

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 7 |
| 1.1 | Computational Modelling | 7 |
| 1.1.1 | Introduction | 7 |
| 1.1.2 | Computational Modelling | 7 |
| 1.1.3 | Programming to support computational modelling | 8 |
| 1.2 | Why Python for scientific computing? | 8 |
| 1.2.1 | Optimisation strategies | 10 |
| 1.2.2 | Get it right first, then make it fast | 10 |
| 1.2.3 | Prototyping in Python | 10 |
| 1.3 | Literature | 11 |
| 1.3.1 | Recorded video lectures on Python for beginners | 11 |
| 1.3.2 | Python tutor mailing list | 11 |
| 1.4 | Python version | 11 |
| 1.5 | These documents | 12 |
| 1.5.1 | The <code>%file</code> magic | 12 |
| 1.5.2 | The <code>!</code> to execute shell commands | 13 |
| 1.5.3 | The <code>#NBVAL</code> tags | 13 |
| 1.6 | Your feedback | 13 |
| 2 | A powerful calculator | 13 |
| 2.1 | Python prompt and Read-Eval-Print Loop (REPL) | 13 |
| 2.2 | Calculator | 14 |
| 2.3 | Integer division | 15 |
| 2.3.1 | How to avoid integer division | 15 |
| 2.3.2 | Why should I care about this division problem? | 16 |
| 2.4 | Mathematical functions | 16 |
| 2.5 | Variables | 18 |
| 2.5.1 | Terminology | 20 |
| 2.6 | Impossible equations | 20 |
| 2.6.1 | The <code>+=</code> notation | 21 |
| 3 | Data Types and Data Structures | 21 |
| 3.1 | What type is it? | 21 |
| 3.2 | Numbers | 22 |
| 3.2.1 | Integers | 22 |
| 3.2.2 | Integer limits | 22 |
| 3.2.3 | Floating Point numbers | 23 |
| 3.2.4 | Complex numbers | 23 |
| 3.2.5 | Functions applicable to all types of numbers | 24 |
| 3.3 | Sequences | 24 |
| 3.3.1 | Sequence type 1: String | 24 |
| 3.3.2 | Sequence type 2: List | 26 |
| 3.3.3 | Sequence type 3: Tuples | 29 |
| 3.3.4 | Indexing sequences | 31 |
| 3.3.5 | Slicing sequences | 32 |
| 3.3.6 | Dictionaries | 33 |
| 3.4 | Passing arguments to functions | 36 |
| 3.4.1 | Call by value | 36 |

| | | |
|----------|---|-----------|
| 3.4.2 | Call by reference | 37 |
| 3.4.3 | Argument passing in Python | 38 |
| 3.4.4 | Performance considerations | 39 |
| 3.4.5 | Inadvertent modification of data | 39 |
| 3.4.6 | Copying objects | 40 |
| 3.5 | Equality and Identity/Sameness | 41 |
| 3.5.1 | Equality | 41 |
| 3.5.2 | Identity / Sameness | 42 |
| 3.5.3 | Example: Equality and identity | 42 |
| 4 | Introspection | 44 |
| 4.1 | dir() | 44 |
| 4.1.1 | Magic names | 47 |
| 4.2 | type | 48 |
| 4.3 | isinstance | 48 |
| 4.4 | help | 49 |
| 4.5 | Docstrings | 56 |
| 5 | Input and Output | 57 |
| 5.1 | Printing to standard output (normally the screen) | 57 |
| 5.1.1 | Simple print | 58 |
| 5.1.2 | Formatted printing | 58 |
| 5.1.3 | "str" and "__str__" | 60 |
| 5.1.4 | "repr" and "__repr__" | 61 |
| 5.1.5 | New-style string formatting | 61 |
| 5.1.6 | Changes from Python 2 to Python 3: print | 62 |
| 5.2 | Reading and writing files | 64 |
| 5.2.1 | File reading examples | 64 |
| 6 | Control Flow | 66 |
| 6.1 | Basics | 66 |
| 6.1.1 | Conditionals | 66 |
| 6.2 | If-then-else | 68 |
| 6.3 | For loop | 69 |
| 6.4 | While loop | 70 |
| 6.5 | Relational operators (comparisons) in if and while statements | 70 |
| 6.6 | Exceptions | 71 |
| 6.6.1 | Raising Exceptions | 73 |
| 6.6.2 | Creating our own exceptions | 74 |
| 6.6.3 | LBYL vs EAFP | 74 |
| 7 | Functions and modules | 75 |
| 7.1 | Introduction | 75 |
| 7.2 | Using functions | 75 |
| 7.3 | Defining functions | 76 |
| 7.4 | Default values and optional parameters | 79 |
| 7.5 | Modules | 79 |
| 7.5.1 | Importing modules | 79 |
| 7.5.2 | Creating modules | 80 |
| 7.5.3 | Use of __name__ | 81 |
| 7.5.4 | Example 1 | 81 |

| | | |
|-----------|---|------------|
| 7.5.5 | Example 2 | 82 |
| 8 | Functional tools | 83 |
| 8.1 | Anonymous functions | 83 |
| 8.2 | Map | 85 |
| 8.3 | Filter | 85 |
| 8.4 | List comprehension | 86 |
| 8.5 | Reduce | 87 |
| 8.6 | Why not just use for-loops? | 89 |
| 8.7 | Speed | 90 |
| 8.8 | The %%timeit magic | 92 |
| 9 | Common tasks | 92 |
| 9.1 | Many ways to compute a series | 92 |
| 9.2 | Sorting | 95 |
| 9.2.1 | Efficiency | 97 |
| 10 | From Matlab to Python | 97 |
| 10.1 | Important commands | 97 |
| 10.1.1 | The for-loop | 97 |
| 10.1.2 | The if-then statement | 98 |
| 10.1.3 | Indexing | 98 |
| 10.1.4 | Matrices | 98 |
| 11 | Python shells | 99 |
| 11.1 | IDLE | 99 |
| 11.2 | Python (command line) | 99 |
| 11.3 | Interactive Python (IPython) | 99 |
| 11.3.1 | IPython console | 99 |
| 11.3.2 | Jupyter Notebook | 100 |
| 11.4 | Spyder | 101 |
| 11.5 | Editors | 101 |
| 12 | Symbolic computation | 101 |
| 12.1 | SymPy | 101 |
| 12.1.1 | Output | 102 |
| 12.1.2 | Symbols | 102 |
| 12.1.3 | isympy | 103 |
| 12.1.4 | Numeric types | 104 |
| 12.1.5 | Differentiation and Integration | 105 |
| 12.1.6 | Ordinary differential equations | 109 |
| 12.1.7 | Series expansions and plotting | 111 |
| 12.1.8 | Linear equations and matrix inversion | 113 |
| 12.1.9 | Non linear equations | 115 |
| 12.1.10 | Output: LaTeX interface and pretty-printing | 116 |
| 12.1.11 | Automatic generation of C code | 116 |
| 12.2 | Related tools | 117 |

| | |
|--|------------|
| 13 Numerical Computation | 117 |
| 13.1 Numbers and numbers | 117 |
| 13.1.1 Limitations of number types | 117 |
| 13.1.2 Using floating point numbers (carelessly) | 119 |
| 13.1.3 Using floating point numbers carefully 1 | 120 |
| 13.1.4 Using floating point numbers carefully 2 | 120 |
| 13.1.5 Symbolic calculation | 121 |
| 13.1.6 Summary | 123 |
| 13.1.7 Exercise: infinite or finite loop | 123 |
| 14 Numerical Python (numpy): arrays | 124 |
| 14.1 Numpy introduction | 124 |
| 14.1.1 History | 124 |
| 14.1.2 Arrays | 124 |
| 14.1.3 Convert from array to list or tuple | 127 |
| 14.1.4 Standard Linear Algebra operations | 127 |
| 14.1.5 More numpy examples... | 129 |
| 14.1.6 Numpy for Matlab users | 129 |
| 15 Visualising Data | 129 |
| 15.1 Matplotlib (Pylab) – plotting $y=f(x)$, (and a bit more) | 129 |
| 15.1.1 Matplotlib and Pylab | 130 |
| 15.1.2 First example | 130 |
| 15.1.3 How to import matplotlib, pylab, pyplot, numpy and all that | 131 |
| 15.1.4 IPython’s inline mode | 134 |
| 15.1.5 Saving the figure to a file | 134 |
| 15.1.6 Interactive mode | 135 |
| 15.1.7 Fine tuning your plot | 135 |
| 15.1.8 Plotting more than one curve | 139 |
| 15.1.9 Histograms | 142 |
| 15.1.10 Visualising matrix data | 144 |
| 15.1.11 Plots of $z=f(x,y)$ and other features of Matplotlib | 147 |
| 15.2 Visual Python | 147 |
| 15.2.1 Basics, rotating and zooming | 148 |
| 15.2.2 Setting the frame rate for animations | 148 |
| 15.2.3 Tracking trajectories | 149 |
| 15.2.4 Connecting objects (Cylinders, springs, ...) | 149 |
| 15.2.5 3d vision | 149 |
| 15.3 Visualising higher dimensional data | 150 |
| 15.3.1 Mayavi, Paraview, Visit | 150 |
| 15.3.2 Writing vtk files from Python (pyvtk) | 150 |
| 16 Numerical Methods using Python (scipy) | 151 |
| 16.1 Overview | 151 |
| 16.2 SciPy | 151 |
| 16.3 Numerical integration | 153 |
| 16.3.1 Exercise: integrate a function | 154 |
| 16.3.2 Exercise: plot before you integrate | 154 |
| 16.4 Solving ordinary differential equations | 154 |
| 16.4.1 Exercise: using odeint | 159 |

| | | |
|-----------|--|------------|
| 16.5 | Root finding | 159 |
| 16.5.1 | Root finding using the bisection method | 160 |
| 16.5.2 | Exercise: root finding using the bisect method | 160 |
| 16.5.3 | Root finding using the fsolve function | 161 |
| 16.6 | Interpolation | 161 |
| 16.7 | Curve fitting | 163 |
| 16.8 | Fourier transforms | 164 |
| 16.9 | Optimisation | 166 |
| 16.10 | Other numerical methods | 168 |
| 16.11 | scipy.io: Scipy-input output | 168 |
| 17 | Pandas - Data Science with Python | 170 |
| 17.1 | Motivational example (Series) | 171 |
| 17.2 | Pandas Series | 172 |
| 17.2.1 | Stock example - Series | 172 |
| 17.2.2 | memory usage | 176 |
| 17.2.3 | Statistics | 177 |
| 17.3 | Create Series from list | 177 |
| 17.4 | Plotting data | 178 |
| 17.5 | Missing values | 180 |
| 17.6 | Series data access: explicit and implicit (loc and iloc) | 181 |
| 17.6.1 | Indexing | 181 |
| 17.6.2 | Slicing | 182 |
| 17.6.3 | Data manipulation | 182 |
| 17.6.4 | Import and Export | 183 |
| 17.7 | Data Frame | 184 |
| 17.7.1 | Stock Example - DataFrame | 184 |
| 17.7.2 | Accessing data in a DataFrame | 185 |
| 17.7.3 | Extracting columns of data | 185 |
| 17.7.4 | Extracting rows of data | 186 |
| 17.7.5 | Data manipulation with shop | 187 |
| 17.8 | Example: European population 2017 | 187 |
| 17.9 | Further reading | 199 |
| 18 | Where to go from here? | 199 |
| 18.1 | Advanced programming | 200 |
| 18.2 | Compiled programming language | 200 |
| 18.3 | Testing | 200 |
| 18.4 | Simulation models | 200 |
| 18.5 | Software engineering for research codes | 200 |
| 18.6 | Data and visualisation | 200 |
| 18.7 | Version control | 200 |
| 18.8 | Parallel execution | 201 |
| 18.8.1 | Acknowledgements | 201 |