

SOME NOTES ON PYTHON FOR FINANCE

ANTOINE JACQUIER

1. A BRIEF ENCOUNTER WITH PROGRAMMING LANGUAGES

1.1. A quick introduction to programming languages. Computing and programming are ubiquitous, in every area of every-day life, and are becoming increasingly important to deal with large flows of information. On financial markets, programming is fundamental to analyse time series of data, to evaluate financial derivatives, to run risk analyses, and to trade at high frequency, for example. Which programming language to use depends on one's needs, and the main factor is time: there are two types of times one should consider:

- Execution time is the time it takes to run the programme itself;
- Development time is the time it takes to write the code.

For ultra high-frequency trading, for instance, execution time is the most important, as the algorithm needs to make a decision very quickly. For long-term trading strategies, however, execution time is less important, and one might favour quicker development time.

1.2. Statically typed languages. For short execution time, lower level languages, which compile directly to machine code, are preferred. They often use static typing, namely data types have to be specified. C++ is the main example, and has been the main language used in quantitative finance; however, there is a non-negligible entry cost to it, understanding its underlying concepts such as memory allocation or pointers. Java is also a statically typed language, but automatically manages low-level memory allocation; that said, it does not compile directly to machine code, and needs a Java Virtual Machine to execute the Java bytecode generated by the programme. Historically slower than C++ (because of the virtual machine layer), recent advances have now made their speeds comparable.

1.3. Interpreted languages. When execution time is not the priority, and development time is preferred (for example for long-term strategies), one can instead use interpreted languages, such as Matlab, Python, or R. While the execution time is slower, these languages are dynamically typed, so that variables' types are automatically recognised by the programme and do not need to be specified by the user. Matlab has been a popular language in quantitative finance, and is still around because of its legacy code. However, in recent years, R (historically the preferred language for Statistics) and Python have been taking over, as they are open source and the range of available packages for applications has been growing exponentially. They are obviously slower than lower level languages, and some in-between languages have recently appeared, in particular Julia, which, when first run, generates machine code for execution.

1.4. Functional and query languages.

2. PYTHON

2.1. Python in Finance. A large number of financial companies, banks, hedge funds, asset managers, have recently adopted Python. JP Morgan's Quartz, based on Python, is used for pricing and risk analyses; BAML has its own version, Athena. One major drawback of Python is its Global Interpreter Lock (GIL), which only allows one thread to execute at every point in time, making it difficult to parallelise. Some Python libraries bypass the issue, for example the multiprocessing one, allowing the user to use multiple cores. Cython, on the other hand, is a static compiler for Python, and allows to convert some slow Python code (in particular loops) into much faster C versions.

2.2. General Python libraries. Python 3.5 is the default version of Python instead of 2.7. It is well supported by many packages to analyse data and perform statistical analysis.

- NumPy is the fundamental basic package for scientific computing with Python.
- SciPy supplements NumPy.
- pandas is a high-performance library for data analysis.
- matplotlib is the standard Python library for plots and graphs.

2.3. Python for Economics and Finance.

- quantdsl is a functional programming language for financial derivatives.
- statistics is a built-in Python library for basic statistical computations.
- ARCH: tools for econometrics.
- statsmodels allows to explore data, estimate statistical models, perform statistical tests.
- QuantEcon: library for economic modelling

2.4. Python libraries for plotting.

- matplotlib is the standard Python library for plots and graphs. It is fairly basic but can basically, with enough commands, generate any graphs.
- Seaborn is a powerful plotting library built on top of matplotlib.

2.5. Python libraries for Machine learning.

- scikit-learn adds to SciPy and NumPy common machine learning and data mining algorithms, such as clustering, regression, and classification.
- Theano has machine learning algorithms using the computer's GPU, and is hence extremely powerful for deep learning and heavy tasks.
- TensorFlow is a Google-supported ML library based on a multi-layer architecture.

3. ONLINE DATA SOURCES

Python makes it easy to query online databases directly, without having to import data locally.

3.0.1. Economics. FRED contains a vast collection of time series data maintained by the St. Louis Federal Reserve. For example, the entire series for the US civilian unemployment rate is available at <https://research.stlouisfed.org/fred2/series/unrate/downloaddata/UNRATE.csv>. Also, the World Bank collects and organises data on a huge range of indicators.

3.0.2. Finance database. Yahoo Finance, Google Finance are publicly available. Options market data, though, are not, but can be accessed via WRDS/OptionMetrics.

DEPARTMENT OF MATHEMATICS, IMPERIAL COLLEGE LONDON
E-mail address: `a.jacquier@imperial.ac.uk`