Python for Scientific Computing*

Why Python for Scientific Computing

- For beginners there really is no difference between version two and three of *Python*.
- *Python* is the most popular coding language for teaching introductory computer science courses at top-ranked U.S. departments. Numerous online courses, lecture notes, and tutorials are readily available online.
- In the private sector, most recent results from *CodeEval* point in the same direction.
- *Python* is used heavily used by tech companies (e.g. Google, Dropbox, etc.) and in the financial sector (e.g. AQR).
- *Python* is so simple to learn, a lot if books explicitly target kids.
- Reliable interfaces to numerous language make for a good trade-off between developer and cycle time: (1) R, (2) MATLAB, (3) Fortran, and (4) C,

SciPy Stack

- *SciPy Library*, a collection of numerical algorithms and domain-specific toolboxes, including signal processing, optimization, statistics and much more
- *NumPy*, the fundamental package for numerical computation. It defines the numerical array and matrix types and basic operations on them
- *matplotlib*, a mature and popular plotting package, that provides publication-quality 2D plotting as well as rudimentary 3D plotting

*For further information or questions and suggestions, please contact us at info@policy-lab.org.

- *pandas*, providing high-performance, easy to use data structures
- *SymPy*, for symbolic mathematics and computer algebra
- *IPython*, a rich interactive interface, letting you quickly process data and test ideas
- *nose*, a framework for testing Python code.

Depending on your particular specialization, these packages might be of additional interest to you.

- *statsmodels*, a *Python* module that allows users to explore data, estimate statistical models, and perform statistical tests.

statsmodels, toghether with *pandas*, is a potential replacement for the *R*, just use *rpy2* to call *R* functions directly from *Python*. All these packages are included in the *Anaconda Distribution*.

Basic Example

- The *IPython* notebook works in your web browser, allowing you to document your computation in an easily reproducible form. See a notebook for Reinhart and Rogoff (2010) as an example here.

Data Visualization

- See the *matplotlib Thumbnail Gallery* for many and much more elaborate examples of data visualization.

Statistical Analysis

- We will fit an *Ordinary Least Squares (OLS)* model using *statsmodels*. See the online documentation for a full list of the library's capabilities.

Integrated Development Environment

- For simple analysis the IPython Notebook or even the command line is sufficient. However, for more involved scientific programming. I found the use of an IDE very useful.
- An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development.

- It provides numerous features that make coding easier such as name completion, renaming of methods and classes across multiple files, extracting methods, variables, parameters, static code analysis.

If we have time, we can get going on the *Getting Started Guide for Students* together.

Integrated Development Environment

If you have any further questions or comments, please do not hesitate to let me know. I also welcome any suggestions for improving this lecture.

References

Reinhart, C. M. and Rogoff, K. S. (2010). Growth in a Time of Debt. *American Economic Review*, 100(2):573–578.