* Ordinary differential equations (ODE)
ode113 function, 618
ode15s function, 632
ode23 function, 617, 620–621
ode23s function, 632
ode23t function, 632
ode23tb function, 632
ode45 function, 618, 620, 637, 655
Ohm's law, 241, 541, 552
One-dimensional optimization, 202–211
- fminbnd function, 210–211
- golden-section search, 203–208
- parabolic interpolation, 209–210

One-point iteration. *See* Simple fixed-point iteration
One-step methods, 581
Open integration formulas, 487, 494
Open methods, 143, 164
Operation counting, 258–261
Optimization, 16, 136, 199
- elevation as function of time, 199
- equilibrium and minimum potential energy, 213–215
- mathematical perspective, 199
- multidimensional, 202, 211–213
- one-dimensional, 202–211
- optimum analytically by root location, 200–201
- single variable function, 200

Optimum analytically by root location, 200–201
optimset function, 182–183
Ordinary differential equations (ODE), 16, 573, 574, 647
- dependent variable, 573–574
- first-order ODEs, 574, 576
- initial-value problem, 577
- Runge-Kutta techniques, 577
- shooting method
  - for linear ODE, 652–654
  - for nonlinear ODEs, 656–658
- solution, 575

Ordinary frequency, 407
Oscillations, 447–449
Overdetermined systems, 238
Overflow error, 109
Overrelaxation, 309

P

Parabola. *See* Second-order polynomial
Parabolic interpolation, 209–210
Parameters, 5, 6
- passing, 85–87

Partial derivatives, 550, 559–560
Partial differential equation (PDE), 574. *See also* Ordinary differential equations (ODE)
Partial pivoting, 261
Passed function, 82
Passing functions
- anonymous functions, 81–82
- building and implementing function function, 84–85
- function functions, 82–83
- to M-files, 81

Passing parameters, 85
- approach for, 86–87

Pause command, 76
pchip. *See* Piecewise cubic Hermite interpolation (pchip)
PDE. *See* Partial differential equation (PDE)
Per-step truncation error estimation, 628
Percent relative error, 102
Periodic function, 405
Permutation matrix, 233
Phase angle, 407, 407
Phase shift, 407
Phase-plane plots, 605, 609
Phasor, 413
Piecewise cubic Hermite interpolation (pchip), 468, 471
Piecewise cubic spline interpolation, 470
Piecewise interpolation. *See also* Multidimensional interpolation; Polynomial interpolation
- heat transfer, 476–480
- interp1 function, 470–473
- in MATLAB, 468
- spline function, 468–470

Pipe friction, 186–190
Pivot equation, 256
Pivoting, 261–264
- determinant evaluation with gauss elimination, 263–264
- *LU* factorization with, 280–282
- MATLAB M-file, 262–263

Pliny's intermittent fountain, 635–639
plot3 function, 608
Point-slope method. *See* Euler's method
polyfit functions, 433
- polynomial regression implemention with, 394

Polynomial interpolation, 430. *See also* Multidimensional interpolation
- examples, 431
- extrapolation, 445–447

inverse interpolation, 444–445
Lagrange interpolating polynomial, 441–444
Newton interpolating polynomial, 433–441
oscillations, 447–449
Polynomial(s), 183–186
MATLAB function, 184–186
method, 328, 329–331
regression, 345, 385–389
implementation, 394
with MATLAB, 392–393
polyval functions, 433
Positional notation, 107
Posttest loop, 76
Potential energy, 213
Power equation, 366
fitting data with, 368–370
Power method, 333–334
for highest eigenvalue, 334–336
Power spectrum, 423
with MATLAB, 424
Preallocation of memory, 74–75
Precision, 100–101, 111–112
Predator-prey
equations, 618
models and chaos, 604–609
Predictor equation, 588
Predictor-corrector approach, 589
Pretest loop, 75
Primary function, 61
Problem solving, 1
real drag, 17–19
Programming with MATLAB
bungee jumper velocity, 87–90
input-output, 61–65
M-files, 54
nesting and indentation, 79–81
passing functions to M-files, 81–87
structured programming, 65–78
Propagated truncation error, 583–584
Proportionality, 291

Q

QR factorization, 394
quadadapt function, 537–539
Quadratic
convergence, 170
interpolation, 435–437
polynomial. *See* Second-order polynomial
splines, 459–462
Quadrature, 489

R

Ralston's method, 595
Random numbers, 353
generation, 345
rand function, 354–356
randn function, 356–358
Range, 111, 349
Rate equations, 573
Regression, 16
Relative tolerance (RelTol), 618
Relaxation, 309–310
Gauss-Seidel method with, 311–312
RelTol. *See* Relative tolerance (RelTol)
Residual, 358
Reverse-wrap-around order, 421, 423
Reynolds number, 17
Richardson extrapolation, 524, 525–526, 555–557
example, 526–527
higher-order corrections, 527
RK methods. *See* Runge-Kutta methods (RK methods)
RK-Fehlberg methods, 617
rk4sys function, 602–603, 605
Romberg integration, 487, 508, 524, 525, 528. *See also* Adaptive quadrature; Gauss quadrature
algorithm, 527–530
M-file to implement, 530
Richardson extrapolation, 525–527
Roots, 135, 164, 165, 184–186
bisection method, 146–152
Brent's method, 176–181
in engineering and science, 139–140
false position, 152–155
graphical methods, 140–141
greenhouse gases and rainwater, 156–159
MATLAB function, 181–183
Newton-Raphson method, 169–174
pipe friction, 186–190
polynomials, 183–186
root-location techniques, 224
root-mean-square current, 540–543
Secant methods, 174–176
simple fixed-point iteration, 165–169
roots function, 184–186
round function, 40
Roundoff errors, 3, 106, 583. *See also* Truncation errors
arithmetic manipulations of computer numbers, 112–114
computer number representation, 106–112
in numerical differentiation, 127–129
Row, 229
row-sum norm, 294
vectors, 229
Runge-Kutta methods (RK methods), 577, 578, 581, 587, 593. *See also* Adaptive Runge-Kutta methods; Euler's method
classical fourth-order, 595–598
increment function, 593
rk4sys function, 602–603
second-order, 594–595
systems of equations, 600–601, 603–604
types of, 594
Runge's function, 447, 468
comparison, 448–449, 469–470

S

S-shape, 374
Saturation-growth-rate equation, 367
Scalars, 29–30
Script files, 54–55
Secant methods, 174–176. *See also* Brent's method
Second finite divided difference, 437
Second forward finite difference, 125
Second-order equation, 574
Second-order polynomial, 435
Second-order Runge-Kutta methods, 594–595
Sequential search, 458
Shooting method, 651
with derivative boundary conditions, 654–656
for linear ODE, 652–654
for nonlinear ODEs, 656–658
trial-and-error approach, 651
Sigmoid, 374

Signed magnitude method, 107
Signum function, 580
Simple fixed-point iteration, 165–169, 313–314
Simpson's rules, 501, 529
 Simpson's 1/3 rule, 486, 501–502, 535, 541
  composite Simpson's 1/3 rule, 503–505
  single application, 502–503
 Simpson's 3/8 rule, 486, 505–507, 535
Simulation, 353–358
 MATLAB function, 353–358
Simulink®, 677–684
Simultaneous equations, determining polynomial coefficients with, 431–432
single-line if structure, 66
Singular value decomposition, 394
Sinusoid, 405
 least-squares fit of, 408–411
 sinusoidal functions, 405
  curve fitting with, 405–411
Smearing, 114
SOR. *See* Successive overrelaxation (SOR)
Spectral norm, 294
Splines, 453
 cubic, 462–468
 drafting technique, 455
 functions, 453
 heat transfer, 476–480
 to higher-order interpolating polynomials, 454
 interpolation, 345
 linear, 455–459
 in MATLAB, 468–470
 quadratic, 459–462
 spline function, 468–470
Square matrices, 230
Standard deviation, 349
Steady-state calculation, 12
Stefan-Boltzmann law, 59
Step halving approach, 616
Stiff system, 628
Stiffness, 628
 Euler's method, 629–630
 explicit and implicit Euler method, 630–632
 MATLAB application, 634–635
 MATLAB functions for, 632–634
 Pliny's intermittent fountain, 635–639
 stiff solution of single ODE, 629
Stimulus-response computations, 290–291
Stokes drag, 17
Stopping criterion, 103
String functions, 35
Structured programming, 65
 animation, 77–78
 decisions, 65–72
 loops, 72–76
Subfunctions, 60–61
Subtractive cancellation, 113
Successive overrelaxation (SOR), 309
Successive substitution. *See* Simple fixed-point iteration
sum function, 40
Sunspots, 425–426
Superposition, 291
Swamee-Jain equation, 187, 189
Switch structure, 70–71
Symmetric matrix, 230
Systems of equations, 598
 Euler's method, 598–600
 Runge-Kutta methods, 600–604

T
Table lookup, 458–459
Taylor series, 114–118, 552–553
 to estimating truncation errors, 120–121
 expansion, 118, 314, 584
  remainder for, 119–120
Taylor theorem, 114, 117
Terminal velocity, 9
Thermocline, 476
Third-order polynomial, 462
Three-point Gauss-Legendre formula, 536–537
Time domain, 414–415
Time plane, 415
Time series, 405
Time-consuming evaluation, 208
Time-variable, 12
Top-down design, 79
Torricelli's law, 635–636
Total numerical error, 125
 control of numerical errors, 129–130
 error analysis of numerical differentiation, 126–129
 trade-off between roundoff and truncation error, 126
Transient variable, 12
Transpose, 233, 234, 283
Transposition matrix, 233
trap function, 499–501
Trapezoidal rule, 486, 494, 495, 529, 531, 541, 592, 593
 composite trapezoidal rule, 497–499
 error of, 495
 integral evaluation by, 532
 MATLAB M-file function, trap, 499–501
 single application of, 495–497
 with unequal segments, 508
trapuneq function, 509
trapz functions, 510
Trend analysis, 344
Trial and error technique, 135, 651
Tridiag function, 266
Tridiagonal matrices, 225, 231
Tridiagonal systems, 264
 MATLAB M-file, 266
 solution of, 265
Truncation errors, 3, 114, 583–584. *See also* Roundoff errors
 numerical differentiation, 121–125
 in numerical differentiation, 127–129
 remainder for Taylor series expansion, 119–120
 Taylor series, 114–118
 Taylor series to estimating, 120–121
Two-dimensional interpolation, 473
 implementing by first applying one-dimensional linear interpolation, 474
Two-point Gauss-Legendre formula, 533–536, 543
 weighting factors and function arguments, 536
2s complement technique, 108

U
Uncertainty. *See* Imprecision
Unconditionally stable approach, 630

Underdetermined systems, 238
Underrelaxation, 309
Undetermined coefficients method,
  531–533
Unequally spaced data, derivatives of,
  557–558
Unimodal, 204
Upper triangular matrix, 230, 280
Upper triangular system, 256

V
van der Pol equation, 632–633
Vandermonde matrices, 432
Variable argument list, 71–72
Variable scope, 57–58
Variance, 349
Vectorization, 74
Vectors, 31–32, 293–294
  vector fields, visualizations of,
    565–567
  vector-matrix multiplication, 38
Visualization
  two-dimensional function, 212
  vector fields, 565–567
Voltage rules, 240, 241
Voltages in circuits, 240–243
Volume integral, 492

W
while structure, 75
while... break structure, 75–76
Wolf sunspot number, 425
Word, 106

X
xmin function, 213

Z
Zeros
  of equation, 135
  zero-order approximation, 115