Python for data processing & analysis

Geert Jan Bex
geertjan.bex@uhasselt.be

Acknowledgements: thanks to Stefan Becuwe, Universiteit Antwerpen for suggestions and corrections

Motivation

- Why yet another programming language?
  - Programming languages have strong & weak points
  - Pick language for task at hand
- Why Python?
  - Useful for data processing
  - Terse language: express a lot in few lines of code
  - Short time to solution
  - Extensive standard library
  - Cross platform

> *Anybody who comes to you and says he has a perfect language is either naïve or a salesman.*
>
> — Bjarne Stroustrup

Python applications

- Python is general purpose programming language, but strong for
  - Data transformation: rewrite data into another format
    - Preprocessing/postprocessing/aggregating data
  - Prototyping
    - Experiment easily in Python, fast implementation later
    - Explorative programming
  - Glue/coordination language
    - Use Python as "scaffolding" for libraries in C/C++/Fortran
  - In-application scripting language
    - E.g., Kitware ParaView, Dassault Systèmes Abaqus™, Adobe Photoshop™
  - Graphical user interfaces
    - Wrap GUI around C/C++/Fortran code

Python versions

- Current 3.x
  - More clean than 2.x
  - Almost all Python libraries supported
- Version 2.7.x
  - Last of the 2.x releases
  - Many Python 3.x features have been retrofitted
  - All libraries support it

Note: in-application scripting may be stuck at Python 2.7!

Python 2 countdown: https://pythonclock.org/

Scope

- Teach programming in Python
  - prerequisite: you should know how to program in some other language, if not consider first completing
    - CodeAcademy
      http://www.codecademy.com/tracks/python
    - LearnPython
      http://www.learnpython.org/
- Highlight Python's strong points
- Discuss Python's weak points and how to mitigate

These sessions won't teach you how to program, how to find algorithms, that's beyond the scope

Training material

- All material available on GitHub
  - Google for '`gjbex github`'
  - https://github.com/gjbex/training-material/
  - repository name: `training-material`
    - Python directory
- Slides
  - https://github.com/gjbex/training-material/blob/master/Python/python_intro.pptx
  - click `Download` button
  - section title slides have links to relevant material for section

WHAT DO YOU WANT TO DO TODAY?

Running Python I

- Running Python from the command line
  - goals: run Python scripts in a shell
  - prerequisites: none
  - relevant sections: How to run Python from the shell?
- Interactive Python
  - goals: using Python for explorative programming using iPython & Jupyter notebooks
  - prerequisites: none
  - relevant sections: How to run Python from the shell?, How to run Python using Anaconda?

Basic Python programming

- Core Python programming
  - goals: Python syntax & semantics, control flow, data types, functions
  - prerequisites: experience in some programming language
  - relevant sections: data types & statements, standard I/O & command line arguments, additional datatypes, file I/O

Intermediate Python programming

- Object oriented programming
  - goals: creating Python classes, inheritance
  - prerequisites: core Python programming
  - relevant sections: object oriented programming, data representation (case study)
- Functional programming
  - goals: writing code using functional programming paradigm
  - prerequisite: core Python programming
  - relevant sections: list transformations, literators

Software engineering I

- Code organization
  - goals: organizing code of a non-trivial software project
  - prerequisites: core Python programming
  - relevant sections: code organization
- Documentation
  - goals: how to document Python code?
  - prerequisites: core Python programming
  - relevant sections: docstring & doctest

Software engineering II

- Testing
  - goals: tests are integral part of software development
  - prerequisites: core Python programming, object oriented programming for unit testing
  - relevant sections: doctest, unit testing
- Error handling
  - goals: catch & handle runtime errors
  - prerequisites: core Python programming, object oriented programming to define your own exceptions
  - relevant sections: error handling

Development I

- Debugging
  - goals: using the Python debugger
  - prerequisites: core Python programming
  - relevant sections: debugging
- Profiling
  - goals: using the Python profiler to identify optimization opportunities
  - prerequisites: core Python programming
  - relevant sections: profiling

Development II

- Python 2 versus Python 3
  - goals: compare Python versions, porting from Python 2 to 3
  - prerequisites: core Python programming
  - relevant sections: Python 2 to 3

Application development I

- Command line arguments & configuration files
  - goals: handling options, flags specified on command line, reading configuration files
  - prerequisites: core Python programming
  - relevant sections: argparse, ConfigParser
- Logging
  - goals: writing application events to log files, using log levels
  - prerequisites: core Python programming
  - relevant sections: logging

Application development II

- Interacting with the operating system
  - goals: file system operations, executing external commands
  - prerequisites: core Python programming
  - relevant sections: file system operations, external commands
- Web applications
  - goals: basic concepts of web application development
  - prerequisites: core Python programming, HTML + CSS
  - relevant section: GUI on the cheap

File formats

- Text-based formats
  - goals: reading & writing text-based file formats
  - prerequisites: core Python programming, file I/O
  - relevant sections: CSV & XML, regular expressions, web scraping, parsing regular languages, pyparsing for context-free languages, string formatting
- Scientific file formats
  - goals: reading & writing HDF5
  - prerequisites: core Python programming, numpy
  - relevant sections: HDF5

Numerical computing

- Linear algebra, numerical analysis
  - goals: various numerical analysis algorithms
  - prerequisites: core Python programming
  - relevant sections: numpy & scipy
- Scientific visualization
  - goals: creating 2D and 3D plots from Python
  - prerequisites: core Python programming, numpy
  - relevant sections: matplotlib, HoloViews, Bokeh

Symbolic computing

- Computer algebra
  - goals: various symbolic computations
  - prerequisites: core Python programming
  - relevant sections: sympy

Data analysis I

- Relational database interaction
  - goals: querying relational database systems
  - prerequisites: core Python programming, object oriented programming for SQLAlchemy
  - relevant sections: Relational databases
- Data analysis
  - goals: analysis using transforming & filtering tabular data, pivot tables, visualization
  - prerequisites: core Python programming
  - relevant sections: pandas

Data analysis II

- Image & video analysis
  - goals: analyzing and transforming images & videos
  - prerequisites: core Python programming, numpy, scipy
  - relevant sections: Image and video processing
- Machine learning
  - goals: analyzing and predicting from data
  - prerequisites: core Python programming, numpy, pandas
  - relevant sections: Machine learning

Data analysis III

- GIS data processing
  - goals: analyzing, transforming and creating GIS data
  - prerequisites: core Python programming, numpy, pandas
  - relevant sections: Graphical Information Systems data processing

Other training sessions

- High performance Python
  - Cython
  - Integrating C/C++/Fortran code, wrapping libraries
    - SWIG
    - f2py3
  - Shared memory programming
    - `multiprocessing`
    - futures
  - Distributed programming with mpi4py
  - PySpark
- Biopython (on demand)