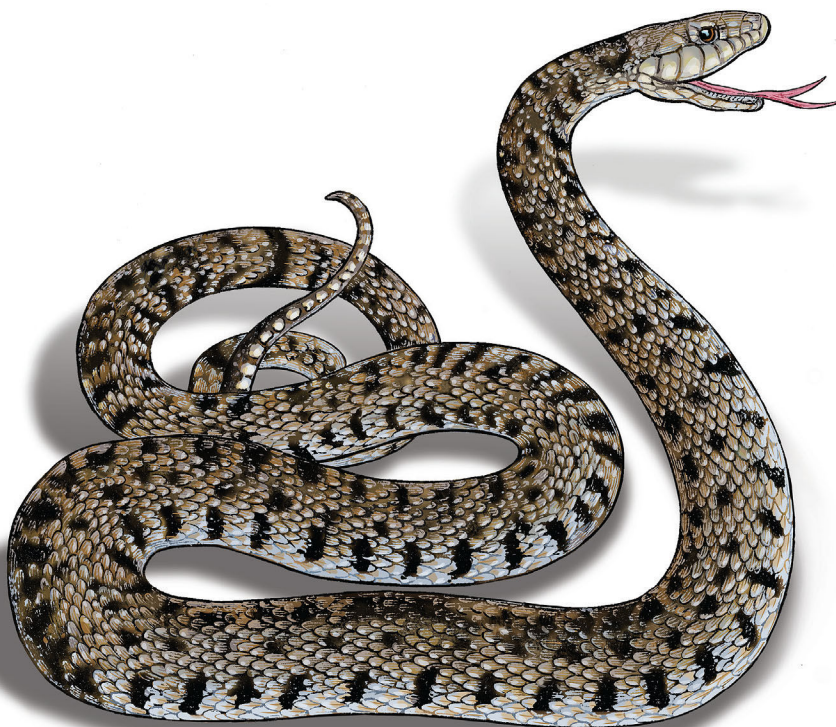


O'REILLY®

# Python for Algorithmic Trading

From Idea to Cloud Deployment



Yves Hilpisch

---

# Python for Algorithmic Trading

*From Idea to Cloud Deployment*

*Yves Hilpisch*

Beijing • Boston • Farnham • Sebastopol • Tokyo

**O'REILLY®**

---

# Table of Contents

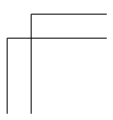
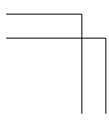
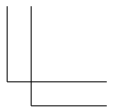
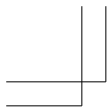
<b>Preface.....</b>	<b>ix</b>
<b>1. Python and Algorithmic Trading.....</b>	<b>1</b>
Python for Finance	1
Python Versus Pseudo-Code	2
NumPy and Vectorization	3
pandas and the DataFrame Class	5
Algorithmic Trading	7
Python for Algorithmic Trading	11
Focus and Prerequisites	13
Trading Strategies	13
Simple Moving Averages	14
Momentum	14
Mean Reversion	14
Machine and Deep Learning	14
Conclusions	15
References and Further Resources	15
<b>2. Python Infrastructure.....</b>	<b>17</b>
Conda as a Package Manager	19
Installing Miniconda	19
Basic Operations with Conda	22
Conda as a Virtual Environment Manager	27
Using Docker Containers	30
Docker Images and Containers	31
Building a Ubuntu and Python Docker Image	31
Using Cloud Instances	36
RSA Public and Private Keys	38

Jupyter Notebook Configuration File	39
Installation Script for Python and Jupyter Lab	40
Script to Orchestrate the Droplet Set Up	42
Conclusions	44
References and Further Resources	44
<b>3. Working with Financial Data.....</b>	<b>47</b>
Reading Financial Data From Different Sources	48
The Data Set	48
Reading from a CSV File with Python	49
Reading from a CSV File with pandas	51
Exporting to Excel and JSON	52
Reading from Excel and JSON	53
Working with Open Data Sources	54
Eikon Data API	57
Retrieving Historical Structured Data	60
Retrieving Historical Unstructured Data	65
Storing Financial Data Efficiently	67
Storing DataFrame Objects	68
Using TsTables	72
Storing Data with SQLite3	77
Conclusions	79
References and Further Resources	79
Python Scripts	80
<b>4. Mastering Vectorized Backtesting.....</b>	<b>83</b>
Making Use of Vectorization	84
Vectorization with NumPy	85
Vectorization with pandas	87
Strategies Based on Simple Moving Averages	90
Getting into the Basics	91
Generalizing the Approach	99
Strategies Based on Momentum	100
Getting into the Basics	101
Generalizing the Approach	106
Strategies Based on Mean Reversion	109
Getting into the Basics	109
Generalizing the Approach	112
Data Snooping and Overfitting	113
Conclusions	115
References and Further Resources	115
Python Scripts	117

SMA Backtesting Class	117
Momentum Backtesting Class	120
Mean Reversion Backtesting Class	122
<b>5. Predicting Market Movements with Machine Learning.....</b>	<b>125</b>
Using Linear Regression for Market Movement Prediction	126
A Quick Review of Linear Regression	127
The Basic Idea for Price Prediction	129
Predicting Index Levels	131
Predicting Future Returns	134
Predicting Future Market Direction	136
Vectorized Backtesting of Regression-Based Strategy	137
Generalizing the Approach	139
Using Machine Learning for Market Movement Prediction	141
Linear Regression with scikit-learn	141
A Simple Classification Problem	143
Using Logistic Regression to Predict Market Direction	147
Generalizing the Approach	152
Using Deep Learning for Market Movement Prediction	155
The Simple Classification Problem Revisited	156
Using Deep Neural Networks to Predict Market Direction	158
Adding Different Types of Features	164
Conclusions	168
References and Further Resources	169
Python Scripts	169
Linear Regression Backtesting Class	169
Classification Algorithm Backtesting Class	172
<b>6. Building Classes for Event-Based Backtesting.....</b>	<b>177</b>
Backtesting Base Class	179
Long Only Backtesting Class	184
Long Short Backtesting Class	187
Conclusions	191
References and Further Resources	192
Python Scripts	193
Backtesting Base Class	193
Long Only Backtesting Class	196
Long Short Backtesting Class	198
<b>7. Working with Real-Time Data and Sockets.....</b>	<b>203</b>
Running a Simple Tick Data Server	205
Connecting a Simple Tick Data Client	208

Signal Generation in Real Time	210
Visualizing Streaming Data with Plotly	213
The Basics	213
Three Real-Time Streams	215
Three Sub-Plots for Three Streams	216
Streaming Data as Bars	218
Conclusions	219
References and Further Resources	220
Python Scripts	220
Sample Tick Data Server	220
Tick Data Client	221
Momentum Online Algorithm	222
Sample Data Server for Bar Plot	222
<b>8. CFD Trading with Oanda.....</b>	<b>225</b>
Setting Up an Account	229
The Oanda API	231
Retrieving Historical Data	233
Looking Up Instruments Available for Trading	233
Backtesting a Momentum Strategy on Minute Bars	233
Factoring In Leverage and Margin	237
Working with Streaming Data	239
Placing Market Orders	240
Implementing Trading Strategies in Real Time	242
Retrieving Account Information	247
Conclusions	250
References and Further Resources	250
Python Script	250
<b>9. FX Trading with FXCM.....</b>	<b>253</b>
Getting Started	255
Retrieving Data	255
Retrieving Tick Data	256
Retrieving Candles Data	258
Working with the API	260
Retrieving Historical Data	261
Retrieving Streaming Data	263
Placing Orders	264
Account Information	266
Conclusions	267
References and Further Resources	268

<b>10. Automating Trading Operations.....</b>	<b>269</b>
Capital Management	270
Kelly Criterion in Binomial Setting	270
Kelly Criterion for Stocks and Indices	276
ML-Based Trading Strategy	281
Vectorized Backtesting	282
Optimal Leverage	288
Risk Analysis	290
Persisting the Model Object	294
Online Algorithm	294
Infrastructure and Deployment	299
Logging and Monitoring	300
Visual Step-by-Step Overview	303
Configuring Oanda Account	303
Setting Up the Hardware	304
Setting Up the Python Environment	304
Uploading the Code	306
Running the Code	306
Real-Time Monitoring	308
Conclusions	308
References and Further Resources	309
Python Script	309
Automated Trading Strategy	309
Strategy Monitoring	312
<b>Appendix. Python, NumPy, matplotlib, pandas.....</b>	<b>313</b>
<b>Index.....</b>	<b>355</b>





---

# Preface

Dataism says that the universe consists of data flows, and the value of any phenomenon or entity is determined by its contribution to data processing....Dataism thereby collapses the barrier between animals [humans] and machines, and expects electronic algorithms to eventually decipher and outperform biochemical algorithms.<sup>1</sup>

—Yuval Noah Harari

Finding the right algorithm to automatically and successfully trade in financial markets is the holy grail in finance. Not too long ago, algorithmic trading was only available and possible for institutional players with deep pockets and lots of assets under management. Recent developments in the areas of open source, open data, cloud compute, and cloud storage, as well as online trading platforms, have leveled the playing field for smaller institutions and individual traders, making it possible to get started in this fascinating discipline while equipped only with a typical notebook or desktop computer and a reliable internet connection.

Nowadays, Python and its ecosystem of powerful packages is the technology platform of choice for algorithmic trading. Among other things, Python allows you to do *efficient data analytics* (with, for example, `pandas` (<http://pandas.pydata.org>)), to apply *machine learning* to stock market prediction (with `scikit-learn` (<http://scikit-learn.org>), for example), or even to make use of Google's *deep learning* technology with `tensorflow` (<http://tensorflow.org>).

This is a book about Python *for* algorithmic trading, primarily in the context of *alpha generating strategies* (see Chapter 1). Such a book at the intersection of two vast and exciting fields can hardly cover all topics of relevance. However, it can cover a range of important meta topics in depth.

---

<sup>1</sup> Harari, Yuval Noah. 2015. *Homo Deus: A Brief History of Tomorrow*. London: Harvill Secker.

These topics include:

*Financial data*

Financial data is at the core of every algorithmic trading project. Python and packages like NumPy and pandas do a great job of handling and working with structured financial data of any kind (end-of-day, intraday, high frequency).

*Backtesting*

There should be no automated algorithmic trading without a rigorous testing of the trading strategy to be deployed. The book covers, among other things, trading strategies based on simple moving averages, momentum, mean-reversion, and machine/deep-learning based prediction.

*Real-time data*

Algorithmic trading requires dealing with real-time data, online algorithms based on it, and visualization in real time. The book provides an introduction to socket programming with ZeroMQ and streaming visualization.

*Online platforms*

No trading can take place without a trading platform. The book covers two popular electronic trading platforms: Oanda (<http://oanda.com>) and FXCM (<http://fxcm.com>).

*Automation*

The beauty, as well as some major challenges, in algorithmic trading results from the automation of the trading operation. The book shows how to deploy Python in the cloud and how to set up an environment appropriate for automated algorithmic trading.

The book offers a unique learning experience with the following features and benefits:

*Coverage of relevant topics*

This is the only book covering such a breadth and depth with regard to relevant topics in Python for algorithmic trading (see the following).

*Self-contained code base*

The book is accompanied by a Git repository with all codes in a self-contained, executable form. The repository is available on the Quant Platform (<http://py4at.pqp.io>).

*Real trading as the goal*

The coverage of two different online trading platforms puts the reader in the position to start both paper and live trading efficiently. To this end, the book equips the reader with relevant, practical, and valuable background knowledge.

### *Do-it-yourself and self-paced approach*

Since the material and the codes are self-contained and only rely on standard Python packages, the reader has full knowledge of and full control over what is going on, how to use the code examples, how to change them, and so on. There is no need to rely on third-party platforms, for instance, to do the backtesting or to connect to the trading platforms. With this book, the reader can do all this on their own at a convenient pace and has every single line of code to do so.

### *User forum*

Although the reader should be able to follow along seamlessly, the author and The Python Quants are there to help. The reader can post questions and comments in the user forum on the Quant Platform (<http://home.tpq.io/ppp>) at any time (accounts are free).

### *Online/video training (paid subscription)*

The Python Quants offer comprehensive online training programs (<https://oreil.ly/Qty90w>) that make use of the contents presented in the book and that add additional content, covering important topics such as financial data science, artificial intelligence in finance, Python for Excel and databases, and additional Python tools and skills.

## Contents and Structure

Here's a quick overview of the topics and contents presented in each chapter.

### *Chapter 1*

The first chapter is an introduction to the topic of algorithmic trading—that is, the automated trading of financial instruments based on computer algorithms. It discusses fundamental notions in this context and also addresses, among other things, what the expected prerequisites for reading the book are.

### *Chapter 2*

This chapter lays the technical foundations for all subsequent chapters in that it shows how to set up a proper Python environment. This chapter mainly uses `conda` as a package and environment manager. It illustrates Python deployment via Docker (<http://docker.com>) containers and in the cloud.

### *Chapter 3*

Financial times series data is central to every algorithmic trading project. This chapter shows you how to retrieve financial data from different public data and proprietary data sources. It also demonstrates how to store financial time series data efficiently with Python.

#### Chapter 4

Vectorization is a powerful approach in numerical computation in general and for financial analytics in particular. This chapter introduces vectorization with NumPy and pandas and applies that approach to backtesting SMA-based, momentum, and mean-reversion strategies.

#### Chapter 5

This chapter is dedicated to generating market predictions by the use of machine learning and deep learning approaches. By mainly relying on past return observations as features, approaches are presented for predicting tomorrow's market direction by using such Python packages as tensorflow (<https://oreil.ly/B44Fb>) and scikit-learn (<http://scikit-learn.org/>).

#### Chapter 6

While vectorized backtesting has advantages when it comes to conciseness of code and performance, it's limited with regard to the representation of certain market features of trading strategies. On the other hand, event-based backtesting, technically implemented by the use of object oriented programming, allows for a rather granular and more realistic modeling of such features. This chapter presents and explains in detail a base class as well as two classes for the backtesting of long-only and long-short trading strategies.

#### Chapter 7

Needing to cope with real-time or streaming data is a reality even for the ambitious individual algorithmic trader. The tool of choice is socket programming, for which this chapter introduces ZeroMQ (<http://zeromq.org>) as a lightweight and scalable technology. The chapter also illustrates how to make use of Plotly (<http://plot.ly>) to create nice looking, interactive streaming plots. It also presents a wrapper class that simplifies the creation of such plots in cases where multiple data streams need to be visualized simultaneously (for example, in a dashboard-like manner).

#### Chapter 8

Oanda (<http://oanda.com>) is a foreign exchange (forex) and Contracts for Difference (CFD) trading platform offering a broad set of tradable instruments, such as those based on foreign exchange pairs, stock indices, commodities, or rates instruments (benchmark bonds). This chapter provides guidance on how to implement automated algorithmic trading strategies with Oanda, making use of the Python wrapper package tpqoa (<http://github.com/yhilpisch/tpqoa>).

#### Chapter 9

FXCM (<http://fxcm.co.uk>) is another forex and CFD trading platform that has recently released a modern RESTful API for algorithmic trading. Available instruments span multiple asset classes, such as forex, stock indices, or

commodities. A Python wrapper package that makes algorithmic trading based on Python code rather convenient and efficient is available (<http://fxcmpy.tpq.io>).

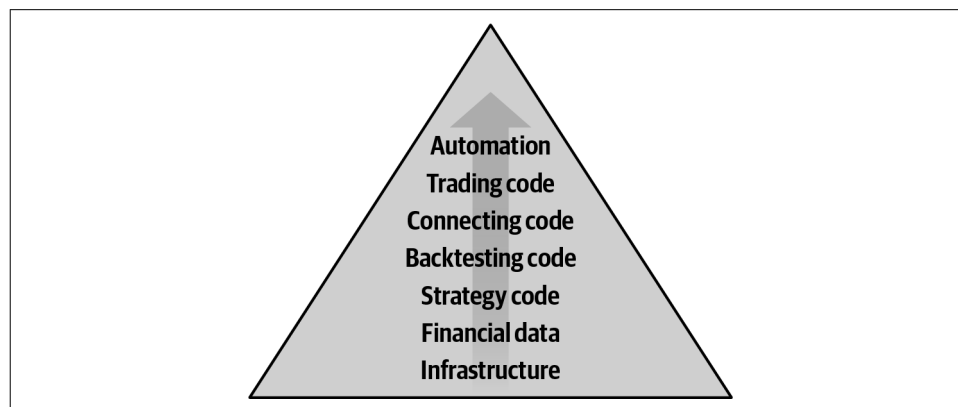
### *Chapter 10*

This chapter deals with capital management, risk analysis and management, as well as with typical tasks in the technical automation of algorithmic trading operations. It covers, for instance, the Kelly criterion for capital allocation and leverage in detail.

### *Appendix*

This appendix provides a concise introduction to the most important Python, NumPy, and pandas topics in the context of the material presented in the main chapters. It represents a starting point from which one can add to one's own Python knowledge over time.

Figure P-1 shows the layers related to algorithmic trading that the chapters cover from the bottom to the top. It necessarily starts with the Python infrastructure (Chapter 2), and adds financial data (Chapter 3), strategy, and vectorized backtesting code (Chapter 4 and Chapter 5). Until that point, data sets are used and manipulated as a whole. Event-based backtesting for the first time introduces the idea that data in the real world arrives incrementally (Chapter 6). It is the bridge that leads to the connecting code layer that covers socket communication and real-time data handling (Chapter 7). On top of that, trading platforms and their APIs are required to be able to place orders (Chapter 8 and Chapter 9). Finally, important aspects of automation and deployment are covered (Chapter 10). In that sense, the main chapters of the book relate to the layers as seen in Figure P-1, which provide a natural sequence for the topics to be covered.



*Figure P-1. The layers of Python for algorithmic trading*

## Who This Book Is For

This book is for students, academics, and practitioners alike who want to apply Python in the fascinating field of algorithmic trading. The book assumes that the reader has, at least on a fundamental level, background knowledge in both Python programming and in financial trading. For reference and review, the Appendix introduces important Python, NumPy, matplotlib, and pandas topics. The following are good references to get a sound understanding of the Python topics important for this book. Most readers will benefit from having at least access to Hilpisch (2018) for reference. With regard to the machine and deep learning approaches applied to algorithmic trading, Hilpisch (2020) provides a wealth of background information and a larger number of specific examples. Background information about Python as applied to finance, financial data science, and artificial intelligence can be found in these books:

Hilpisch, Yves. 2018. *Python for Finance: Mastering Data-Driven Finance*. 2nd ed. Sebastopol: O'Reilly.

———. 2020. *Artificial Intelligence in Finance: A Python-Based Guide*. Sebastopol: O'Reilly.

McKinney, Wes. 2017. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. 2nd ed. Sebastopol: O'Reilly.

Ramalho, Luciano. 2021. *Fluent Python: Clear, Concise, and Effective Programming*. 2nd ed. Sebastopol: O'Reilly.

VanderPlas, Jake. 2016. *Python Data Science Handbook: Essential Tools for Working with Data*. Sebastopol: O'Reilly.

Background information about algorithmic trading can be found, for instance, in these books:

Chan, Ernest. 2009. *Quantitative Trading: How to Build Your Own Algorithmic Trading Business*. Hoboken et al: John Wiley & Sons.

Chan, Ernest. 2013. *Algorithmic Trading: Winning Strategies and Their Rationale*. Hoboken et al: John Wiley & Sons.

Kissel, Robert. 2013. *The Science of Algorithmic Trading and Portfolio Management*. Amsterdam et al: Elsevier/Academic Press.

Narang, Rishi. 2013. *Inside the Black Box: A Simple Guide to Quantitative and High Frequency Trading*. Hoboken et al: John Wiley & Sons.

Enjoy your journey through the algorithmic trading world with Python and get in touch by emailing [pyalgo@tpq.io](mailto:pyalgo@tpq.io) if you have questions or comments.

## Conventions Used in This Book

The following typographical conventions are used in this book:

### *Italic*

Indicates new terms, URLs, email addresses, filenames, and file extensions.

### Constant width

Used for program listings, as well as within paragraphs, to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

### Constant width bold

Shows commands or other text that should be typed literally by the user.

### *Constant width italic*

Shows text that should be replaced with user-supplied values or by values determined by context.



This element signifies a tip or suggestion.



This element signifies a general note.



This element indicates a warning or caution.

## Using Code Examples

You can access and execute the code that accompanies the book on the Quant Platform at <https://py4at.pqp.io>, for which only a free registration is required.

If you have a technical question or a problem using the code examples, please send email to [bookquestions@oreilly.com](mailto:bookquestions@oreilly.com).

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it in your programs and documentation. You do not

need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but generally do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example, this book may be attributed as: "*Python for Algorithmic Trading* by Yves Hilpisch (O'Reilly). Copyright 2021 Yves Hilpisch, 978-1-492-05335-4."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at [permissions@oreilly.com](mailto:permissions@oreilly.com).

## O'Reilly Online Learning

**O'REILLY**® For more than 40 years, *O'Reilly Media* has provided technology and business training, knowledge, and insight to help companies succeed.

Our unique network of experts and innovators share their knowledge and expertise through books, articles, and our online learning platform. O'Reilly's online learning platform gives you on-demand access to live training courses, in-depth learning paths, interactive coding environments, and a vast collection of text and video from O'Reilly and 200+ other publishers. For more information, visit <http://oreilly.com>.

## How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.  
1005 Gravenstein Highway North  
Sebastopol, CA 95472  
800-998-9938 (in the United States or Canada)  
707-829-0515 (international or local)  
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at <https://oreil.ly/py4at>.



Email [bookquestions@oreilly.com](mailto:bookquestions@oreilly.com) to comment or ask technical questions about this book.

For news and information about our books and courses, visit <http://oreilly.com>.

Find us on Facebook: <http://facebook.com/oreilly>

Follow us on Twitter: <http://twitter.com/oreillymedia>

Watch us on YouTube: <http://youtube.com/oreillymedia>

## Acknowledgments

I want to thank the technical reviewers—Hugh Brown, McKlayne Marshall, Ramathan Ramakrishnamoorthy, and Prem Jebaseelan—who provided helpful comments that led to many improvements of the book’s content.

As usual, a special thank you goes to Michael Schwed, who supports me in all technical matters, simple and highly complex, with his broad and in-depth technology know-how.

Delegates of the Certificate Programs in Python for Computational Finance and Algorithmic Trading also helped improve this book. Their ongoing feedback has enabled me to weed out errors and mistakes and refine the code and notebooks used in our online training classes and now, finally, in this book.

I would also like to thank the whole team at O’Reilly Media—especially Michelle Smith, Michele Cronin, Victoria DeRose, and Danny Elfanbaum—for making it all happen and helping me refine the book in so many ways.

Of course, all remaining errors are mine alone.

Furthermore, I would also like to thank the team at Refinitiv—in particular, Jason Ramchandani—for providing ongoing support and access to financial data. The major data files used throughout the book and made available to the readers were received in one way or another from Refinitiv’s data APIs.

To my family with love. I dedicate this book to my father Adolf whose support for me and our family now spans almost five decades.

## About the Author

---

**Dr. Yves J. Hilpisch** is founder and CEO of The Python Quants (<http://tpq.io>), a group focusing on the use of open source technologies for financial data science, artificial intelligence, algorithmic trading, and computational finance. He is also founder and CEO of The AI Machine (<http://aimachine.io>), a company focused on AI-powered algorithmic trading via a proprietary strategy execution platform.

In addition to this book, he is the author of the following books:

- *Artificial Intelligence in Finance* (<https://aiif.tpq.io>) (O'Reilly, 2020)
- *Python for Finance* (<https://py4fi.tpq.io>) (2nd ed., O'Reilly, 2018)
- *Derivatives Analytics with Python* (<https://dawp.tpq.io>) (Wiley, 2015)
- *Listed Volatility and Variance Derivatives* (<https://lvvd.tpq.io>) (Wiley, 2017)

Yves is an adjunct professor of computational finance and lectures on algorithmic trading at the CQF Program (<http://cqf.com>). He is also the director of the first online training programs leading to university certificates in Python for Algorithmic Trading (<http://certificate.tpq.io>) and Python for Computational Finance (<http://compfinance.tpq.io>).

Yves wrote the financial analytics library DX Analytics (<http://dx-analytics.com>) and organizes meetups, conferences, and bootcamps about Python for quantitative finance and algorithmic trading in London, Frankfurt, Berlin, Paris, and New York. He has given keynote speeches at technology conferences in the United States, Europe, and Asia.

## Colophon

---

The animal on the cover of *Python for Algorithmic Trading* is a Common Barred Grass Snake (*Natrix helvetica*).

Many of the animals on O'Reilly covers are endangered; all of them are important to the world. To learn more about how you can help, go to [animals.oreilly.com](http://animals.oreilly.com).

The cover illustration is by Jose Marzan, based on a black and white engraving from *English Cyclopaedia Natural History*. The cover fonts are Gilroy Semibold and Guardian Sans. The text font is Adobe Minion Pro; the heading font is Adobe Myriad Condensed; and the code font is Dalton Maag's Ubuntu Mono.