Finanças Quantitativas com Python

Wilson Freitas
TDC 2016

Sobre mim

- Físico
- Padeiro
- Corredor
- Trabalha no mercado financeiro como *Quant*

Finanças Quantitativas ?!?

QUANT FINANCE

What my friends think I do.

What my mom thinks I do.

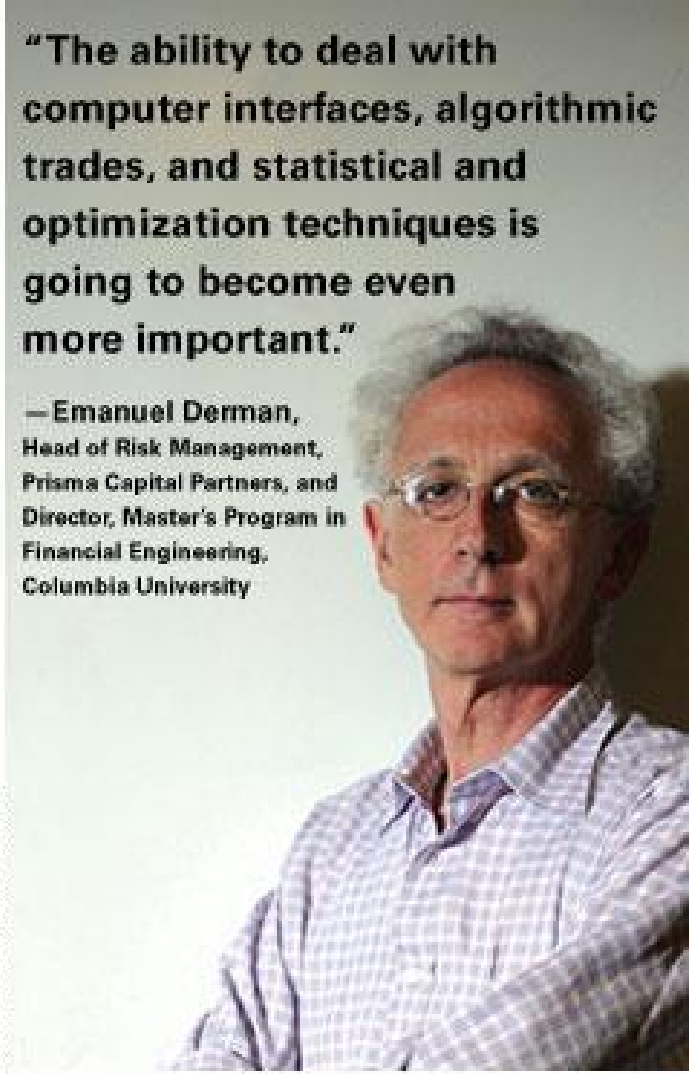
What society thinks I do.

What my boss thinks I do.

What I think I do.

What I actually do.

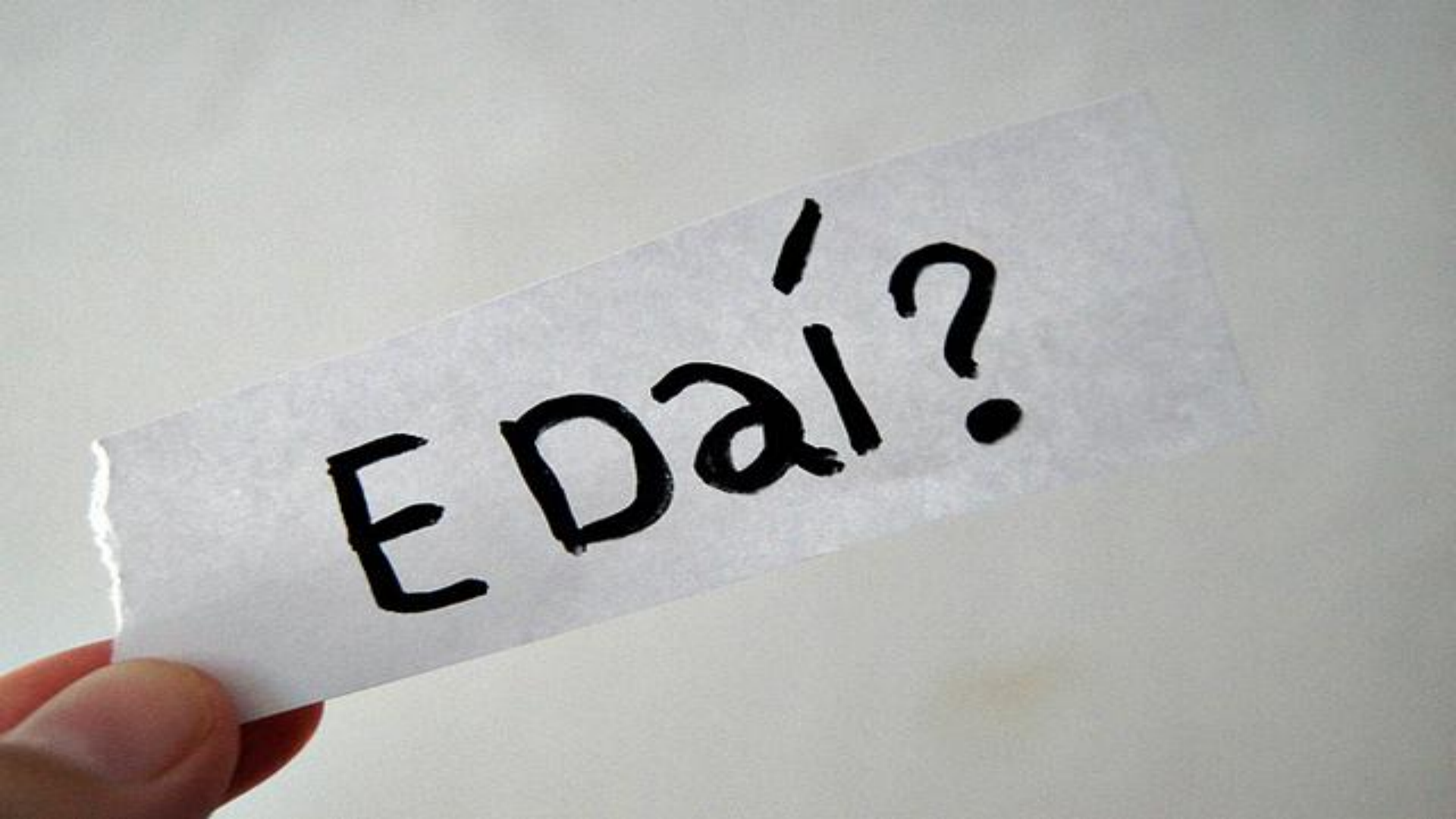

model
Mathematical
pricing
finance
financial
models
derivatives
Black
analysis
probability
Finance
Risk
distribution
time
volatility
options
world
market
price
risk
portfolio
management
Pricing
Derivatives
Scholes
Markowitz
Sharpe
fundamental
mathematics
stochastic
risk-neutral
Model
Gaussian
Financial
Wikipedia
changes
Brownian
mathematical
Merton
securities
prices
fair
value
security
Volatility
theory
process
calculus
equations
topics
differential
advanced
modeling
Quantitative
method
List
Paul
Derman
mean
Asset
See
Influential
estimation
Sciences
normalized
Meucci
Taleb
see
versus
Dow
modeled
distributions
current
Memorial
field
relationship
Monte
bonds
asset
best
community
thus
References
Wikimedia
tools
future
option
partial
Modelers
Manifesto
difference
Numerical
must
Privacy
work
technical
series
stock
denoted
mathbb
Ito
measure
Carlo
may
given
Swan
Nassim
use
Theory
continuous-time
markets
Sbochastic
variance
Finalo
Valuation
Challenges
Tools
sell-side
utility
main
Economic
define
links
called
economics
developed
theorem
ISBN
January
return
set
along
articles
processes
practice
i.e
quantitative
investment
general
New
also
focuses
continuous
derivative
Abtilio
arbitrage-free
sophisticated



In physics it takes 3 laws to explain 99% of the data

In finance it takes more than 99 equations to explain about 3%

Emanuel Derman


E Daí?

Bloomberg Businessweek
SPECIAL DOUBLE ISSUE June 15 — June 28, 2015 | bloomberg.com

```
import datetime

class Issue():
    """TODO write docs here"""
    def __init__(self, **kwargs):
        # TODO: Validate input
        self.__dict__.update(kwargs)

    def publish(self):
        return ('This is the {0.pubdate:%B %d, %Y} issue of {0.title}. ' +
                'It is {0.pages:,} pages long, and ' +
                'costs ${0.price:.5}. '
                'It is about {0.subject}.').format(self)

if __name__ == '__main__':
    bbw = Issue(title='Bloomberg Businessweek',
                price=5.99,
                # That price is only USD;
                # TODO figure out international pricing/currencies
                pages=112,
                pubdate=datetime.datetime(2015, 6, 15),
                subject="code")
    print(bbw.publish())
```

If You Can't Read That, You'd Better Read This Code: An Essay p.13

Computational Finance – Why Python Is Taking Over

Technology plays an important role in the financial industry. However, some areas have nevertheless been dominated by human beings for quite a long time. Among these areas is trading, where real people in general have at least the final word. This might change soon, as Robin Wigglesworth on the front page of the *Financial Times* from November 23, 2015 points out: "Traders used to be first-class citizens of the financial world, but that's not true anymore. Technologists are the priority now ..." In this regard, banks, hedge funds, and asset managers are competing more and more with the big Silicon Valley employers, like Google and Facebook, for the brightest IT talent available.

In this context, the open-source programming language Python is becoming an increasingly important technology – used nowadays extensively in Silicon Valley as well as in finance. There are a few good reasons why this is the case.

The Python language

Python's syntax and idioms make it particularly attractive for science in general and finance in particular. Consider the "Hello World!" example of quantitative finance, the geometric Brownian motion model of Black–Scholes–Merton (1973) describing, for instance, the evolution of a stock index. In its risk-neutral form, the stochastic differential equation (SDE) reads as follows (with the parameters and symbols defined as usual and in particular Z_t being a Brownian motion):

$$dS_t = rS_t dt + \sigma S_t dZ_t$$

An Euler discretization scheme is then given by (with z_T being a standard normally distributed random variable)

$$S_T = S_0 \exp\left(\left(r - \frac{1}{2}\sigma^2\right)T + \sigma\sqrt{T}z_T\right)$$

In LaTeX you would describe this difference equation, for example, by

```
S_T = S_0 \exp {( r - 0.5 \sigma^2 ) T + \sigma
    \sqrt{T} z_T)
```

Something the majority of quants have done many, many times when documenting their work. The Python code for the difference equation is pretty close to its LaTeX representation:

```
In [1]: from pylab import *
S_0 = 100.; r = 0.01; T = 0.5; sigma = 0.2
   # parameters
z_T = standard_normal(10000)
   # pseudo-random numbers
S_T = S_0 * exp (( r - 0.5 * sigma ** 2 )
   * T + sigma * sqrt(T) * z_T)
```

With four lines of Python code, we import the necessary functionality (mainly from the NumPy library), draw 10,000 pseudo-random numbers, and simulate 10,000 end-of-period values. This vectorized approach not only leads to concise code (without loops) but is also quite fast, since the library used is mainly implemented in C.

The Python ecosystem

There are some other languages that have a similar syntax and also use approaches like vectorization. But there is one feature that distinguishes Python from other (general-purpose) programming languages: its comprehensive ecosystem of open-source scientific libraries. To mention just a few:

- **NumPy** (fast, vectorized array operations)
- **SciPy** (collection of scientific classes/functions)
- **Cython** (static compiler for hybrid Python and C code)
- **pandas** (times series and tabular data management)
- **ibis** (Pythonic interaction with, e.g., relational databases)
- **PyTables** (hardware-bound IO operations)
- **TsTables** (high-performance tick data storage/retrieval)
- **scikit-learn** (machine learning algorithms)
- **statsmodels** (statistical classes/functions)
- **xlwings** (Python–Excel integration)

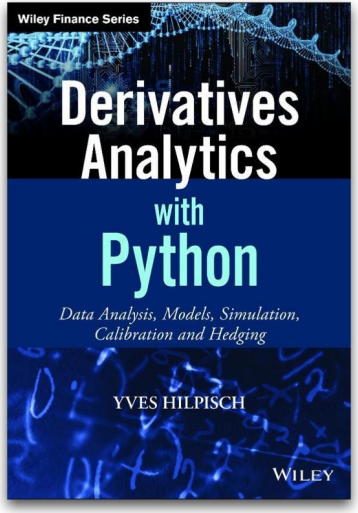
In addition, there are a wealth of libraries for efficient and nice visualizations, like matplotlib, seaborn, or plotly.

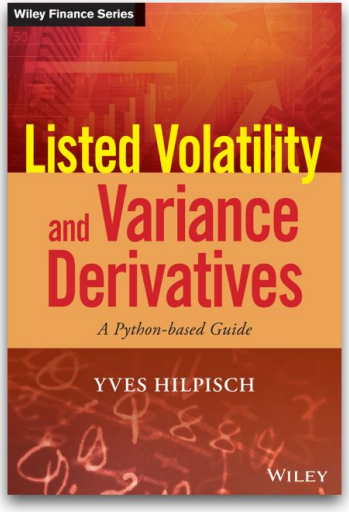
Financial libraries

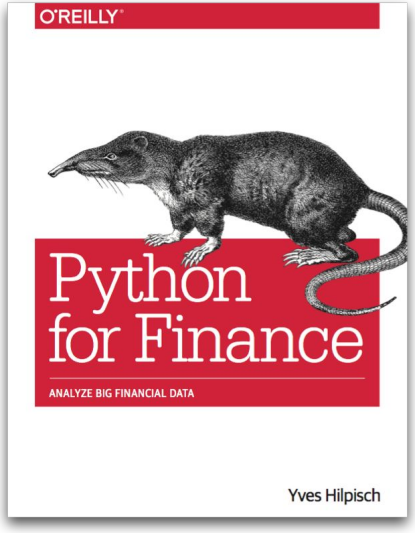
When it comes to dedicated financial libraries, the number of options is also steadily growing. A few examples are:

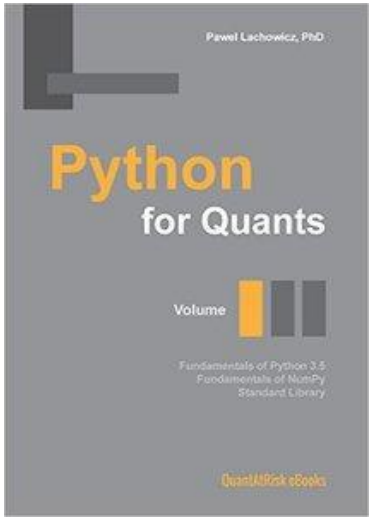
- **DX Analytics** (global valuation of multi-risk derivatives and portfolios)
- **pyfolio** (portfolio management, performance analysis)
- **PyThalesians** (data, backtesting, trading)
- **zipline** (backtesting of trading algorithms and strategies)

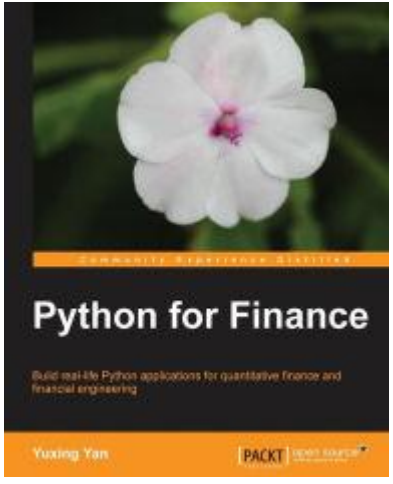
However, there is no single, unified financial library available at the beginning of 2016 that has accomplished what pandas has done on the data analytics side.

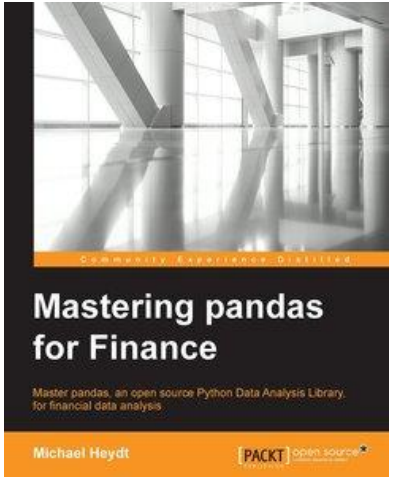

Wiley Finance Series
Derivatives Analytics with Python
Data Analysis, Models, Simulation, Calibration and Hedging
YVES HILPISCH
WILEY

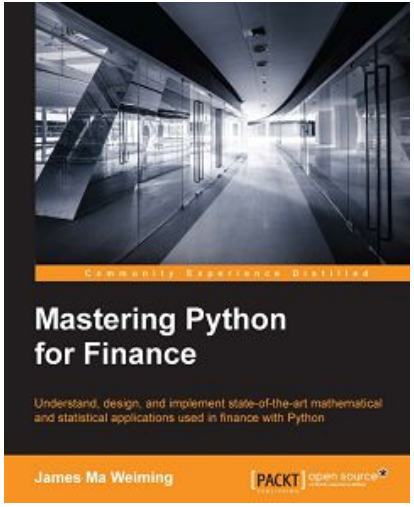

Wiley Finance Series
Listed Volatility and Variance Derivatives
A Python-based Guide
YVES HILPISCH
WILEY

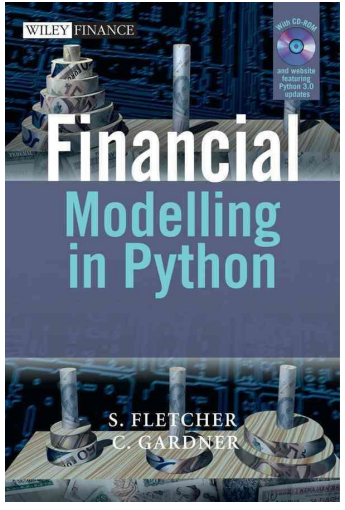

O'REILLY
Python for Finance
ANALYZE BIG FINANCIAL DATA
Yves Hilpisch

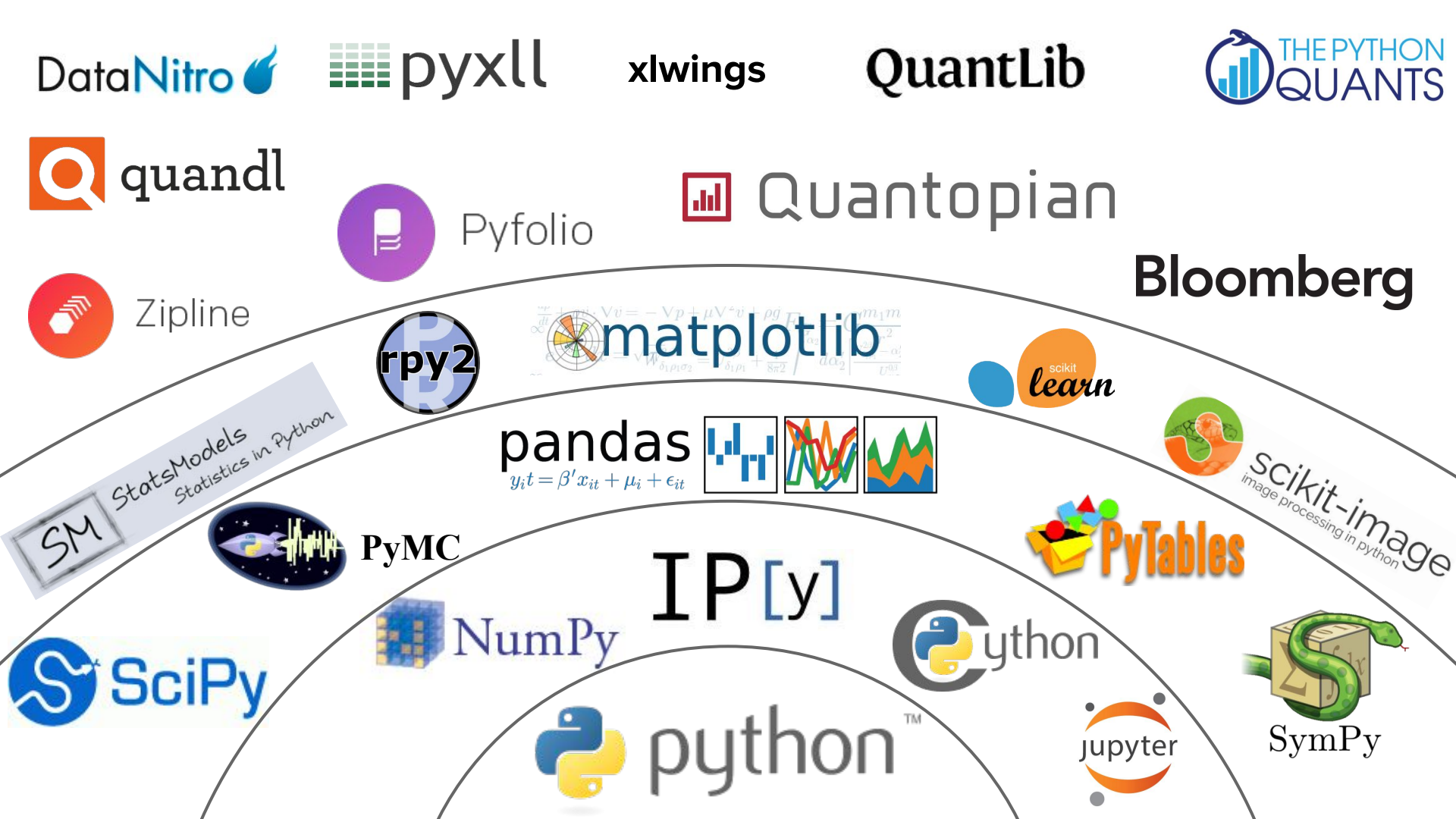

Pawel Lachowicz, PhD
Python for Quants
Volume
Fundamentals of Python 3.5
Fundamentals of NumPy
Standard Library
QuantAtRisk eBooks


Python for Finance
Build real-life Python applications for quantitative finance and financial engineering
Yuxing Yan
PACKT open source


Mastering pandas for Finance
Master pandas, an open source Python Data Analysis Library, for financial data analysis
Michael Heydt
PACKT open source


Mastering Python for Finance
Understand, design, and implement state-of-the-art mathematical and statistical applications used in finance with Python
James Ma Weiming
PACKT open source


WILEY FINANCE
Financial Modelling in Python
S. FLETCHER
C. GARDNER


DataNitro
pyxll
xlwings
QuantLib
THE PYTHON QUANTS
quandl
Quantopian
Pyfolio
Bloomberg
Zipline
rpy2
matplotlib
scikit learn
pandas
$y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$
StatsModels
Statistics in Python
scikit-image
image processing in python
PyMC
PyTables
IP[y]
NumPy
Cython
SciPy
python
jupyter
SymPy

Acesso a Dados

- Python possui 20+ bibliotecas para acesso a dados (awesome-quant)

- Bloomberg tem uma API em Python para acesso sua infraestrutura
  - API é liberada, mas precisa assinar a Bloomberg $$$$

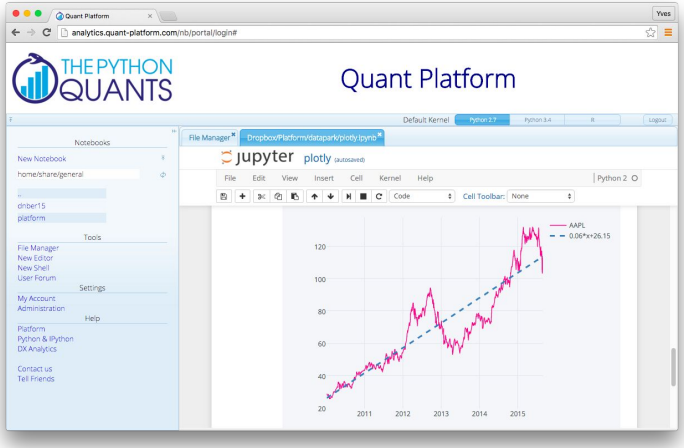
- Quandl tem o melhor serviço de acesso a dados públicos (IMHO)
  - Tem dados do Banco Central do Brasil (BCB)

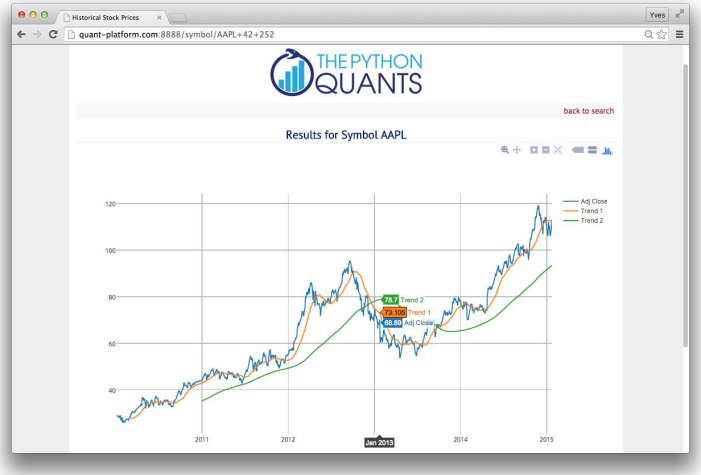
Plataformas

The Python Quants

Plataforma para modelagem de finanças quantitativas:

- Estratégias de Trading
- Gestão de Risco
- Modelagem e Análise de Dados Financeiros

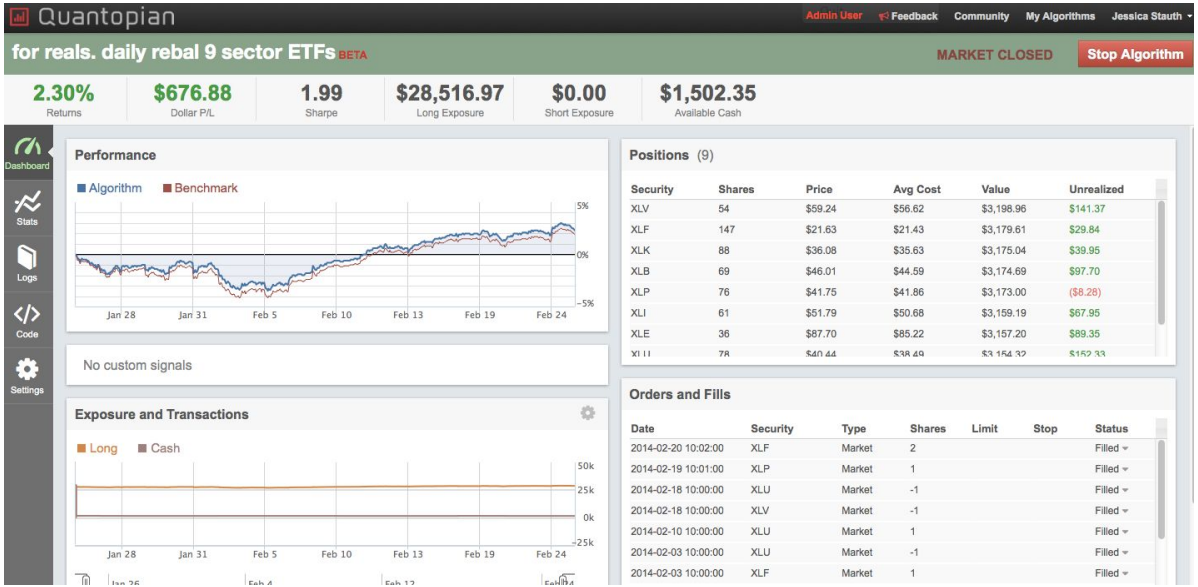




Quantopian

Plataforma para desenvolvimento de algoritmos e estratégias de trading:

- zipline - Pythonic algorithmic trading library
- pyfolio - Portfolio and Risk analytics in Python



Integração com Excel

http://www.desentupidoradenoticias.com.br/2015/03/6-alimentos-que-fazem-emagrecer-de.html

0%

É o quanto se faz sem integrar com Excel

Não, *Office não são opções

Python tem diversas opções

openpyxl - Read/Write Excel 2007 xlsx/xlsm files

xlrd - Extract data from Excel spreadsheets (.xls and .xlsx, versions 2.0 onwards) on any platform

xlsxwriter - Write files in the Excel 2007+ XLSX file format

xlwt - Generate spreadsheet files that are compatible with Excel 97/2000/XP/2003, OpenOffice.org Calc, and Gnumeric.

Tem mais

DataNitro - DataNitro also offers full-featured Python-Excel integration.

ExcelPython - ExcelPython is a free, open-source library.

ExPy - ExPy is freely available demonstration software that is simple to install. Once installed, Excel users have access to built-in Excel functions that wrap Python code. Documentation and examples are provided at the site.

PyXLL - PyXLL is a widely used tool (free for personal or educational use) that implements UDFs written in Python as add-in functions for Excel.

mahalo iā ʻoe

wilsonfreitas

@aboutwilson