



Symbols

./ (array division) 54, 74
.^ (array exponentiation) 54, 74
.* (array multiplication) 51, 74
= (assignment operator) 21
\ (backslash) 126, 271
: (colon operator) 54, 56, 74
. (dot) 154, 160
... (ellipses) 35, 136, 149
[] (empty vector) 50
== (equal to) 54, 74
> (greater than) 54, 74
>= (greater than or equal to) 54, 74
< (less than) 54, 74
<= (less than or equal to) 54, 74
- (minus sign) 52
~= (not equal to) 54, 74
2-D plots/plotting 234–239
 enhancement tools 237
 parametric 238–239
 simple plots 235–237
3-D plots/plotting 239–243
 linear 239–241
 parametric plots 241–242
& (element-wise AND) 53
| (element-wise OR) 53
/ (matrix division). *see* matrix division (/)
^ (matrix exponentiation) 271, 286
* (matrix multiplication) 51, 268–270, 272–273, 286
% (percent sign) 33, 126
; (semicolon) 28, 62
&& (short-circuit AND) 53, 87
|| (short-circuit OR) 53, 87
~ (unary not) 54, 87, 100
_ (underscore character) 22

A

A* algorithm 411–413
 code for 412
A/D (analog-to-digital) device 317
abstraction 18–19, 46, 106, 268
acosd() function 193
activation stack 186–187
actual parameters 109
adjacency matrix 399, 406
 creation of 400
algorithms 19
A* 411–413
 Breadth-First Search 407–408
 bubble sort 373–375, 383
 complex, analyzing 370–371
 Dijkstra’s 408–411
 insertion sort 371–373, 382–383
 measuring cost of 368–371
 merge sort 377–379, 383
 performance analysis of 380–382
 Prim’s 404–406
 quick sort 375–377, 383
 radix sort 379–380, 383
 for sorting data 371–380
all() function 87, 100
alpha() function 255, 261
ALU. *see* Arithmetic and Logic Unit (ALU)
American Standard Code for Information Interchange (ASCII) 122, 129
ampersand (&) 53
analog-to-digital (A/D) device 317
AND
 element-wise (&) 53
 short-circuit (&&) 53, 54, 87
any() function 87, 100
API. *see* Application Programmer Interface (API)

Application Programmer Interface (API) 108
Arithmetic and Logic Unit (ALU) 5
arithmetic operations 213
 with arrays 64
 on character strings 125
 with vectors 51–52
array division (./) 54
array exponentiation (.^) 54
array multiplication (.*) 51
arrays 60–71
 arithmetic operations with 64
 cell (*see* cell arrays)
 of character strings 131–132
 concatenation 66–67
 creating 62
 elements of 61, 62–64
 inserting data into 214
 library functions with 65–66
 linear 46
 linearized 67–71
 logical operations with 64–65
 matrices *vs.*, 60
 operations 64–71
 properties of 61
 reshaping 67
 slicing of 67
 structure 150–156 (*see also* structure arrays)
 transpose of 61
ASCII. *see* American Standard Code for Information Interchange (ASCII)
assignment operator (=) 21
.au files 317
auread() function 317, 334
auwrite() function 334
auxiliary (local) functions 111
axis() function 232, 260

I-2 Index

B

Babbage, Charles 3
back dividing 271, 273, 286
backslash (\) 126, 271
backward difference
 approximation 356
bar() function 239, 260
bar3() function 242, 261
barh() function 239, 260
barh3() function 243, 261
Basic Input/Output System (BIOS) 7, 9
before() function 215
behavioral abstraction 268
BFS. *see* Breadth-First Search (BFS)
big o algebra 368–371
 O(1) (independent of N) 369
 O(logN) (binary search) 369–370
 O(2^N) (exponential growth) 370
 O(N) (linear with N) 369
 O(N²) (proportional to N²) 370
binary files 168
binary search (O(logN)) 369–370
BIOS. *see* Basic Input/Output System (BIOS)
bits 6
black box view, functions 106–107
bodies of rotation 250–255
 continuous functions, rotating 251–253
 discrete functions, rotating 253–255
boolean value 50, 82, 84, 86–87, 100
Breadth-First Search (BFS) 407–408
break points 36
break statement 94, 96, 97, 100
bubble sort 373–375, 383
building (operation) 213, 216

C

C (programming language) 12
cache memory 7
CAD. *see* computer-aided design (CAD)
case keyword 88, 89, 100
casting 122–123
catch keyword 192, 193, 194, 207
CAToString() function 394
ceil() function 55, 74

cell arrays 142–146
 accessing 143–145
 conversion to string 394
 creating 142–143
 extracting/sorting 382
 inserting data into 214
 processing 145–146
 using 145
central difference
 approximation 357
Central Processing Unit (CPU) 5, 7, 186
char() function 123, 125, 132, 136
character generators 121
character mapping 122
character strings 121–135
 arithmetic operation on 125
 arrays of 131–132
 casting 122–123
 comparison of 129–131
 concatenation of 124
 conversion from numbers to 125–127
 conversion to numbers 127–129
 and delimiter 123
 example using 132–135
 format control strings 126
 logical operation on 125
 mapping 122
 MATLAB implementation 123–125
 as numerical values 122
 operations 129–131
 slicing of 124
 and token 123
class() function 145, 160
classes 24
clc command 26, 33
clear command 33
clf command 232, 260
close all command 232, 260
code blocks 81, 82
coef() function 347–348
colon operator (:) 56
color mapped images 294
color masking 296–301
colormap() function 232, 260
Colossus 3–4
column vector 61
Command History window 26–27
Command window 25–26, 92, 107, 112, 122
comments 33

compile-time errors 12
compilers 11–12
compound surfaces, assembling 256
computer
 hardware (*see* hardware, computer)
 internal details 6
 internal organization of 5
 memory (*see* memory, computer)
 software (*see* software, computer)
computer-aided design (CAD) 10
computer architectures, history of 3–5
computer languages 2–3, 10–11
concatenation
 of arrays 66–67
 of character strings 124
 of sounds 318–320
 of vectors 55
conditional execution 82–83
continue statement 96, 100
continuous function, rotating 251–253
contour() function 255, 261
Control Unit 5
CPU. *see* Central Processing Unit (CPU)
cross() function 60, 74
csvread() function 170, 173, 177, 179
csvwrite() function 170, 179
cubic spline interpolation 343–344
cumsum() function 353, 354, 361
cumtrapz() function 354, 361
Current Directory window 30–31, 34–35, 107, 108
curve fitting 345–351
 example of 349–351
 linear regression 345–347
 polynomial regression 347–349
cycles, graphs 396
cylinder, construction of 248–249
cylinder() function 107–109, 255, 261

D

D/A (digital-to-analog) device 317
data abstraction 19, 46, 268
data bus 6
data collection. *see also* problem-solving
 building 216
 filtering 217–218

- inserting data into 213–215

mapping 216–217

searching 219–220

sorting 220

summarizing 218–219

traversing 215–216

data typing 22–24

deal() function 143–145, 160

debugging 36

del12() function 248

delimited text files 169, 172–173

delimiter 123, 168

dequeue() function 390, 391

derivative, of function 356

design templates 83–84

for functions in MATLAB, 107

for if statement 84

for loop 91

for switch statement 88

for while loop 94–95

diag() function 62, 74

diagonal array 61

diff() function 357, 361

difference engine, Babbage 3

differentiation 356–357

digital-to-analog (D/A) device 317

Dijkstra’s algorithm 408–411

code for 410

directional edges 396

discrete functions, rotating 253–255

disp() function 58, 74, 129, 136

division

matrix 271, 273–274

dlmread() function 170, 172–173, 179

dlmwrite() function 170, 173, 179

documentation section 107

dot (.) notation 154, 160

dot operator 64

double() function 123, 136

drivers 7

E

edges, graphs 396

Editor window 32–33, 36

element-wise AND (&) 53, 54

element-wise OR (|) 53, 54

elements

arrays 61, 62–64

vectors 47

ellipses (. . .) 35, 136, 149

ellipsoid() function 255, 261

else keyword 83, 100

elseif keyword 84, 100

empty vector ([]) 50

encapsulation 106, 111–112

end keyword 193

end statement 83, 84, 89, 92, 100

endless recursion 188

engineering applications

ceramic composition 283–285

detecting edges 306–309

electrical circuit analysis 285–286

encryption 132–135

forces and moments 58–60

geographic data,

visualizing 256–259

geopolitical data,

processing 221–226

graphs 415

liquid levels, computation of 97–99

music synthesizer 332–334

physical structure,

assembling 156–160

robot arm motion 202–206

soil volume, computation of 71–73

solid object measurement 113–114

sorting 384–386

spacecraft launch 36–39

spreadsheet data 177–179

synthesizer notes, shaping 359–360

enqueue() function 390, 391, 393

equal to (= =) 54, 74

error() function 193, 207

Excel spreadsheets 170–172

exceptions 190–194

generic implementation for 191–192

historical approach 191

MATLAB implementation 193–194

execution errors 12

exponential growth ($O(2^N)$) 370

extrapolation 344–345

eye() function 270, 286

F

fact() function 195

false values 50, 74, 83, 84, 86, 100

Fast Fourier Transform (FFT)

324–328

implementation 326–327

overview 325–326

simple spectral analysis

using 327–328

fclose() function 174, 180

FFT. *see* Fast Fourier Transform (FFT)

fft() function 327, 334

fgetl() function 174, 180

fgets() function 174, 180

fib() function 199

Fibonacci series 198–199

.field operator 152

fieldnames() function 147, 153, 154

figure() function 232, 260

Figure window 31–32

files

binary 168

delimited text 169

opening/closing 174

reading/writing 170

text (*see* text files)

fill() function 239, 260

filtering (operation) 213, 217–218

find() function 68, 74

fix() function 55, 74

floor() function 55, 74

flowcharts 83

folding (operation) 213

fopen() function 174, 180

for loop 90–94, 100

breaking out of 94

example of 92

indexing implementation

using 93–94

MATLAB implementation 91–92

structure of 91

template 91

while loop *vs.*, 90

formal parameters 109

format control strings 126

forward difference

approximation 356

fprintf() function 92, 129, 130,

136, 176, 180

frame, stack 187

frequency, sound 322–324

function name section 107

functional programming 20

function(s)

acosd() 193

all() 87, 100

alpha() 255

any() 87, 100

auread() 317, 334

I-4

Index

function(s) <i>(continued)</i>		
axis()	232, 260	
bar()	239, 260	
bar3()	242	
barh()	239, 260	
barh3()	243	
before()	215	
black box view of	106–107	
CAToString()	394	
ceil()	55	
char()	123, 125, 132, 136	
class()	145, 160	
coef()	347–348	
colormap()	232, 260	
contour()	255	
cross()	60, 74	
csvread()	170, 173, 177, 179	
csvwrite()	170, 179	
cumsum()	353, 354, 361	
cumtrapz()	354, 361	
cylinder()	107–109, 255	
deal()	143–145, 160	
defined	106, 107–108	
del2()	248	
dequeue()	390, 391	
derivative of	356	
diag()	62	
diff()	357, 361	
disp()	58, 129, 136	
dlmread()	170, 172–173, 179	
dlmwrite()	170, 173, 179	
double()	123, 136	
ellipsoid()	255	
enqueue()	390, 391, 393	
error()	193, 207	
eye()	270, 286	
fact()	195	
fclose()	174	
fft()	327, 334	
fgetl()	174, 180	
fgets()	174, 180	
fib()	199	
fieldnames()	147, 153, 154	
figure()	232, 260	
fill()	239, 260	
find()	68	
fix()	55	
floor()	55	
fopen()	174, 180	
fprintf()	92, 129, 130, 136, 176, 180	
getfield()	155	
gplot()	405	
grAdjacency()	399	
grid off()	232, 260	
grid on()	232, 260	
gt()	373, 376	
hist()	239, 260	
hold off()	233, 260	
hold on()	233, 261	
ifft()	327, 334	
image()	295, 310	
imread()	295, 310	
imshow()	295, 310	
imwrite()	295, 310	
input()	89, 90, 96, 112, 127–128, 136, 192	
instances	187	
integral of	351	
interp1()	341, 344, 361	
interp2()	343, 361	
interp3()	343, 361	
int2str()	125, 136	
inv()	271, 286	
isa()	146, 161	
is_before()	391–392, 393	
iscell()	146, 161	
ischar()	125, 136, 146, 161	
isempty()	390	
isfield()	155, 161	
islogical()	146, 161	
isnumeric()	146, 161	
isPal()	197	
isspace()	125, 136	
isstruct()	146, 161	
it()	372, 376	
largest()	145	
lasterror()	193, 207	
legend()	233, 261	
length()	48, 58, 61	
lightangle()	248	
linspace()	47, 74, 310	
load()	180	
loglog()	237, 261	
magic()	62, 75	
MATLAB implementation (<i>see</i> functions, in MATLAB)		
max()	66, 92	
mean()	55, 66	
mesh()	243	
meshc()	255	
meshgrid()	243, 245, 255	
meshz()	255	
min()	66	
nargin()	110	
nargout()	110	
num2str()	125, 136	
ones()	47, 62, 305	
peek()	390	
pie()	239, 261	
pie3()	243	
plot()	232, 235, 239, 261	
plot3()	239	
plotyy()	237	
polar()	239, 261	
polyfit()	347–348, 361, 384, 385	
polyval()	348, 361	
rand()	47, 62	
randn()	47	
read()	173	
readStruct()	179	
reshape()	67, 279, 286	
rmfield()	147, 153, 155	
rot90()	310	
round()	55	
save()	180	
semilogx()	237, 261	
semilogy()	237, 261	
setfield()	155, 161	
shading()	233, 261	
size()	48, 58, 61, 146	
sort()	155, 161, 382	
sound()	318, 335	
sphere()	255	
spline()	344, 361	
sprintf()	126, 129, 130, 136, 348	
sscanf()	127, 129, 136	
strcmp()	130, 131, 136	
strcmpi()	131, 136	
str2num()	127, 128, 129, 136	
strtok()	129, 180	
struct()	149, 150, 161	
subplot()	233–234, 261	
sum()	55, 66, 155	
surf()	243, 244	
surfc()	247, 255	
surfz()	255	
text()	233, 261	
textscan()	175, 180	
title()	233, 261	
toString()	394	
tril()	310	
uint8/16()	123, 136, 310	
view()	247, 261	
waterfall()	255	
wavread()	317, 335	
wavwrite()	335	
xlabel()	233, 261	
xlsread()	170, 177, 180	
xlswrite()	170, 172, 180	
ylabel()	233, 261	
zeros()	47, 62	
zeros of	199–202	
zlabel()	233, 261	
functions, in MATLAB, 46, 107–114		
auxiliary (local)	111	
calling	109	
defined	107–108	
encapsulation in	111–112	
and global variables	112–113	
returning multiple results		
from	110–111	
storing/using	109	

structures 148–150	I	interpreted code 13
template of 107	I/O. <i>see</i> Input/Output (I/O)	<code>int2str()</code> function 125, 136
G	identity matrix 270	<code>inv()</code> function 271, 273, 274, 286
Gaussian Elimination 271	<code>if</code> statements 83–88, 100, 131	<code>isa()</code> function 146, 161
generations, of computer	example 85	<code>is_before()</code> function 391–392, 393
language 10–11	in logical expressions 86–87	<code>iscell()</code> function 146, 161
<code>getfield()</code> function 155	MATLAB implementation 84–86	<code>ischar()</code> function 125, 136, 146, 161
global keyword 112, 207	script with 86	<code>isempty()</code> function 390
Global Scope 112	short-circuit evaluation 87–88	<code>isfield()</code> function 155, 161
global variables 112–113, 115	template for 84	<code>islogical()</code> function 146, 161
<code>gplot()</code> function 405	<code>ifft()</code> function 327, 334	<code>isnumeric()</code> function 146, 161
<code>grAdjacency()</code> function 399	<code>image()</code> function 295, 310	<code>isPal()</code> function 197
graphs 396–404	images 291–309	<code>isspace()</code> function 125, 136
A* algorithm 411–413	color mapped 294	<code>isstruct()</code> function 146, 161
Breadth-First Search 407–408	color masking with 296–301	<code>it()</code> function 372, 376
building 398–401	displaying 295	iteration 90
creating 31, 32	format of 294–295	J
cycles 396	gray scale 293	Joint Photographic Experts Group
defined 389	kaleidoscope, creation of 301–303	(JPEG) 294, 301
Dijkstra’s algorithm 408–411	nature of 292	K
examples 396–397, 415	operation on 295–306	kaleidoscope, creation of 301–303
minimum spanning trees of (<i>see</i>	reading 295	L
minimum spanning trees (MSTs))	resolution of 292	<code>largest()</code> function 145
nodes 389	stretching/shrinking 295–296	<code>lasterror()</code> function 193, 207
paths on 396, 406–414	on surface 303–306	least squares technique 346
processing 397–398	true color 293	legalist approach 195
searching 403–404	types 293–295	<code>legend()</code> function 233, 261
traversal 401–403	writing 295	<code>length()</code> function 48, 58, 61, 74
weighted 396, 398	<code>imread()</code> function 295, 310	less than (<) 54, 74
gray scale images 293	<code>imshow()</code> function 295, 310	less than or equal to (<=) 54, 74
greater than (>) 54, 74	<code>imwrite()</code> function 295, 310	library functions
greater than or equal to (>=) 54, 74	in-line coding 195	with arrays 65–66
<code>grid off()</code> function 232, 260	inner dimensions 269	with vectors 54, 55
<code>grid on()</code> function 232, 260	<code>input()</code> function 89, 90, 96, 100,	<code>lightangle()</code> function 248, 261
<code>gt()</code> function 373, 376	112, 127–128, 136, 192	linear arrays 46
H	Input/Output (I/O) 6, 168–179	linear equations, simultaneous
hardware, computer 5–6	devices 5, 7	281–283
interaction with software 8	high-level 169–173 (<i>see also</i> high-	linear interpolation 340–343
hardwiring 6	level I/O functions)	linear matrices 47
heap 8	lower-level 174–177 (<i>see also</i>	linear regression 345–347
help command 108, 115	lower-level I/O functions)	linearized array 67–71
helper functions 111	and MATLAB workspace 168–169	line(s)
heterogeneous collections 142	inserting data, in collection 213–215	intersecting 282–283
high-level I/O functions 169–173	template for 215	rotating 275–276
with delimited text files 172–173	insertion sort 371–373, 382–383	linker 12
with Excel spreadsheets 170–172	integral, of function 351	<code>linspace()</code> function 47, 74, 310
exploration 169–170	integration 351–355	
<code>hist()</code> function 239, 260	<code>interp1()</code> function 341, 344, 361	
<code>hold off()</code> function 233, 260	<code>interp2()</code> function 343, 361	
<code>hold on()</code> function 233, 261	<code>interp3()</code> function 343, 361	
homogeneous collections 46	interpolation 340–345	
	cubic spline 343–344	
	extrapolation 344–345	
	linear 340–343	

I-6 Index

- `load()` function 180
- loader 12
- Local Scope 112
- logic errors 12, 23
- logical expressions 86–87
- logical indexing 50
- logical operations
 - with arrays 64–65
 - on character strings 125
 - with vectors 52–54
- logical value 50
- `loglog()` function 237, 261
- loop-and-a-half iteration style 96–97
- lower-level I/O functions 174–177
 - opening/closing files 174
- M
- `magic()` function 62, 75
- mapping
 - character 122
 - operation 213, 216–217
- mass memory 7
- MATLAB
 - advantages 18
 - components of 18
 - and data manipulation 20–24
 - introduction to 13–14, 17–18
 - and problem-solving 14–15
 - programming concepts 14
 - starting/stopping 20–21
 - user interface 24–33 (*see also* user interface)
- matrix(-ces) 267–286
 - adjacency 399, 400, 406
 - arrays *vs.*, 60
 - examples using 283–286
 - identity 270
 - implementation 271–274
 - linear 47
 - operations on 268–274
 - rotating coordinates 274–281
 - sparse 399, 400
- matrix division (/) 271, 273–274, 286
 - for solving simultaneous linear equations 281–283
- matrix exponentiation (^) 271, 286
- matrix multiplication (*) 51, 268–270, 272–273, 286
 - for 2-D rotation 274–278
 - for 3-D rotation 278–281
- `max()` function 66, 75, 92
- `mean()` function 55, 66, 75
- mechanical memory 6–7
- memory, computer 6–8
 - layout 8
- Mercator projection 303
- merge sort 377–379, 383
- `mesh()` function 243, 261
- `meshc()` function 255, 262
- `meshgrid()` function 243, 245, 255, 262
- `meshz()` function 255, 262
- `min()` function 66, 75
- minimum spanning trees (MSTs) 404–406
- minus, unary (–) 52
- multiplication
 - array 51
 - matrix. *see* matrix multiplication (*)
- music synthesizer 332–334
- musical sounds 321–324
 - about 321
 - changing frequency of 322–324
- N
- `NaN` keyword 344, 361, 392
- `nargin()` function 110, 115
- `nargout()` function 110, 115
- Newton’s method 202
- nodes, graphs 389
- not equal to (~=) 54
- numbers
 - conversion, to strings 125–127
 - conversion from strings to 127–129
- numerical indexing 49–50
- numerical methods 339–360
 - analytical operations 357
 - curve fitting 345–351 (*see also* curve fitting)
 - differentiation 356–357
 - example using 359–360
 - implementation 357–358
 - integration 351–355
 - interpolation 340–345 (*see also* interpolation)
- numerical values 122
- `num2str()` function 125, 136
- O
- object code 12
- object-oriented programming (OOP) 20
- objects 24
- `ones()` function 47, 62, 75, 305
- OOP. *see* object-oriented programming (OOP)
- operating systems (OS) 7–8, 9
- operation(s)
 - analytical 357
 - on arrays 64–71
 - character string 129–131
 - frequency domain 328–332
 - on graphs 397–398
 - on queues 390
 - summary of 212–220
 - on vectors 51–58
- operators
 - dot 64
 - .field 152
 - logical 53
 - precedence 54
- OR
 - element-wise (|) 53, 54
 - short-circuit (||) 53, 54, 87
- OS. *see* operating systems (OS)
- otherwise keyword 88, 89, 100
- P
- page buffer 7
- palindromes, determination 197–198
- parabolic dish 245–247
- paradigms, programming 20
- parameters
 - cell arrays of 145
 - formal *vs.* actual 109
 - value 126
 - variable numbers of 109–110
- parameters section 107
- parametric plots
 - 2-D 238–239
 - 3-D 241–242
- passing by reference 109
- passing by value 109
- paths, on graphs 396
 - A* algorithm 411–413
 - Breadth-First Search 407–408
 - Dijkstra’s algorithm 408–411
 - searching 406–414
- `pause()` function 323
- `peek()` function 390
- percent sign (%) 33, 126
- `pie()` function 239, 261
- `pie3()` function 243, 262
- pixels 292
- plaid surface 243
- playback 316–317
- `plot()` function 232, 235, 239, 261
- `plot3()` function 239, 262
- plots (plotting) 231–259

2-D 234–239 (<i>see also</i> 2-D plots/ plotting)	readStruct() function 179	of sounds 318–320
3-D 239–243 (<i>see also</i> 3-D plots/ plotting)	recording, sound 316–317	of vectors 56–58
data, manipulation of 256	recursion 185–206	software, computer 8–10
enhancement tools 237	activation stack 186–187	categories of 8
figures as containers for 232	defined 187–188	interaction with hardware 8
functions for enhancement 232–233	endless 188	tools (<i>see</i> software tools)
manually editing 234–235	examples 197–202	software tools 9–10
subplots 233–234	implementation 188–190	solid-state memory 6–7
surface plots 243–256 (<i>see also</i> surface plots)	reshape() function 67, 279, 286	sort() function 155, 161, 382
plotyy() function 237	resolution	sorting 213, 220, 367–386
polar() function 239, 261	of images 292	algorithm for 371–380
polyfit() function 347–348, 361, 384, 385	of recorded data 317	applications 382–383
polynomial regression 347–349	<return info section>, 107	bubble 373–375, 383
polyval() function 348, 361	RGB (red, green, and blue) 292	example using 384–386
Prim’s algorithm 404–406	rmfield() function 147, 153, 155	insertion 371–373, 382–383
priority queues 391–393	ROM. <i>see</i> Read-Only Memory (ROM)	and measuring algorithm
problem-solving 14–15, 211–226. <i>see</i> <i>also</i> data collection	rot90() function 310	cost 368–371
assembling solution steps for 212	rotations	merge 377–379, 383
example 221–226	2-D 275–278	quick 375–377, 383
inserting into collection 213–215	3-D 278–281	radix 379–380, 383
larger problems 220–221	round() function 55, 75	sound() function 318, 319, 323, 335
plan for 212	runtime errors 12	sound(s) 315–334
procedural abstraction 19, 106, 268	S	example using 332–334
procedural programming 20	save() function 180	Fast Fourier Transform 325–328
program bugs 12	scalar vectors 51	(<i>see also</i> Fast Fourier Transform
programming 211	scale, playing a musical 322–323	(FFT))
programming languages 10–11	scripts 33–39	frequency domain
overview of 18–20	creating 33–34	operations 328–332
Q	debugging 36	intensity 316
queue(s) 390–396	example using 36–39	musical 321–324
implementation 390–391	punctuating 35	physics of 316
nature of 390	running 35	recording/playback 316–317
operations on 390	searching (operation) 213, 219–220	slicing/concatenating 318–320
overview 390	semicolon (;) 28, 62	source code 12
priority 391–393	semilogx() function 237, 261	spacecraft launch, example 36–39
testing 393–396	semilogy() function 237, 261	sparse matrix 399, 400
quick sort 375–377, 383	setfield() function 155, 161	sphere, construction of 249–250
R	shading() function 233, 261	sphere() function 255, 262
radix sort 379–380, 383	short-circuit AND (&&) 53, 54, 87	spline() function 344, 361
RAM. <i>see</i> Random-Access Memory (RAM)	short-circuit evaluation 87–88	spreadsheets 170–172
rand() function 47, 62, 75	short-circuit OR () 53, 54, 87	sprintf() function 126, 129, 130, 136, 348
randn() function 47, 62, 75	shortening, of vector 50–51	square array 61
Random-Access Memory (RAM) 7	shrinking images 295–296	sscanf() function 127, 129, 136
read() function 173	Simpson’s rule 351, 353	stack 8, 186–187
Read-Only Memory (ROM) 7	simultaneous linear equations, solving 281–283	strcmp() function 130, 131, 136
	size() function 48, 58, 61, 75, 146	strcmpi() function 131, 136
	slicing	stretching images 295–296
	of arrays 67	strings
	of character strings 124	cell arrays conversion to 394
		character 121–135 (<i>see also</i> character strings)

I-8 Index

strings (<i>continued</i>)	traversing (operation)	library functions with
conversion from numbers	graphs	logical operations with
to	<code>tril()</code> function	operating on
<code>str2num()</code> function	true color images	scalar
strong typing	true values	shortening
<code>strtok()</code> function	try keyword	size of
<code>struct()</code> function	tune, playing	slicing
structure arrays	type, data	vectors of indices
constructing	typographical errors	<code>view()</code> function
elements, accessing	U	virtual memory
inserting data into	<code>uint8/16()</code> function	von Neumann architecture
manipulation	unary minus	
<code>structure(s)</code>	unary not	W
constructing/accessing	underscore character	<code>waterfall()</code> function
functions	untyped languages	<code>.wav</code> files
manipulation	user interface	<code>wavread()</code> function
<code>subplot()</code> function	Command History window	<code>wavwrite()</code> function
<code>sum()</code> function	Command window	weak typing
<code>surf()</code> function	Current Directory window	weighted graph
surface, images on	Editor window	<code>while</code> loop
surface plots	Figure window	breaking
3-D parametric surfaces	Variable Editor window	example
bodies of rotation	Workspace window	loop-and-a-half iteration
<i>also</i> bodies of rotation)	utilities, operating systems	style
compound surfaces, assembly	V	for loop <i>vs.</i> , 90
of	value parameters	MATLAB implementation
cube	value(s)	structure of
functions to create	assigning, to variables	template for
manipulation of	boolean/logical	<code>who</code> command
parabolic dish	parameters	<code>whos</code> command
<code>surf()</code> function	Variable Editor window	workspace, saving
<code>surfz()</code> function	variable scoping	Workspace window
switch statement	variable(s)	wrapper function
MATLAB implementation	assigning values to	template for
template for	global	
synthesizer notes, shaping	names	X
T	vector(s)	<code>xlabel()</code> function
technology, advancement in	arithmetic operations with	<code>xlsread()</code> function
text files	concatenation of	<code>xlswrite()</code> function
delimited	creating	Y
reading	elements	<code>ylabel()</code> function
writing	extracting/sorting	Z
<code>text()</code> function	indexing of	<code>zeros()</code> function
<code>textscan()</code> function	inserting data into	<code>zlabel()</code> function
<code>title()</code> function		
token		
<code>toString()</code> function		
trapezoidal rule		