Python est un langage de programmation très polyvalent et modulaire, qui est utilisé aussi bien pour écrire des applications comme YouTube, que pour traiter des données scientifiques. Par conséquent, il existe de multiples installations possibles de Python. L'utilisateur débutant peut donc se sentir dérouté par l'absence d'une référence unique pour Python scientifique. Nous conseillons donc un logiciel unique pour la formation, la suite scientifique Anaconda développée par l'entreprise Continuum. Anaconda rassemble tout le nécessaire pour l'enseignement de Python scientifique: le langage Python et ses modules scientifiques.

En particulier, Anaconda fournit un environnement de travail adapté à l'enseignement et au calcul scientifique, spyder .

un installeur à partir de http://continuum.io/downloads.html, correspondant à votre système d'exploitation (Windows, Mac OS X, Linux). Il faut choisir entre 32 bits ou 64 bits selon que votre système d'exploitation est 32 bits ou 64 bits.

http://continuum.io/downloads

Choose Your Installer: x64 python 2.7

Windows 64-bit
Python 2.7
Graphical Installer

- Size: 366M

Choose Your Installer: x64 python 3.4

I want Python 2.7*
Windows 64-bit
Python 3.4
Graphical Installer

- Size: 363M

Python for Science

Geared toward scientists and engineers, Python for Science provides a strong foundation which will enable you to work and develop scientific and engineering work-flows much more rapidly.

Overview

This course begins with an abbreviated primer on Python (language syntax, data structures, basic data processing, Python functions, modules and classes). The remainder of the course covers open source Python tools relevant to solving your day-to-day scientific and engineering programming problems. Specific topics examined include: array computation and mathematics with NumPy; statistics, linear algebra, optimization, interpolation, and advanced computation with SciPy; working with tabular data in Pandas to generate summary statistics and rolling window calculations; and using matplotlib to create 2-d and 3-d visualizations.

To interface with the rest of your libraries, we teach ways of wrapping C/C++ and/or Fortran code in Python, along with ways to compile Python for dramatically improved performance. We will also cover how to call Python functions from Excel, manipulate spreadsheets and documents using Python, and expose your Python code over the web.

Syllabus

The course is taught over four days. It covers the Python language, critical library modules for science, and data visualization.

1. **Introduction to Python.**
   The first section of the course is an introduction to the Python programming language. It introduces the details of how to start and stop the interpreter and write programs, and explains basic data-types, files, functions, and error handling.

2. **Working with Data.**
   Section 2 provides a detailed tour of how to represent and work with data in Python. It covers tuples, lists, dictionaries, and sets, as well as NumPy arrays. Students will learn the critical aspects of Python's underlying object model including variables, reference counting, copying, and type checking and how to effectively use Python's very powerful list processing primitives.

3. **Program Organization and Functions.**
   More information about how to organize larger programs into functions is provided in Section 3. A major focus is placed on design functions and the technical details of functions, including scoping rules and documentation strings.

4. **Modules and Libraries.**

Section 4 addresses how to organize programs into modules and use those modules as a tool for creating extensible programs. It concludes with a tour of some of the most commonly used library modules including those related to system administration, text processing, subprocesses, XML parsing, binary data handling, and databases.

5. **Classes and Objects.**

In Section 5, students are given an introduction to object-oriented programming in Python. Topics such as how to create new objects, overload operators, and utilize Python special methods will be explained, along with the basic principles of object oriented programming including inheritance and composition.

6. **NumPy.**

NumPy is Python's array and mathematics library. Section 6 covers how one can and should use NumPy to simplify and speed up code. It starts with a brief overview of NumPy's memory model and teaches ways to slice and dice data using NumPy's sophisticated indexing operations. Mathematical use cases, various interpolation functions, matrix decompositions, computing eigenvalues, solving systems of equations and inverting matrices are also covered.

7. **SciPy.**

SciPy is Python's scientific complement to NumPy. SciPy is quite broad and used in many different domains. Section 7 covers key areas relevant to finance, starting with SciPy's statistical functions, how to generate different distributions and perform statistical computations. It also covers some interpolation, linear algebra, optimizations, signal processing, and fourier transforms. These are useful in efficiently computing some moving window functions (moving averages, exponential moving averages, bollinger bands).

8. **Pandas.**

Pandas is Python's answer to the R dataframe, which provides a very convenient way to work with tabular data. It helps organize your data by handling missing data and data alignment automatically. In Section 8, students learn about Pandas ability to do simple olap aggregations and drill downs without additional code, as well as executing efficient computations for statistical functions, like moving window functions and rolling regressions.

9. **Integrating with Native Code.**

Python is fantastic, but often we have to work with code in pre-existing C/C++ libraries. One of the easiest ways to do this is with Python's built-in ctypes libraries, but SWIG/Cython/Boost and hand-generated wrappings are all ways to get the job done as well. We will preview using Numba to compile Python syntax directly to machine code. In Section 9, students will examine several use cases that show how to write Python code first and optimize for speed later.

10. **Data Visualization.**

Matplotlib is a Python visualization library which makes it easy to generate static plots. Section 10 will show students how to use Matplotlib to create basic line plots, scatter plots,

density estimation plots, as well as date handling. If you have any specific plotting requirements, let us know and we can cover those use cases as well.