Python Programming for Finance

Juan F. Imbet
Université Paris Dauphine
Master 203 Financial Markets

Content

1. Introduction and Basic Programming: History, Variables, Control Flow, Conditional Evaluation, Arrays, Functions, Generators.
2. Intermediate Programming: Numerical Python (numpy), Linear Algebra
3. Data Analysis: Dataframes (pandas), cleaning and processing data through a Financial Example.
4. Portfolio Management I: Statistics and Simulation
5. Portfolio Management II: Optimization and Heuristics, Parallelization
6. Web Scrapping: How to automatize data collection online through a Financial Example
7. Data Distribution: How to build a functional API to distribute data.
8. A grasp of advanced Python usage: Coding Standards, General User Interfaces, Extending Python's Functionality with C++

Grading

- Final Exam 70%
- Project 30%

Course Materials

- The course material, slides and code, will be available on the Github repository of the course, and will be updated on a weekly basis.
- We will run Python code using Google's research colab platform. However, it is recommended to install Python ≥ 3.7 locally.

What is Python

Python is a **high level, interpreted, interactive,** and **multi-paradigm** scripting language.

- **High level**: Python has a strong abstraction from the details of the computer.
- **Interpreted**: There is no need to compile your program before executing it.
- **Interactive**: You can open a Python terminal and directly write your code.
- **Multi-paradigm**: Python supports different programming and implementation paradigms, such as object orientation and imperative, functional, or procedural programming.

History of Python

- First developed by Guido van Rossum in the early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
- The name was inspired by Monty Pythonâs Flying Circus (sketch comedy from the British comedy troupe).

Why Python?

- Easy-to-learn
- Easy-to-read
- Easy-to-maintain
- Easy-to-scale
- Relative fast
- Open source

Python files usually end in .py, while other files that depend on Python but are not written fully in Python like Jupyter notebooks end in .ipynb.

Python in Finance

- Corporations often have a team whose task is to perform quantitative analysis. (Credit risk, quant trading...)
- Although many of the core programs used by corporations are written in C, C++, Java, COBOL or other robust languages, Python serves as a wrapper to process, analyze, display, and distribute data.
- The numerical libraries in Python makes it comparable to R or Stata in performing Statistics, and to Matlab for linear algebra.
- Helps automatize research tasks and increase productivity.
- Increasing popularity among developers (2020 Stackoverflow survey) JavaScript (69.7%), HTML/CSS (62.4%), SQL (56.9%), Python (41.6%) and Java (38.4%).
- Is Free

Setup for the Course (Required for next class)

1. Open a Github account
2. Get familiar with Git and Github. I recommend to install it locally and use it for your own work.
3. Course's lessons will appear here.
4. Login (Google account) to Google's Colab
5. Use Google Colab with Github
6. Install Python locally using Anaconda plus an editor of your choice,

What is a programming language

- A high-level programming language provides instructions to perform arithmetic operations in the computer.
- The computer does not understand these instructions, unless they are translated to machine code.
- This task often requires an intermediary (Assembly language)

Basic computing

- All computer programs can be decomposed into simple and fundamental operations (Bit or logical operations).
- A bit (binary digit) is the simplest unit of information a standard computer handles. It can only hold two values (1 or 0), and is physically represented by a transistor being on or off in old computers, or by the presence or absence of electrons in semiconductors.
- As bits only take two values, it is natural that computers perform binary arithmetic.

Floating point arithmetic

- Since computers perform mathematical operations, they require a way to represent numbers. Most computers use up to 64 (32) bits to represent a real number.
- The 64 (32) architecture allows computers to perform floating point arithmetic by representing the whole real line with float numbers.
- Transforms the real line into a discrete and finite set of numbers.
- Gaps between numbers are not constant, and depend on the magnitude of the number.
- When arithmetic operations result in a number that cannot be expressed exactly as a floating point, the computer approximates the value.