

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Scripting versus Traditional Programming	1
1.1.1	Why Scripting is Useful in Computational Science	2
1.1.2	Classification of Programming Languages	4
1.1.3	Productive Pairs of Programming Languages	5
1.1.4	Gluings Existing Applications	6
1.1.5	Scripting Yields Shorter Code	7
1.1.6	Efficiency	8
1.1.7	Type-Specification (Declaration) of Variables	9
1.1.8	Flexible Function Interfaces	11
1.1.9	Interactive Computing	12
1.1.10	Creating Code at Run Time	13
1.1.11	Nested Heterogeneous Data Structures	14
1.1.12	GUI Programming	16
1.1.13	Mixed Language Programming	17
1.1.14	When to Choose a Dynamically Typed Language	19
1.1.15	Why Python?	20
1.1.16	Script or Program?	21
1.2	Preparations for Working with This Book	22
<b>2</b>	<b>Getting Started with Python Scripting</b>	<b>27</b>
2.1	A Scientific Hello World Script	27
2.1.1	Executing Python Scripts	28
2.1.2	Dissection of the Scientific Hello World Script	29
2.2	Working with Files and Data	32
2.2.1	Problem Specification	32
2.2.2	The Complete Code	33
2.2.3	Dissection	33
2.2.4	Working with Files in Memory	36
2.2.5	Array Computing	37
2.2.6	Interactive Computing and Debugging	39
2.2.7	Efficiency Measurements	42
2.2.8	Exercises	43
2.3	Gluings Stand-Alone Applications	46
2.3.1	The Simulation Code	47
2.3.2	Using Gnuplot to Visualize Curves	49
2.3.3	Functionality of the Script	50
2.3.4	The Complete Code	51
2.3.5	Dissection	53
2.3.6	Exercises	55
2.4	Conducting Numerical Experiments	58
2.4.1	Wrapping a Loop Around Another Script	59

2.4.2	Generating an HTML Report	60
2.4.3	Making Animations	61
2.4.4	Varying Any Parameter	63
2.5	File Format Conversion	66
2.5.1	A Simple Read/Write Script	66
2.5.2	Storing Data in Dictionaries and Lists	68
2.5.3	Making a Module with Functions	69
2.5.4	Exercises	71
<b>3</b>	<b>Basic Python</b>	<b>73</b>
3.1	Introductory Topics	74
3.1.1	Recommended Python Documentation	74
3.1.2	Control Statements	75
3.1.3	Running Applications	76
3.1.4	File Reading and Writing	78
3.1.5	Output Formatting	79
3.2	Variables of Different Types	81
3.2.1	Boolean Types	81
3.2.2	The None Variable	82
3.2.3	Numbers and Numerical Expressions	82
3.2.4	Lists and Tuples	84
3.2.5	Dictionaries	90
3.2.6	Splitting and Joining Text	94
3.2.7	String Operations	95
3.2.8	Text Processing	96
3.2.9	The Basics of a Python Class	98
3.2.10	Copy and Assignment	100
3.2.11	Determining a Variable's Type	104
3.2.12	Exercises	106
3.3	Functions	110
3.3.1	Keyword Arguments	111
3.3.2	Doc Strings	112
3.3.3	Variable Number of Arguments	112
3.3.4	Call by Reference	114
3.3.5	Treatment of Input and Output Arguments	115
3.3.6	Function Objects	116
3.4	Working with Files and Directories	117
3.4.1	Listing Files in a Directory	118
3.4.2	Testing File Types	118
3.4.3	Removing Files and Directories	119
3.4.4	Copying and Renaming Files	120
3.4.5	Splitting Pathnames	121
3.4.6	Creating and Moving to Directories	122
3.4.7	Traversing Directory Trees	122
3.4.8	Exercises	125

<b>4</b>	<b>Numerical Computing in Python</b> .....	131
4.1	A Quick NumPy Primer .....	132
4.1.1	Creating Arrays .....	132
4.1.2	Array Indexing .....	136
4.1.3	Loops over Arrays .....	138
4.1.4	Array Computations .....	139
4.1.5	More Array Functionality .....	142
4.1.6	Type Testing .....	144
4.1.7	Matrix Objects .....	145
4.1.8	Exercises .....	146
4.2	Vectorized Algorithms .....	147
4.2.1	From Scalar to Array in Function Arguments .....	147
4.2.2	Slicing .....	149
4.2.3	Exercises .....	150
4.3	More Advanced Array Computing .....	151
4.3.1	Random Numbers .....	152
4.3.2	Linear Algebra .....	153
4.3.3	Plotting .....	154
4.3.4	Example: Curve Fitting .....	157
4.3.5	Arrays on Structured Grids .....	159
4.3.6	File I/O with NumPy Arrays .....	163
4.3.7	Functionality in the Numpyutils Module .....	165
4.3.8	Exercises .....	168
4.4	Other Tools for Numerical Computations .....	173
4.4.1	The ScientificPython Package .....	173
4.4.2	The SciPy Package .....	178
4.4.3	The Python–Matlab Interface .....	183
4.4.4	Symbolic Computing in Python .....	184
4.4.5	Some Useful Python Modules .....	186
<b>5</b>	<b>Combining Python with Fortran, C, and C++</b> .....	189
5.1	About Mixed Language Programming .....	189
5.1.1	Applications of Mixed Language Programming .....	190
5.1.2	Calling C from Python .....	190
5.1.3	Automatic Generation of Wrapper Code .....	192
5.2	Scientific Hello World Examples .....	194
5.2.1	Combining Python and Fortran .....	195
5.2.2	Combining Python and C .....	201
5.2.3	Combining Python and C++ Functions .....	208
5.2.4	Combining Python and C++ Classes .....	210
5.2.5	Exercises .....	214
5.3	A Simple Computational Steering Example .....	215
5.3.1	Modified Time Loop for Repeated Simulations .....	216
5.3.2	Creating a Python Interface .....	217
5.3.3	The Steering Python Script .....	218
5.3.4	Equipping the Steering Script with a GUI .....	222
5.4	Scripting Interfaces to Large Libraries .....	223

<b>6</b>	<b>Introduction to GUI Programming</b>	227
6.1	Scientific Hello World GUI	228
6.1.1	Introductory Topics	228
6.1.2	The First Python/Tkinter Encounter	230
6.1.3	Binding Events	233
6.1.4	Changing the Layout	234
6.1.5	The Final Scientific Hello World GUI	238
6.1.6	An Alternative to Tkinter Variables	240
6.1.7	About the Pack Command	241
6.1.8	An Introduction to the Grid Geometry Manager	243
6.1.9	Implementing a GUI as a Class	245
6.1.10	A Simple Graphical Function Evaluator	247
6.1.11	Exercises	248
6.2	Adding GUIs to Scripts	250
6.2.1	A Simulation and Visualization Script with a GUI	250
6.2.2	Improving the Layout	253
6.2.3	Exercises	256
6.3	A List of Common Widget Operations	257
6.3.1	Frame	259
6.3.2	Label	260
6.3.3	Button	262
6.3.4	Text Entry	262
6.3.5	Balloon Help	264
6.3.6	Option Menu	265
6.3.7	Slider	265
6.3.8	Check Button	266
6.3.9	Making a Simple Megawidget	266
6.3.10	Menu Bar	267
6.3.11	List Data	269
6.3.12	Listbox	269
6.3.13	Radio Button	272
6.3.14	Combo Box	274
6.3.15	Message Box	275
6.3.16	User-Defined Dialogs	277
6.3.17	Color-Picker Dialogs	278
6.3.18	File Selection Dialogs	279
6.3.19	Toplevel	280
6.3.20	Some Other Types of Widgets	281
6.3.21	Adapting Widgets to the User's Resize Actions	282
6.3.22	Customizing Fonts and Colors	284
6.3.23	Widget Overview	286
6.3.24	Exercises	289

<b>7</b>	<b>Web Interfaces and CGI Programming</b> .....	295
7.1	Introductory CGI Scripts .....	296
7.1.1	Web Forms and CGI Scripts .....	297
7.1.2	Generating Forms in CGI Scripts .....	299
7.1.3	Debugging CGI Scripts .....	301
7.1.4	A General Shell Script Wrapper for CGI Scripts ....	302
7.1.5	Security Issues .....	304
7.2	Adding Web Interfaces to Scripts .....	306
7.2.1	A Class for Form Parameters .....	306
7.2.2	Calling Other Programs .....	308
7.2.3	Running Simulations .....	309
7.2.4	Getting a CGI Script to Work .....	311
7.2.5	Using Web Applications from Scripts .....	313
7.2.6	Exercises .....	316
<b>8</b>	<b>Advanced Python</b> .....	319
8.1	Miscellaneous Topics .....	319
8.1.1	Parsing Command-Line Arguments .....	319
8.1.2	Platform-Dependent Operations .....	322
8.1.3	Run-Time Generation of Code .....	323
8.1.4	Exercises .....	324
8.2	Regular Expressions and Text Processing .....	326
8.2.1	Motivation .....	326
8.2.2	Special Characters .....	329
8.2.3	Regular Expressions for Real Numbers .....	331
8.2.4	Using Groups to Extract Parts of a Text .....	334
8.2.5	Extracting Interval Limits .....	335
8.2.6	Extracting Multiple Matches .....	339
8.2.7	Splitting Text .....	344
8.2.8	Pattern-Matching Modifiers .....	345
8.2.9	Substitution and Backreferences .....	347
8.2.10	Example: Swapping Arguments in Function Calls ...	348
8.2.11	A General Substitution Script .....	351
8.2.12	Debugging Regular Expressions .....	353
8.2.13	Exercises .....	354
8.3	Tools for Handling Data in Files .....	362
8.3.1	Writing and Reading Python Data Structures .....	362
8.3.2	Pickling Objects .....	364
8.3.3	Shelving Objects .....	366
8.3.4	Writing and Reading Zip and Tar Archive Files ....	366
8.3.5	Downloading Internet Files .....	367
8.3.6	Binary Input/Output .....	368
8.3.7	Exercises .....	371
8.4	A Database for NumPy Arrays .....	371
8.4.1	The Structure of the Database .....	371
8.4.2	Pickling .....	374
8.4.3	Formatted ASCII Storage .....	375

8.4.4	Shelving	376
8.4.5	Comparing the Various Techniques	377
8.5	Scripts Involving Local and Remote Hosts	378
8.5.1	Secure Shell Commands	378
8.5.2	Distributed Simulation and Visualization	380
8.5.3	Client/Server Programming	382
8.5.4	Threads	382
8.6	Classes	384
8.6.1	Class Programming	384
8.6.2	Checking the Class Type	388
8.6.3	Private Data	389
8.6.4	Static Data	390
8.6.5	Special Attributes	390
8.6.6	Special Methods	391
8.6.7	Multiple Inheritance	392
8.6.8	Using a Class as a C-like Structure	393
8.6.9	Attribute Access via String Names	394
8.6.10	New-Style Classes	394
8.6.11	Implementing Get/Set Functions via Properties	395
8.6.12	Subclassing Built-in Types	396
8.6.13	Building Class Interfaces at Run Time	399
8.6.14	Building Flexible Class Interfaces	403
8.6.15	Exercises	409
8.7	Scope of Variables	413
8.7.1	Global, Local, and Class Variables	413
8.7.2	Nested Functions	415
8.7.3	Dictionaries of Variables in Namespaces	416
8.8	Exceptions	418
8.8.1	Handling Exceptions	419
8.8.2	Raising Exceptions	420
8.9	Iterators	421
8.9.1	Constructing an Iterator	421
8.9.2	A Pointwise Grid Iterator	423
8.9.3	A Vectorized Grid Iterator	427
8.9.4	Generators	428
8.9.5	Some Aspects of Generic Programming	432
8.9.6	Exercises	436
8.10	Investigating Efficiency	437
8.10.1	CPU-Time Measurements	437
8.10.2	Profiling Python Scripts	441
8.10.3	Optimization of Python Code	442
8.10.4	Case Study on Numerical Efficiency	445

<b>9</b>	<b>Fortran Programming with NumPy Arrays</b>	451
9.1	Problem Definition	451
9.2	Filling an Array in Fortran	453
9.2.1	The Fortran Subroutine	454
9.2.2	Building and Inspecting the Extension Module	455
9.3	Array Storage Issues	457
9.3.1	Generating an Erroneous Interface	457
9.3.2	Array Storage in C and Fortran	459
9.3.3	Input and Output Arrays as Function Arguments	459
9.3.4	F2PY Interface Files	466
9.3.5	Hiding Work Arrays	470
9.4	Increasing Callback Efficiency	470
9.4.1	Callbacks to Vectorized Python Functions	471
9.4.2	Avoiding Callbacks to Python	473
9.4.3	Compiled Inline Callback Functions	474
9.5	Summary	478
9.6	Exercises	479
<b>10</b>	<b>C and C++ Programming with NumPy Arrays</b>	483
10.1	Automatic Interfacing of C/C++ Code	484
10.1.1	Using F2PY	485
10.1.2	Using Instant	486
10.1.3	Using Weave	487
10.2	C Programming with NumPy Arrays	488
10.2.1	The Basics of the NumPy C API	489
10.2.2	The Handwritten Extension Code	491
10.2.3	Sending Arguments from Python to C	492
10.2.4	Consistency Checks	493
10.2.5	Computing Array Values	494
10.2.6	Returning an Output Array	496
10.2.7	Convenient Macros	497
10.2.8	Module Initialization	499
10.2.9	Extension Module Template	500
10.2.10	Compiling, Linking, and Debugging the Module	502
10.2.11	Writing a Wrapper for a C Function	503
10.3	C++ Programming with NumPy Arrays	506
10.3.1	Wrapping a NumPy Array in a C++ Object	506
10.3.2	Using SCXX	508
10.3.3	NumPy-C++ Class Conversion	511
10.4	Comparison of the Implementations	519
10.4.1	Efficiency	519
10.4.2	Error Handling	523
10.4.3	Summary	524
10.5	Exercises	525

<b>11 More Advanced GUI Programming</b> .....	529
11.1 Adding Plot Areas in GUIs .....	529
11.1.1 The BLT Graph Widget .....	530
11.1.2 Animation of Functions in BLT Graph Widgets .....	536
11.1.3 Other Tools for Making GUIs with Plots .....	538
11.1.4 Exercises .....	539
11.2 Event Bindings .....	541
11.2.1 Binding Events to Functions with Arguments .....	542
11.2.2 A Text Widget with Tailored Keyboard Bindings .....	544
11.2.3 A Fancy List Widget .....	547
11.3 Animated Graphics with Canvas Widgets .....	550
11.3.1 The First Canvas Encounter .....	551
11.3.2 Coordinate Systems .....	552
11.3.3 The Mathematical Model Class .....	556
11.3.4 The Planet Class .....	557
11.3.5 Drawing and Moving Planets .....	559
11.3.6 Dragging Planets to New Positions .....	560
11.3.7 Using Pmw's Scrolled Canvas Widget .....	564
11.4 Simulation and Visualization Scripts .....	566
11.4.1 Restructuring the Script .....	567
11.4.2 Representing a Parameter by a Class .....	569
11.4.3 Improved Command-Line Script .....	583
11.4.4 Improved GUI Script .....	584
11.4.5 Improved CGI Script .....	585
11.4.6 Parameters with Physical Dimensions .....	586
11.4.7 Adding a Curve Plot Area .....	588
11.4.8 Automatic Generation of Scripts .....	589
11.4.9 Applications of the Tools .....	590
11.4.10 Allowing Physical Units in Input Files .....	596
11.4.11 Converting Input Files to GUIs .....	601
<b>12 Tools and Examples</b> .....	605
12.1 Running Series of Computer Experiments .....	605
12.1.1 Multiple Values of Input Parameters .....	606
12.1.2 Implementation Details .....	609
12.1.3 Further Applications .....	614
12.2 Tools for Representing Functions .....	618
12.2.1 Functions Defined by String Formulas .....	618
12.2.2 A Unified Interface to Functions .....	623
12.2.3 Interactive Drawing of Functions .....	629
12.2.4 A Notebook for Selecting Functions .....	633
12.3 Solving Partial Differential Equations .....	640
12.3.1 Numerical Methods for 1D Wave Equations .....	641
12.3.2 Implementations of 1D Wave Equations .....	644
12.3.3 Classes for Solving 1D Wave Equations .....	651
12.3.4 A Problem Solving Environment .....	657
12.3.5 Numerical Methods for 2D Wave Equations .....	663



12.3.6	Implementations of 2D Wave Equations . . . . .	666
12.3.7	Exercises . . . . .	675
<b>A</b>	<b>Setting up the Required Software Environment . . .</b>	<b>677</b>
A.1	Installation on Unix Systems . . . . .	677
A.1.1	A Suggested Directory Structure . . . . .	677
A.1.2	Setting Some Environment Variables . . . . .	678
A.1.3	Installing Tcl/Tk and Additional Modules . . . . .	679
A.1.4	Installing Python . . . . .	680
A.1.5	Installing Python Modules . . . . .	681
A.1.6	Installing Gnuplot . . . . .	683
A.1.7	Installing SWIG . . . . .	684
A.1.8	Summary of Environment Variables . . . . .	684
A.1.9	Testing the Installation of Scripting Utilities . . . . .	685
A.2	Installation on Windows Systems . . . . .	685
<b>B</b>	<b>Elements of Software Engineering . . . . .</b>	<b>689</b>
B.1	Building and Using Modules . . . . .	689
B.1.1	Single-File Modules . . . . .	689
B.1.2	Multi-File Modules . . . . .	693
B.1.3	Debugging and Troubleshooting . . . . .	694
B.2	Tools for Documenting Python Software . . . . .	696
B.2.1	Doc Strings . . . . .	696
B.2.2	Tools for Automatic Documentation . . . . .	698
B.3	Coding Standards . . . . .	702
B.3.1	Style Guide . . . . .	702
B.3.2	Pythonic Programming . . . . .	706
B.4	Verification of Scripts . . . . .	711
B.4.1	Automating Regression Tests . . . . .	711
B.4.2	Implementing a Tool for Regression Tests . . . . .	715
B.4.3	Writing a Test Script . . . . .	719
B.4.4	Verifying Output from Numerical Computations . . . .	720
B.4.5	Automatic Doc String Testing . . . . .	724
B.4.6	Unit Testing . . . . .	726
B.5	Version Control Management . . . . .	728
B.5.1	Mercurial . . . . .	729
B.5.2	Subversion . . . . .	732
B.6	Exercises . . . . .	734
	<b>Bibliography . . . . .</b>	<b>739</b>
	<b>Index . . . . .</b>	<b>741</b>