

# 人工智慧投資分析



Tamkang  
Universit

淡江大學

## AI for Investment Analysis

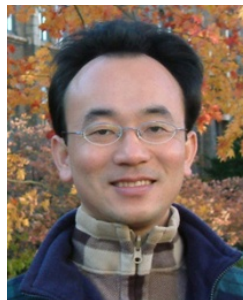
### 投資組合最佳化與程式交易

### (Portfolio Optimization and Algorithmic Trading)

1082AIIA10

MBA, IMTKU (M2399) (8409) (Spring 2020)

Wed 3, 4 (10:10-12:00) (B206)



Min-Yuh Day

戴敏育

Associate Professor

副教授

Dept. of Information Management, Tamkang University

淡江大學 資訊管理學系

<http://mail.tku.edu.tw/myday/>

2020-06-10



# 課程大綱 (Syllabus)

| 週次 (Week) | 日期 (Date)  | 內容 (Subject/Topics)   |
|-----------|------------|---|
| 1         | 2020/03/04 | 人工智慧投資分析課程介紹<br>(Course Orientation on AI for Investment Analysis)                  |
| 2         | 2020/03/11 | AI 金融科技: 金融服務創新應用<br>(AI in FinTech: Financial Services Innovation and Application) |
| 3         | 2020/03/18 | 機器人理財顧問與AI交談機器人<br>(Robo-Advisors and AI Chatbots)                                  |
| 4         | 2020/03/25 | 投資心理學與行為財務學<br>(Investing Psychology and Behavioral Finance)                        |
| 5         | 2020/04/01 | 財務金融事件研究法<br>(Event Studies in Finance)   |
| 6         | 2020/04/08 | 人工智慧投資分析個案研究 I<br>(Case Study on AI for Investment Analysis I)                      |

# 課程大綱 (Syllabus)

| 週次 (Week) | 日期 (Date)  | 內容 (Subject/Topics)  |
|-----------|------------|--|
| 7         | 2020/04/15 | Python AI投資分析基礎<br>(Foundations of AI Investment Analysis in Python)                                   |
| 8         | 2020/04/22 | Python Pandas 量化投資分析<br>(Quantitative Investing with Pandas in Python)                                 |
| 9         | 2020/04/29 | 期中報告 (Midterm Project Report)  |
| 10        | 2020/05/06 | Python Scikit-Learn 機器學習投資分析<br>(Machine Learning for Investment Analysis with Scikit-Learn in Python) |
| 11        | 2020/05/13 | TensorFlow 深度學習投資分析 I<br>(Deep Learning for Investment Analysis with TensorFlow I)                     |
| 12        | 2020/05/20 | TensorFlow 深度學習投資分析 II<br>(Deep Learning for Investment Analysis with TensorFlow II)                   |

# 課程大綱 (Syllabus)

| 週次 (Week) | 日期 (Date)  | 內容 (Subject/Topics)  |
|-----------|------------|--|
| 13        | 2020/05/27 | 人工智慧投資分析個案研究 II<br>(Case Study on Artificial Intelligence for Investment Analysis II)  |
| 14        | 2020/06/03 | TensorFlow 深度學習投資分析 III<br>(Deep Learning for Investment Analysis with TensorFlow III) |
| 15        | 2020/06/10 | 投資組合最佳化與程式交易<br>(Portfolio Optimization and Algorithmic Trading)                       |
| 16        | 2020/06/17 | 期末報告 I (Final Project Presentation I)  |
| 17        | 2020/06/24 | 期末報告 II (Final Project Presentation II)  |
| 18        | 2020/07/01 | 教師彈性補充教學   |

# Portfolio Optimization and Algorithmic Trading

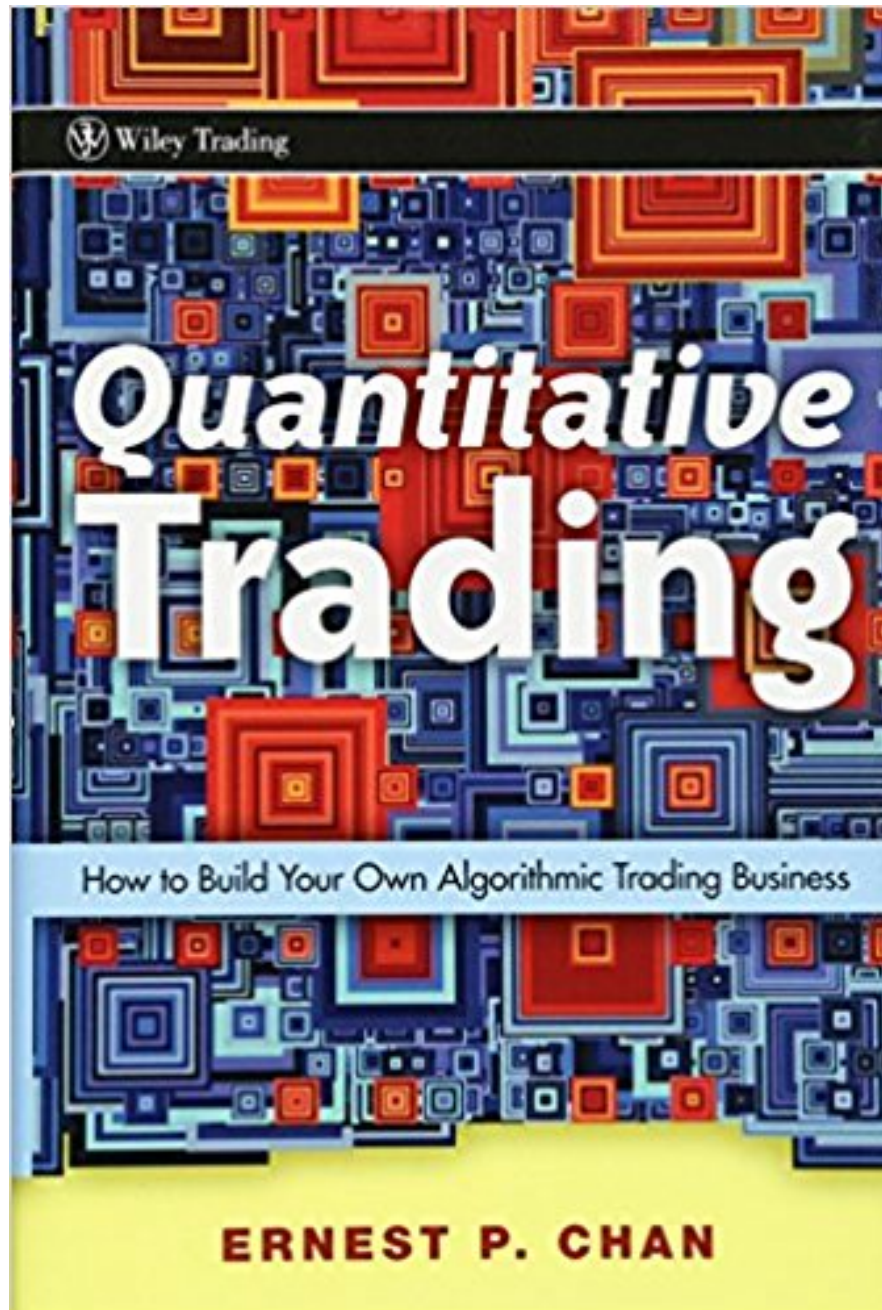
# Outline

- **Portfolio Optimization**
- **Algorithmic Trading**

# Portfolio Optimization

# Algorithmic Trading





Wiley Trading Series

# *Algorithmic* **TRADING**

WINNING STRATEGIES AND THEIR RATIONALE

+ website

ERNEST P. CHAN

WILEY

Wiley Trading Series

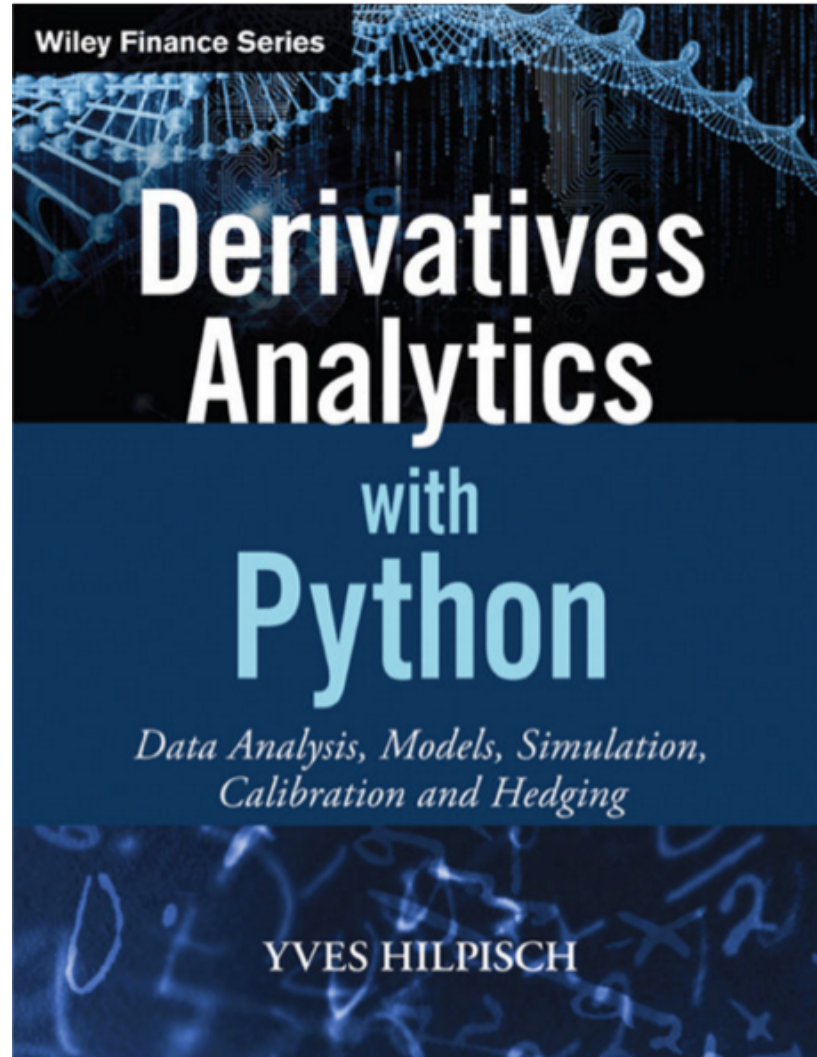
# MACHINE TRADING

DEPLOYING COMPUTER ALGORITHMS  
TO CONQUER THE MARKETS

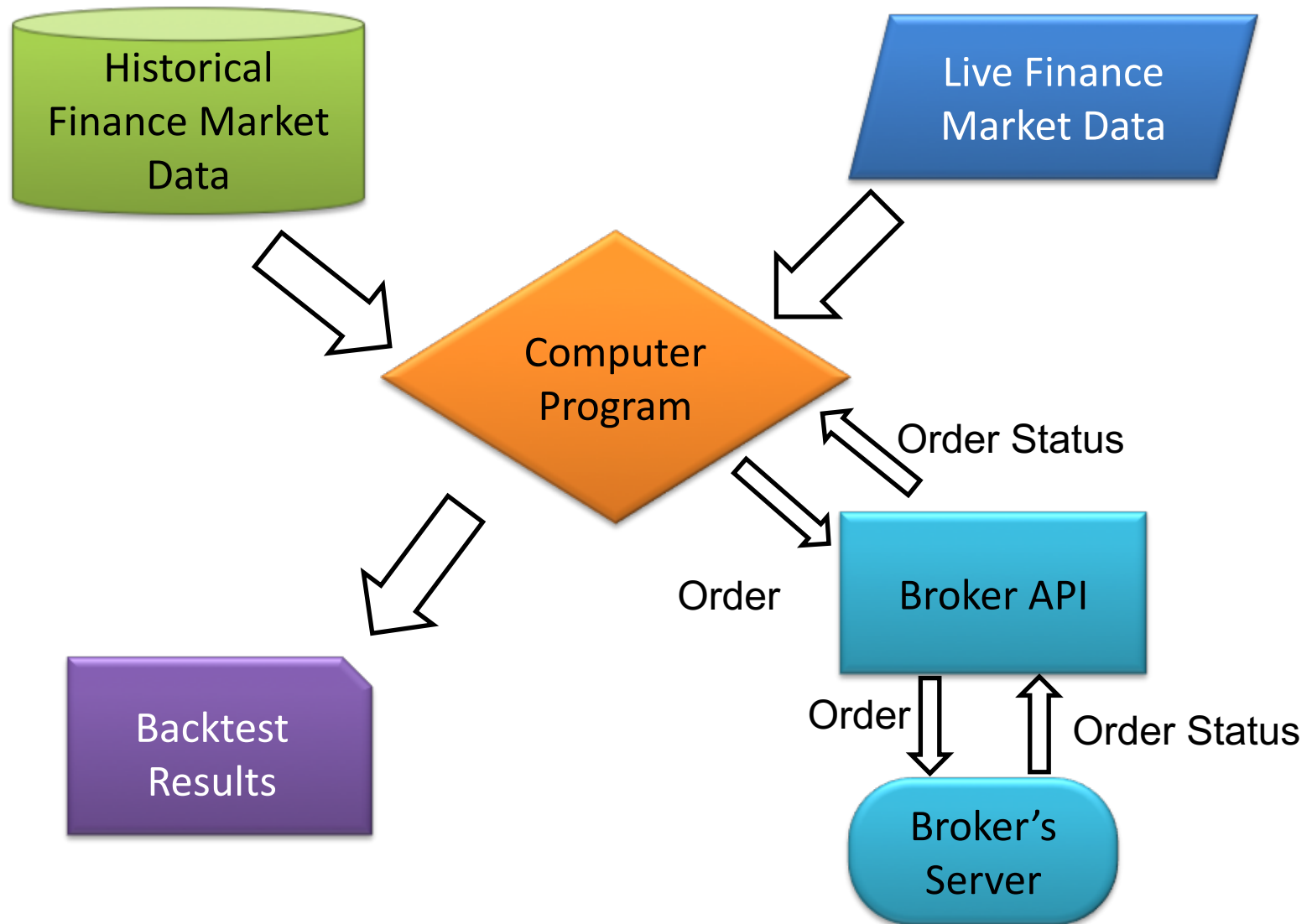
ERNEST P. CHAN

WILEY

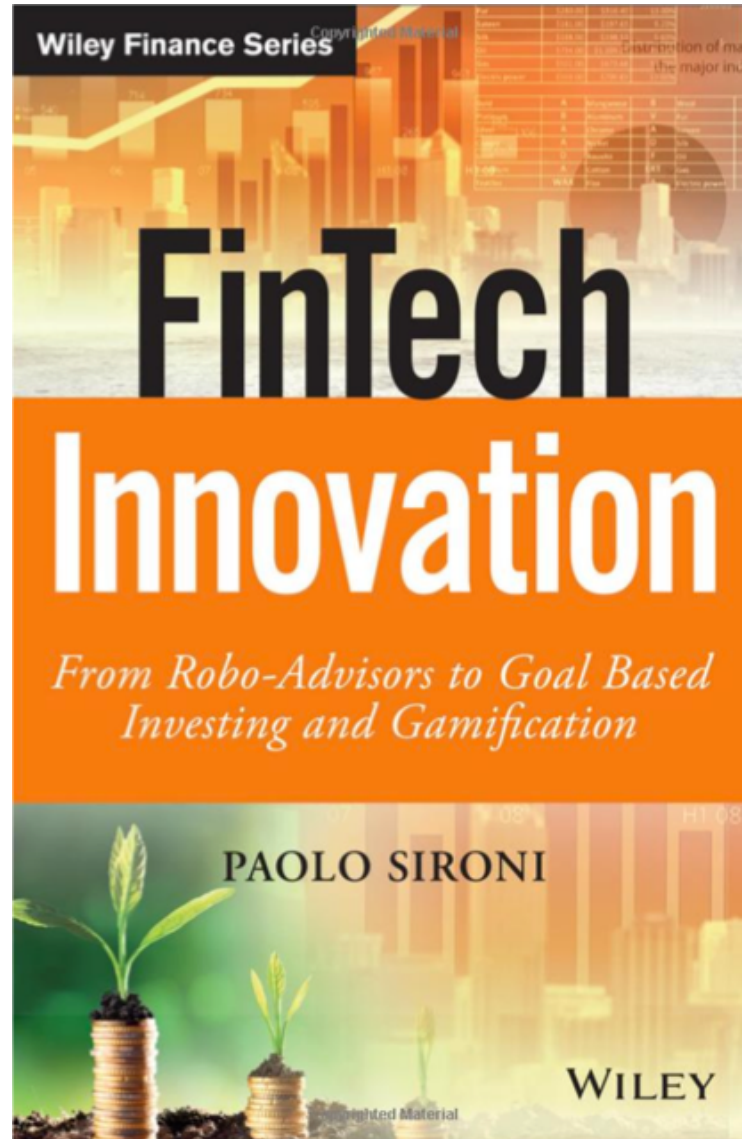
**Yves Hilpisch (2015),  
Derivatives Analytics with Python:  
Data Analysis, Models, Simulation, Calibration and Hedging, Wiley**



# Algorithmic Trading



**FinTech Innovation:  
From Robo-Advisors to Goal Based Investing and Gamification,  
Paolo Sironi, Wiley, 2016**



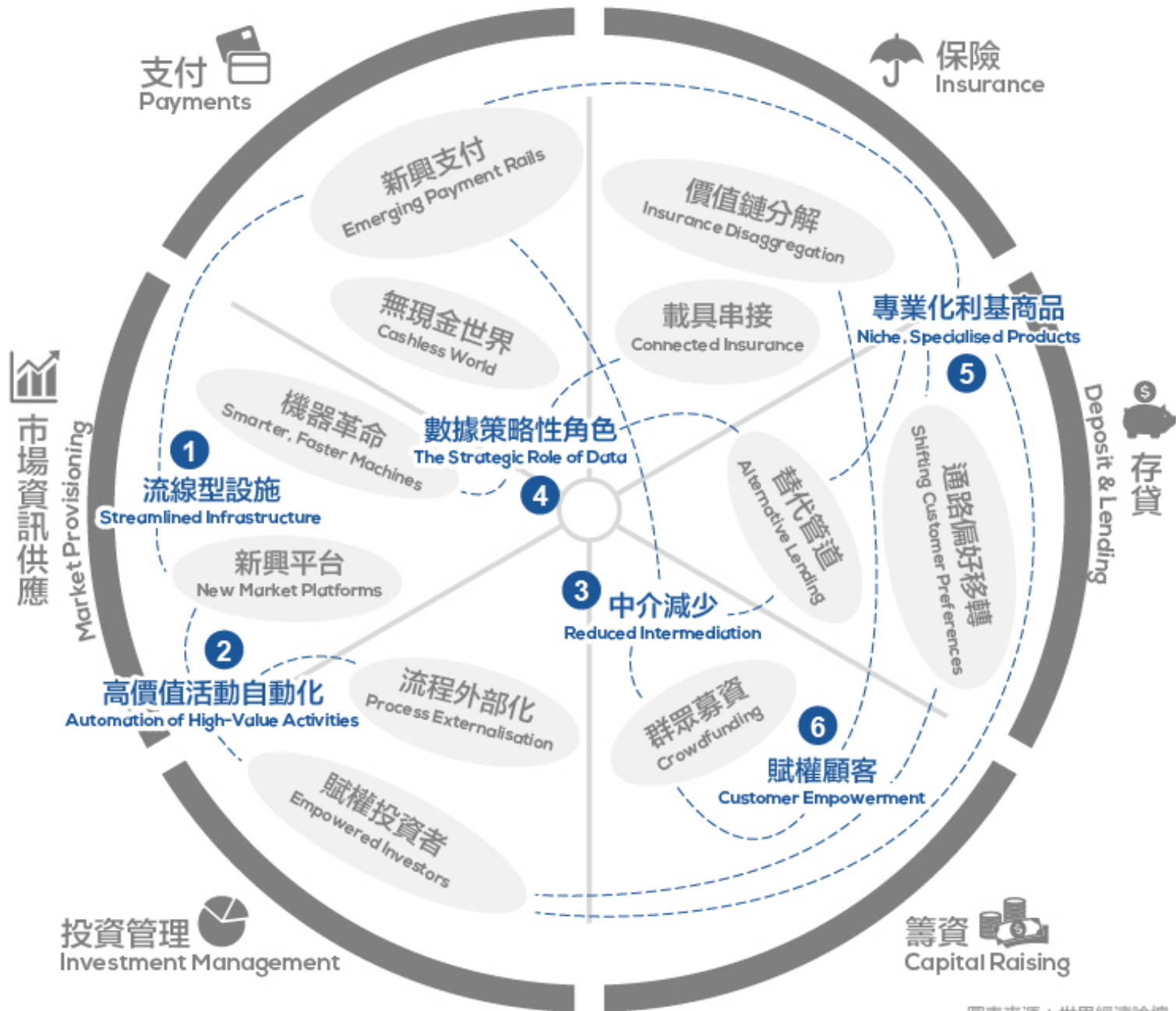
# FinTech: Financial Services Innovation



# FinTech: Financial Services Innovation

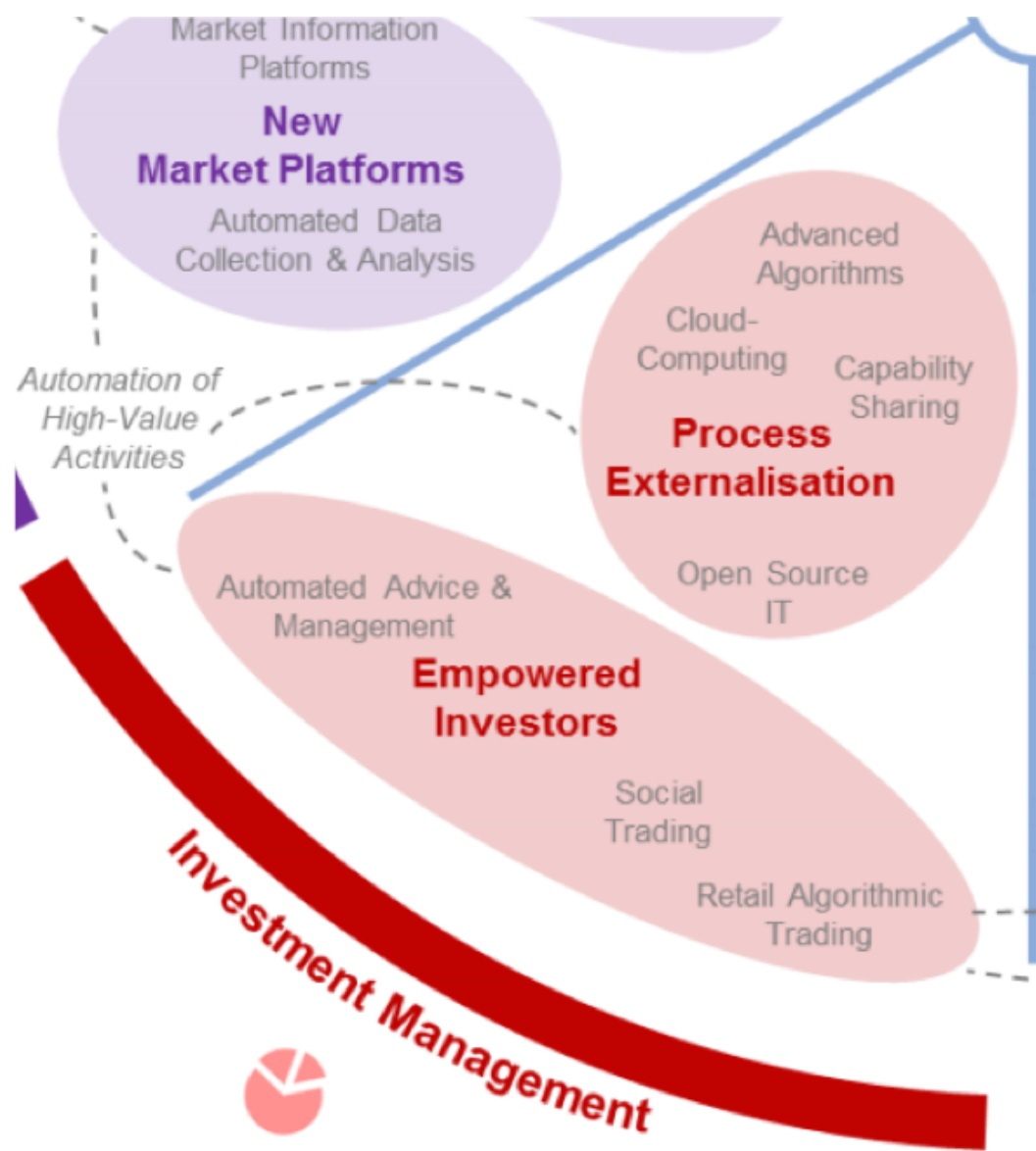
1. Payments
2. Insurance
3. Deposits & Lending
4. Capital Raising
- 5. Investment Management**
6. Market Provisioning





圖表來源：世界經濟論壇

# 5 FinTech: Investment Management



# 5 FinTech: Investment Management Empowered Investors Process Externalization

投資管理



創新

關鍵趨勢

**賦權投資者**  
Empowered  
Investors

社群交易、機器推薦與財富管理、零售演算法交易 (Retail Algorithmic Trading)

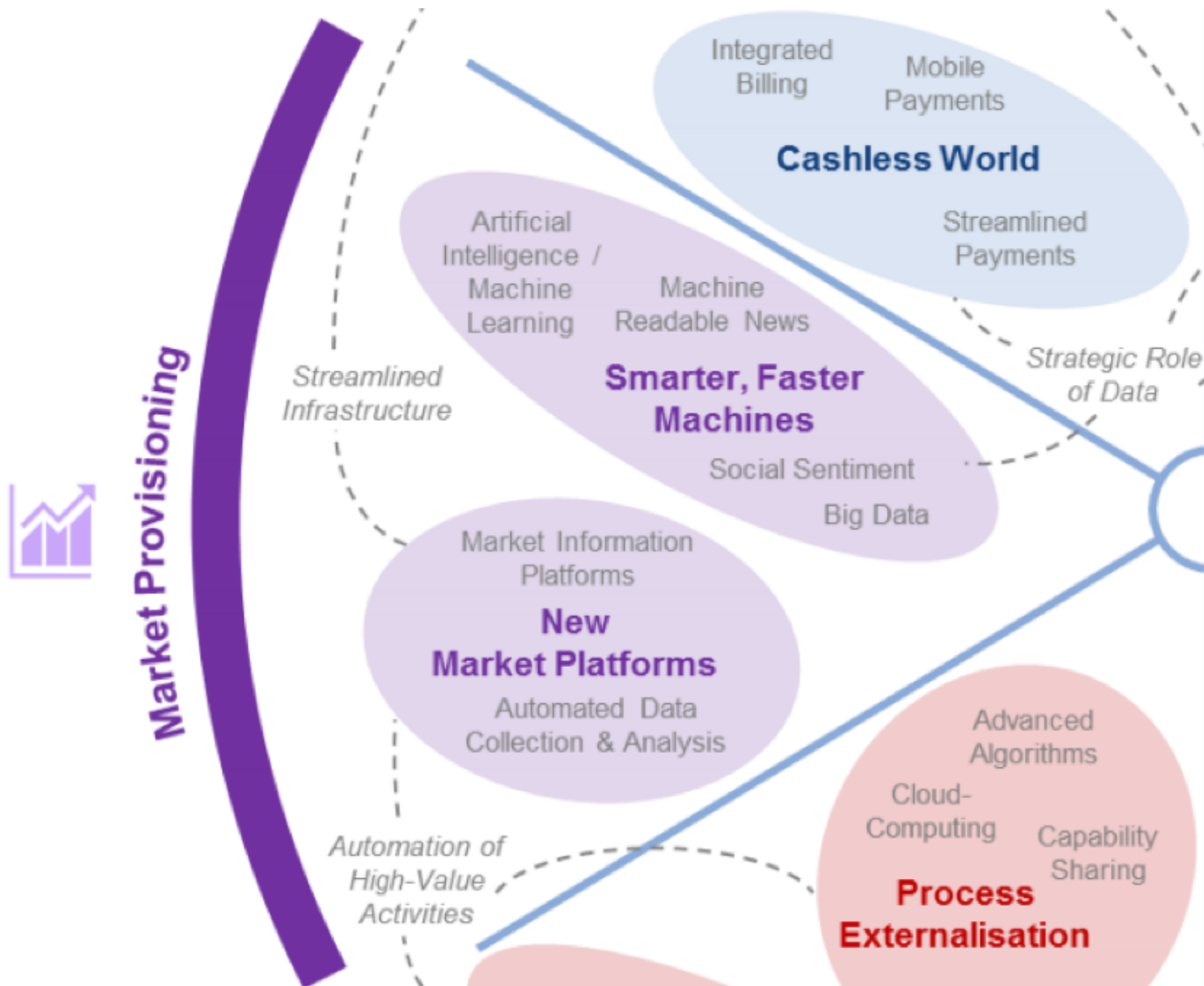
**流程外部化**  
Process  
Externalisation

流程即服務 (Process-as-a-Service, PaaS)、能力共享 (Capability Sharing)、進階分析、自然語言

圖表來源：Fugle團隊整理

# 6

# FinTech: Market Provisioning



# 6

## FinTech: Market Provisioning Smarter, Faster Machines New Market Platforms

市場資訊供應



創新

關鍵趨勢

**機器革命**  
Smarter, Faster  
Machines

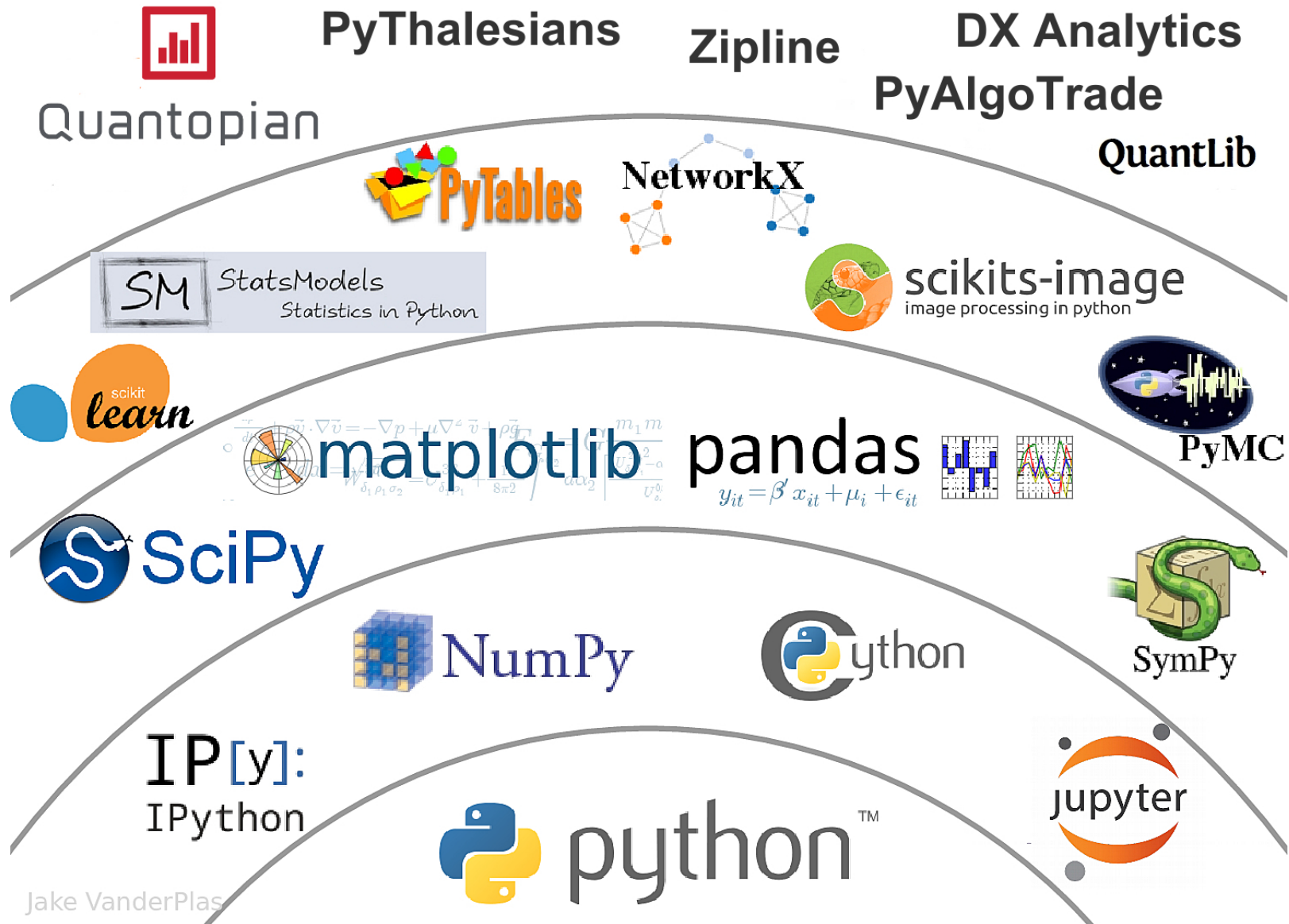
機器易用數據 (Machine Accessible Data)、人工智慧 / 機器學習、大數據

**新興平台**  
New Market  
Platforms

固定收益商品平台 ALGOMI、基金 / 組合型基金平台 NOVUS、私募 / 創投平台 BISON、未公發股權平台 LIQUITY、原物料商品與衍生性合約平台 ClauseMatch

圖表來源：Fugle團隊整理

# The Quant Finance PyData Stack



Jake VanderPlas

Source: [http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview\\_slides.ipynb#/5](http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb#/5)

# Zipline

a Pythonic

**Algorithmic Trading** Library

<http://www.zipline.io/>

# Zipline

- Zipline: Pythonic **algorithmic trading** library.
- Event-driven system
  - supports both **backtesting** and **live-trading**.
- Zipline is currently used in production as the backtesting and live-trading engine powering **Quantopian**
  - a free, community-centered, hosted platform for building and executing trading strategies.



# Quantopian



Get Funded

Research

Contest

Community

QuantCon

Learn

Help

Log In

Sign Up

## Become an Expert in Quant Finance

Quantopian provides free education, data, and tools so anyone can pursue quantitative finance. Select members license their algorithms and share in the profits.

Start Learning

## Community Achievements

All numbers are as of June 1, 2018

<https://www.quantopian.com/>

# Sign up for Quantopian

Sign up for Quantopian

Research and Develop Your Investment Ideas

Get started

I accept the [Terms Of Use](#) and [Privacy Policy](#).



[https://www.quantopian.com/users/sign\\_up](https://www.quantopian.com/users/sign_up)

# Quantopian

## Sample Mean Reversion Algorithm

Q

Capital

Research

Community

Learn

Help



< Sample Mean Reversion Algorithm

< All Backtests

Algorithm

Backtest

Settings: From 2015-03-27 to 2017-05-24 with \$1,000,000 initial capital

Live Trade Algorithm

Share Results



Calendar: US Equities

Status:  Backtest complete

Results Overview

Total Returns  
-13.4%

Benchmark Returns  
22.2%

Alpha  
-0.08

Beta  
0.13

Sharpe  
-0.82

Sortino  
-1.15

Volatility  
0.08

Max Drawdown  
-17.3%

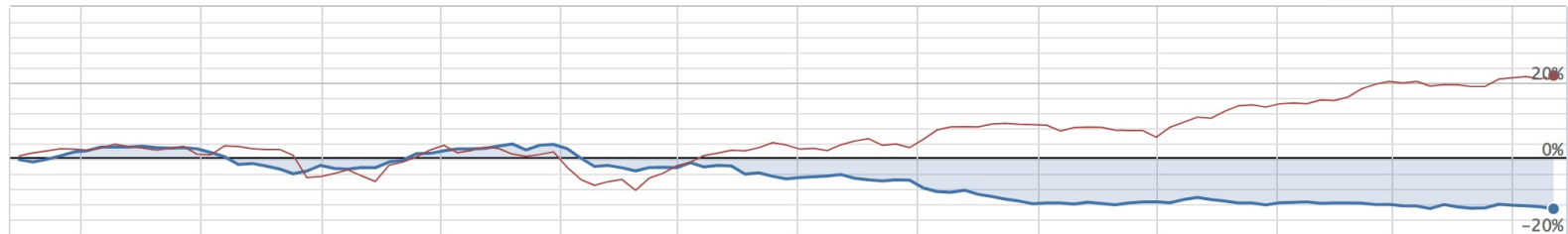
Cumulative performance: ■ Algorithm -13.24% ■ Benchmark (SPY) 21.9%

Week of May 22, 2017

Week

Month

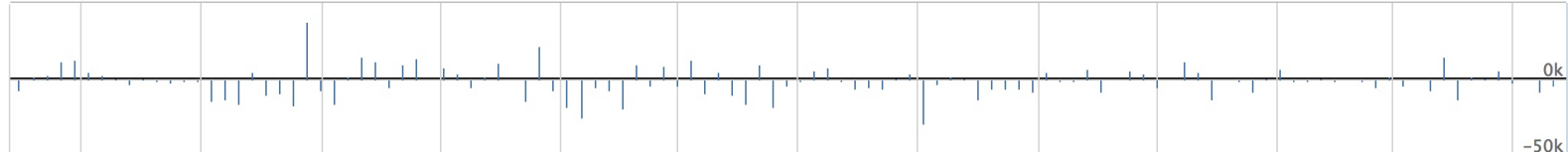
All



Custom data: ■ short\_count 150 ■ long\_count 150 ■ leverage 1



Weekly returns (\$3,518)



# Quantopian

## Sample Mean Reversion Algorithm

Q

Capital

Research

Community

Learn

Help



< Sample Mean Reversion Algorithm

< All Backtests

Algorithm

Backtest

Settings: From 2015-03-27 to 2017-05-24 with \$1,000,000 initial capital

Calendar: US Equities

Status:  Backtest complete

Live Trade Algorithm

Share Results



Results Overview

Total Returns  
-13.4%

Benchmark Returns  
22.2%

Alpha  
-0.08

Beta  
0.13

Sharpe  
-0.82

Sortino  
-1.15

Volatility  
0.08

Max Drawdown  
-17.3%

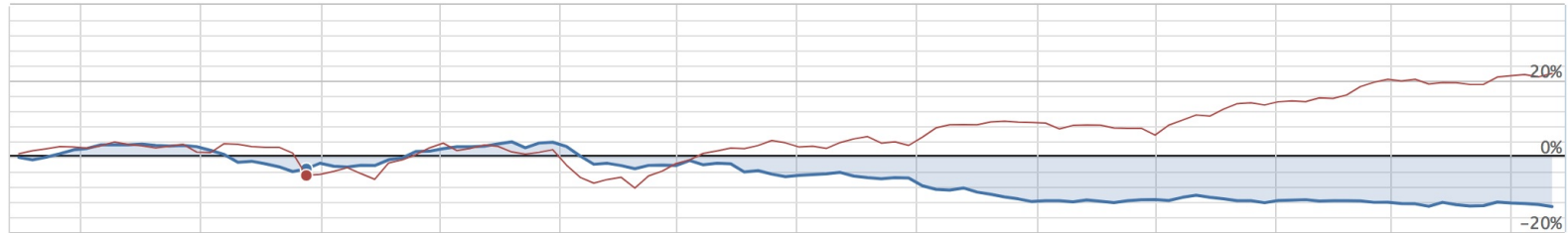
Cumulative performance: ■ Algorithm -3.3% ■ Benchmark (SPY) -5.02%

Week of Aug 24, 2015

Week

Month

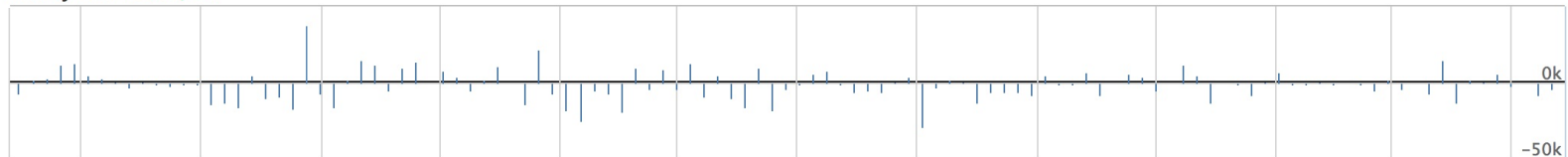
All



Custom data: ■ short\_count 149 ■ long\_count 149.2 ■ leverage 0.99



Weekly returns \$37,746



# Quantopian

## Sample Mean Reversion Algorithm

Sample Mean Reversion Algorithm

[◀ All Backtests](#)
[Algorithm](#)
[Backtest](#)

Settings: From 2015-03-27 to 2017-05-24 with \$1,000,000 initial capital

Live Trade Algorithm

Share Results



Calendar: US Equities

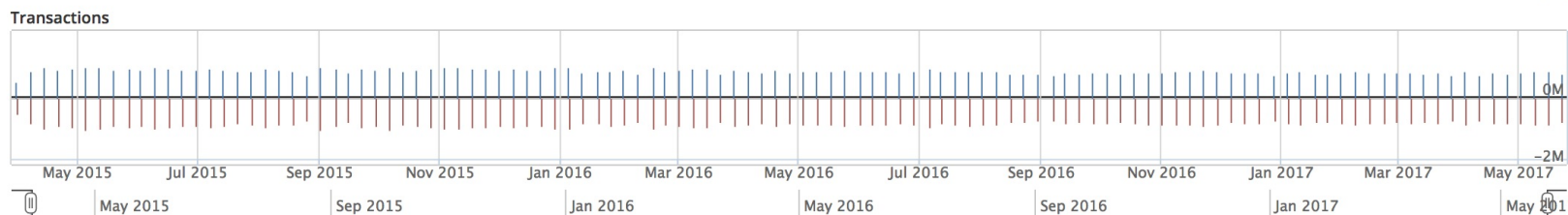
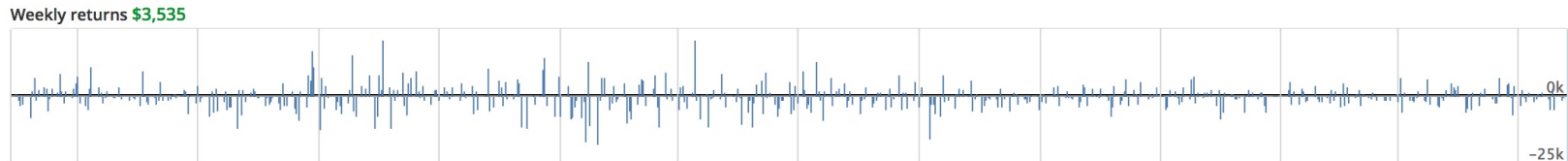
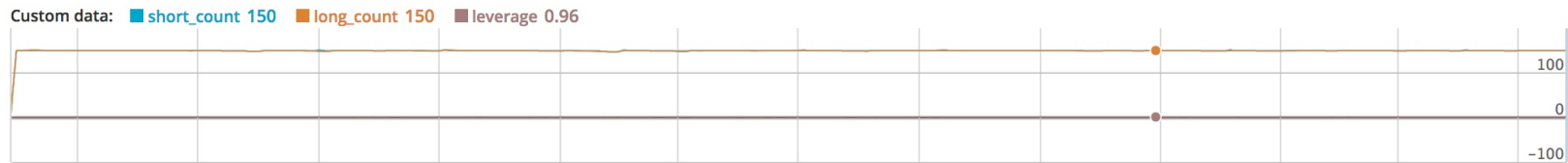
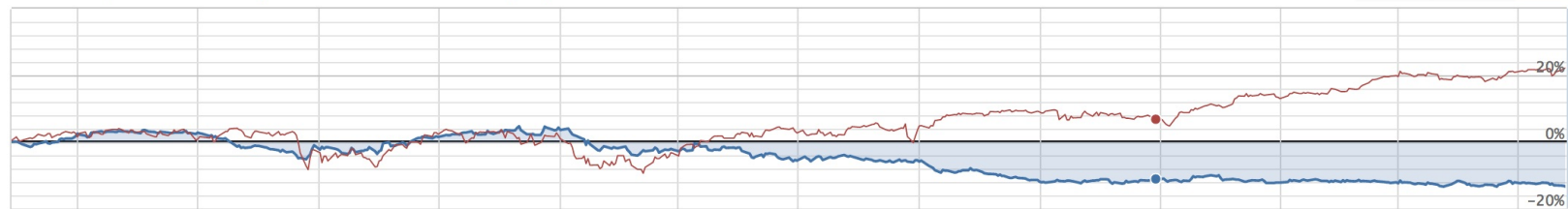
Status:  Backtest complete

### Results Overview

| Total Returns | Benchmark Returns | Alpha | Beta | Sharpe | Sortino | Volatility | Max Drawdown |
|---------------|-------------------|-------|------|--------|---------|------------|--------------|
| -13.4%        | 22.2%             | -0.08 | 0.13 | -0.82  | -1.15   | 0.08       | -17.3%       |

Cumulative performance: ■ Algorithm -11.1% ■ Benchmark (SPY) 6.8%

Oct 29, 2016



# Quantopian

## Sample Mean Reversion Algorithm

Sample Mean Reversion Algorithm

← All Backtests

Algorithm

Backtest

Settings: From 2015-03-27 to 2017-05-24 with \$1,000,000 initial capital

Calendar: US Equities

Status: ✓ Backtest complete

Live Trade Algorithm

Share Results



### Results Overview

#### Transaction Details

#### Daily Positions & Gains

#### Log Output

#### RISK METRICS

#### Returns

#### Benchmark Returns

#### Alpha

#### Beta

#### Sharpe

#### Sortino

#### Volatility

#### Benchmark Volatility

#### Max Drawdown

### Transaction Details

Expand All · Collapse All

Group by day

| Date                  | Asset | Transaction | Unit Price | Quantity | Position Value |
|-----------------------|-------|-------------|------------|----------|----------------|
| 2017-05-15 - 11:07 PM | PRAA  | SELL        | \$37.32    | -2       | (\$74.65)      |
| 2017-05-15 - 11:07 PM | PRTA  | SELL        | \$55.83    | -31      | (\$1,730.64)   |
| 2017-05-15 - 11:07 PM | PSTG  | BUY         | \$11.68    | 44       | \$513.96       |
| 2017-05-15 - 11:07 PM | PTCT  | SELL        | \$13.31    | -10      | (\$133.09)     |
| 2017-05-15 - 11:07 PM | QLYS  | BUY         | \$43.50    | 10       | \$435.03       |
| 2017-05-15 - 11:07 PM | RGR   | SELL        | \$64.07    | -2       | (\$128.14)     |
| 2017-05-15 - 11:07 PM | RRD   | BUY         | \$13.30    | 62       | \$824.60       |
| 2017-05-15 - 11:07 PM | RXN   | BUY         | \$23.45    | 9        | \$211.05       |
| 2017-05-15 - 11:07 PM | SUPN  | SELL        | \$33.50    | -12      | (\$401.98)     |
| 2017-05-15 - 11:07 PM | TCO   | SELL        | \$59.08    | -7       | (\$413.54)     |
| 2017-05-15 - 11:07 PM | TIVO  | BUY         | \$17.15    | 2        | \$34.30        |
| 2017-05-15 - 11:07 PM | TLRD  | BUY         | \$12.11    | 52       | \$629.77       |
| 2017-05-15 - 11:07 PM | TPC   | SELL        | \$28.20    | -5       | (\$140.99)     |
| 2017-05-15 - 11:07 PM | TROX  | SELL        | \$19.21    | -60      | (\$1,152.54)   |
| 2017-05-15 - 11:07 PM | TWNK  | BUY         | \$15.69    | 17       | \$266.75       |

# Quantopian

## Sample Mean Reversion Algorithm

Sample Mean Reversion Algorithm

Settings: From 2007-01-01 to 2016-12-31  
with \$1,000,000 initial capital  
Calendar: US Equities

All Backtests Algorithm Backtest

Settings: From 2007-01-01 to 2016-12-31 with \$1,000,000 initial capital  
Calendar: US Equities  
Status: ✓ Backtest complete

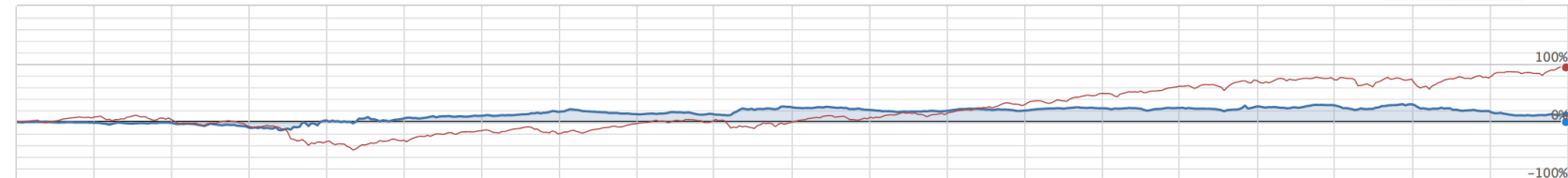
Live Trade Algorithm Share Results

- Results Overview
- Transaction Details
- Daily Positions & Gains
- Log Output
- RISK METRICS
- Returns
- Benchmark Returns
- Alpha
- Beta
- Sharpe
- Sortino
- Volatility
- Benchmark Volatility
- Max Drawdown

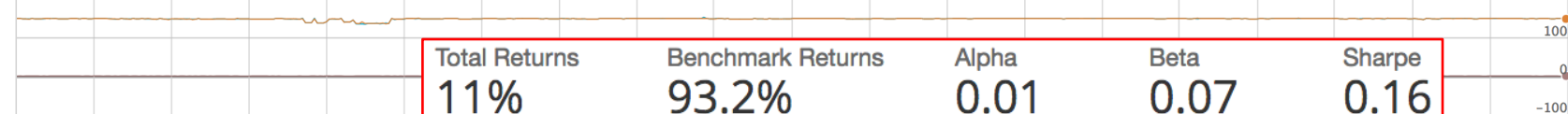
|               |                   |       |      |        |         |            |              |
|---------------|-------------------|-------|------|--------|---------|------------|--------------|
| Total Returns | Benchmark Returns | Alpha | Beta | Sharpe | Sortino | Volatility | Max Drawdown |
| 11%           | 93.2%             | 0.01  | 0.07 | 0.16   | 0.23    | 0.10       | -16.2%       |

Cumulative performance: Algorithm 10.97% Benchmark (SPY) 94.12%

Week of Dec 26, 2016 Week Month All

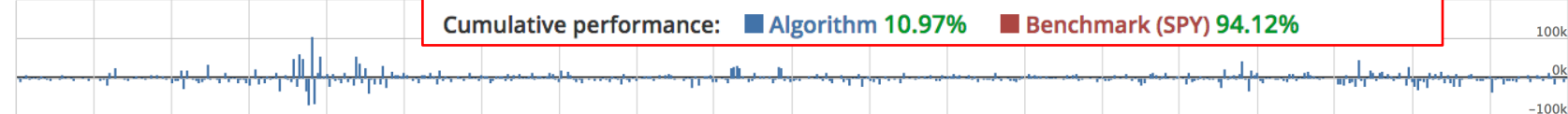


Custom data: short\_count 149 long\_count 149 leverage 0.98

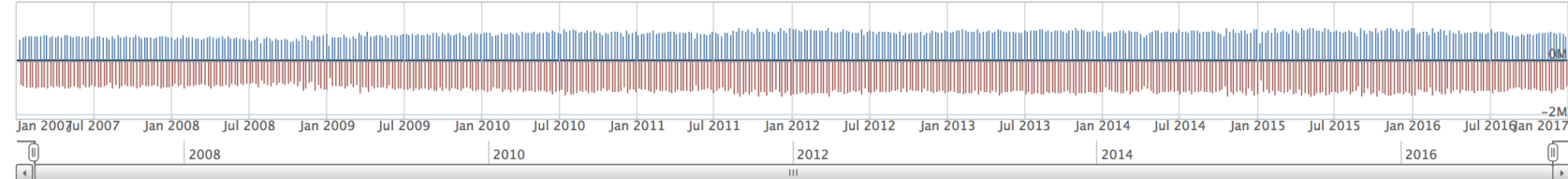


|   |                   |       |      |        |
|---|-------------------|-------|------|--------|
| Total Returns   | Benchmark Returns | Alpha | Beta | Sharpe |
| 11%   | 93.2%             | 0.01  | 0.07 | 0.16   |
| Cumulative performance: Algorithm 10.97% Benchmark (SPY) 94.12% |                   |       |      |        |

Weekly returns \$1,458



Transactions \$864,718 bought, (\$857,837) sold



# Quantopian

## Sample Mean Reversion Algorithm

Total Returns  
11%

Benchmark Returns  
93.2%

Alpha  
0.01

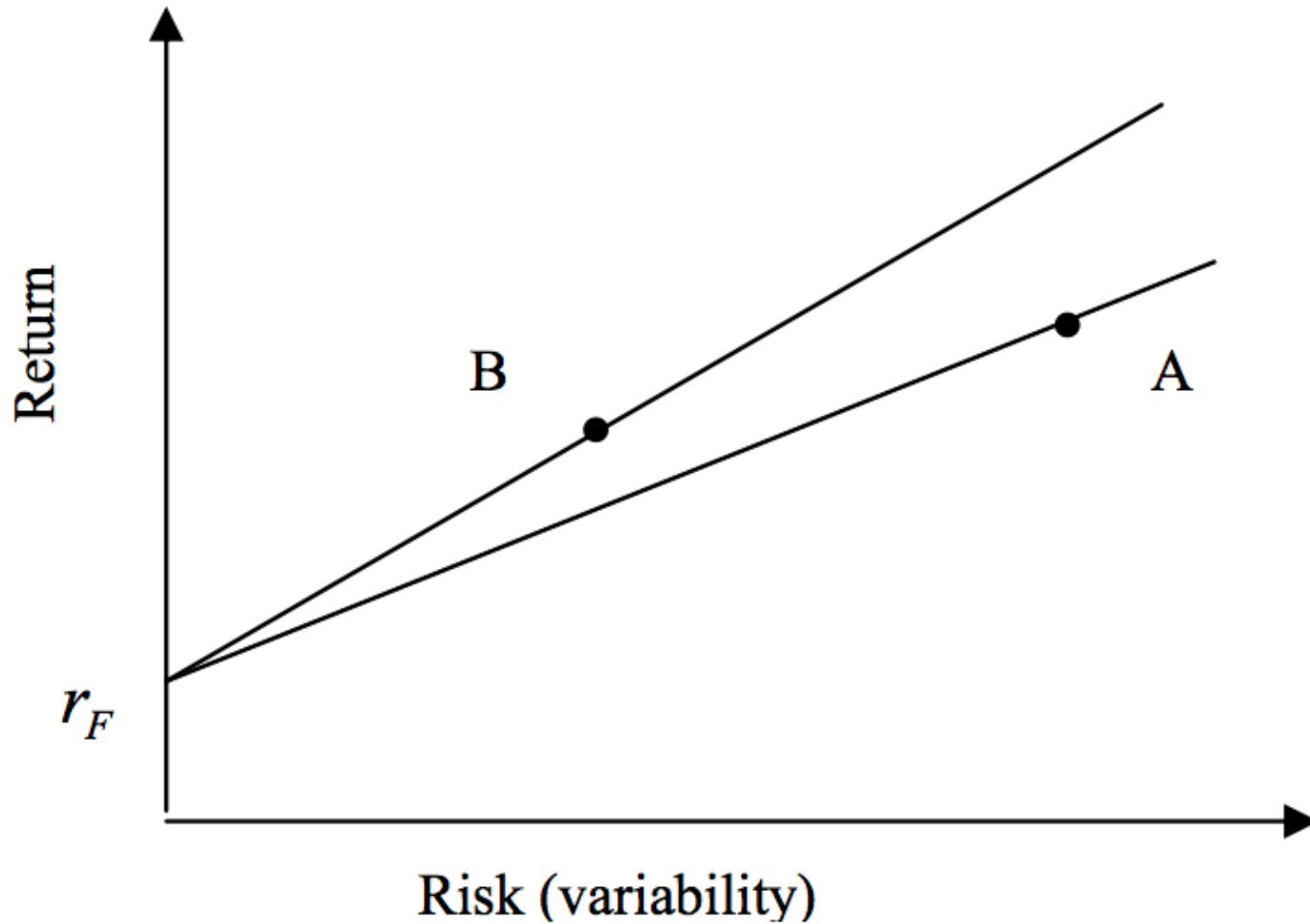
Beta  
0.07

Sharpe  
0.16

Cumulative performance: ■ Algorithm 10.97% ■ Benchmark (SPY) 94.12%



# Risk and Return



# Sharpe Ratio

## Sharpe Ratio

$$= \frac{\textit{Portfolio Return} - \textit{Risk Free Return}}{\textit{Portfolio Risk}}$$

# Sharpe Ratio

$$\text{Sharpe Ratio } SR = \frac{r_P - r_F}{\sigma_P}$$

Where

$r_P$  = portfolio return

$r_F$  = risk free rate

$\sigma_P$  = portfolio risk (variability, standard deviation of return)

# Sortino Ratio

$$\text{Sortino Ratio} = \frac{r_P - r_T}{\sigma_D}$$

Where

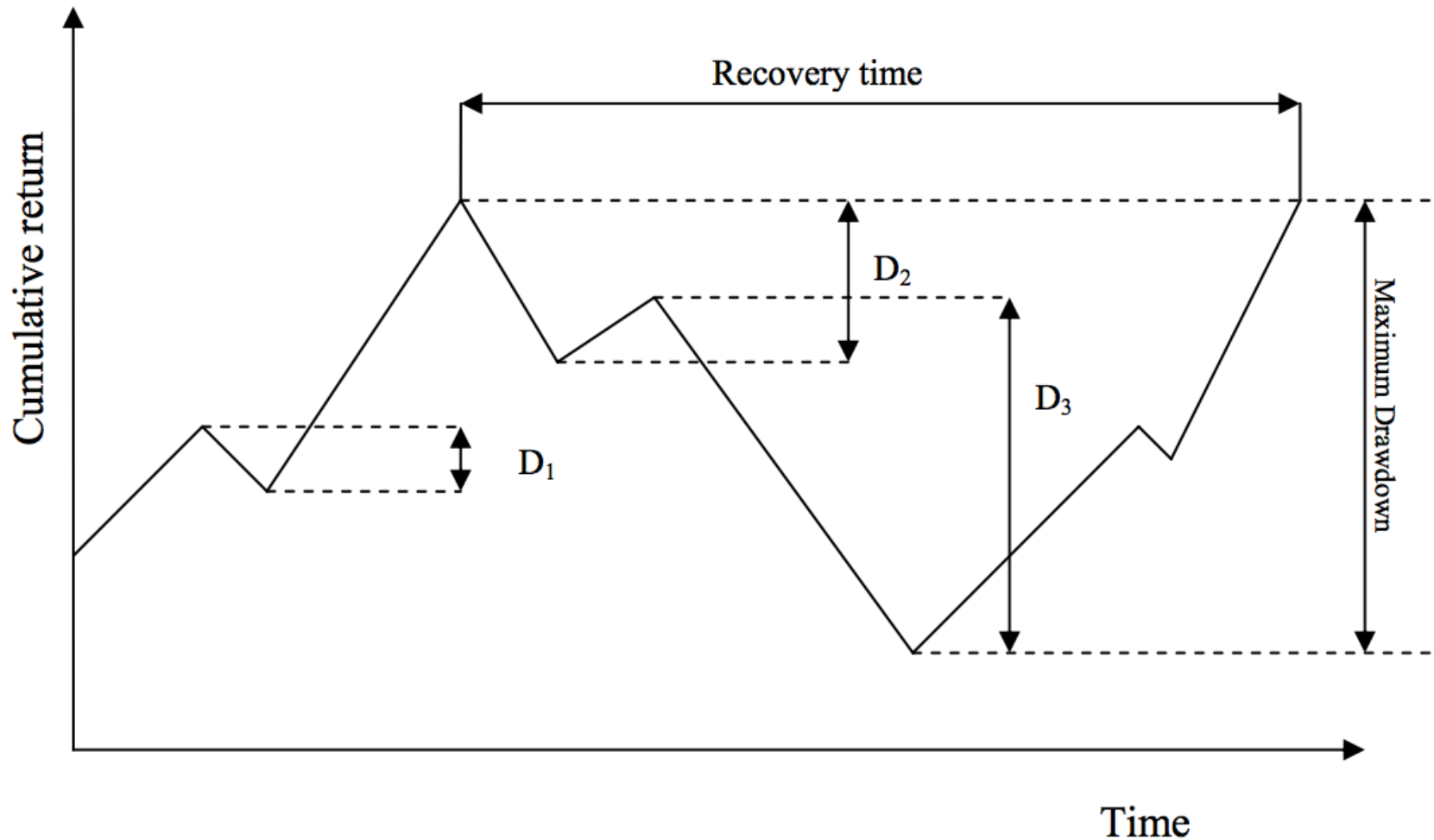
$r_P$  = portfolio return

$r_T$  = Minimum Target Return

$\sigma_D$  = Downside Risk

$$\text{Downside Risk } \sigma_D = \sqrt{\frac{\sum_{i=1}^n \min[(r_i - r_T), 0]^2}{n}}$$

# Max Drawdown



# Quantopian

## Sample Mean Reversion Algorithm

Save

Build Algorithm

Enter Contest

Collaborate

API Reference



Exit Fullscreen

```
1 """
2 This is a sample mean-reversion algorithm on Quantopian for you to test and adapt.
3 This example uses a dynamic stock selector, pipeline, to select stocks to trade.
4 It orders stocks from the top 1% of the previous day's dollar-volume (liquid
5 stocks).
6
7 Algorithm investment thesis:
8 Top-performing stocks from last week will do worse this week, and vice-versa.
9
10 Every Monday, we rank high dollar-volume stocks based on their previous 5 day returns.
11 We long the bottom 10% of stocks with the WORST returns over the past 5 days.
12 We short the top 10% of stocks with the BEST returns over the past 5 days.
13
14 This type of algorithm may be used in live trading and in the Quantopian Open.
15 """
16
17 # Import the libraries we will use here.
18 from quantopian.algorithm import attach_pipeline, pipeline_output
19 from quantopian.pipeline import Pipeline
20 from quantopian.pipeline.data.builtin import USEquityPricing
21 from quantopian.pipeline.factors import Returns
22 from quantopian.pipeline.filters.morningstar import Q1500US
23
24
25 def initialize(context):
26     """
27     Called once at the start of the program. Any one-time
28     startup logic goes here.
29     """
30     # Define context variables that can be accessed in other methods of
31     # the algorithm.
32     context.long_leverage = 0.5
33     context.short_leverage = -0.5
34     context.returns_lookback = 5
35
36     # Rebalance on the first trading day of each week at 11AM.
37     schedule_function(rebalance,
38                       date_rules.week_start(days_offset=0),
39                       time_rules.market_open(hours=1, minutes=30))
40
41     # Record tracking variables at the end of each day.
42     schedule_function(record_vars,
43                       date_rules.every_day()
```

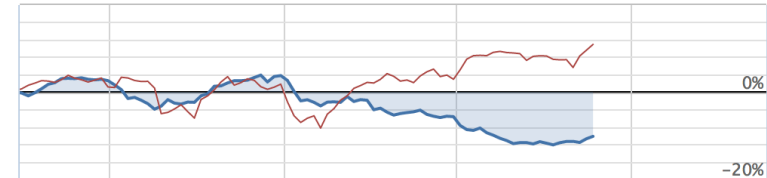
Backtesting

77.6%

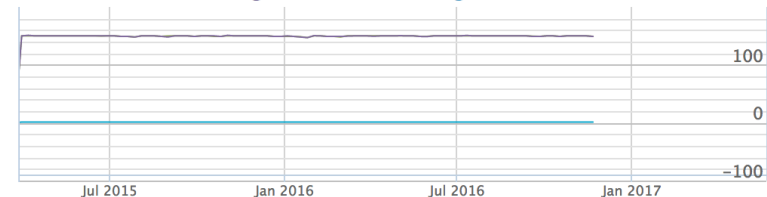
Cancel

| RETURNS | ALPHA | BETA | SHARPE | DRAWDOWN |
|---------|-------|------|--------|----------|
| -10%    | -0.07 | 0.12 | -0.72  | -16.4%   |

Algorithm 3.75% Benchmark (SPY) 1.77% Week of Dec 28, 2015



short\_count 149 long\_count 149 leverage 1



Logs

Runtime Errors

More ^

```
2015-03-30 23:00 rebalance:146 INFO This week's longs: ARCB, ACXM, ARW, AVT, AXP,
BBY, CAMP, CDE, CMCS_A, CY, ATGE, S, HL, HMSY, HPQ, IDCC, BIIB, IDTI, IMGX, IONS,
JCP, KLAC, LRCX, MAT, MBI, MGM, MTG, REGN, RGLD, SMTC, SONC, TDS, TDW, TER, TRMB,
WDC, WERN, WOR, YRCW, ZBRA, CREE, RIG, GMCR, HST, BCRX, KNX, SSYS, FOXA, VECO, MYGN,
SNDK, NVAX, SCCO, ANDE, HUBG, CRR, NBIX, GWR, OCN, CIEN, TIVO, POWI, WAC, BRGM, AMKR,
NVDA, BRCD, LPNT, JNPR, VIAY, FNSR, ON, SGMO, CYH, ARNA, KERX, MDGO, ARRY, UTII,
EXAS, SGMS, BTU, JOY, GME, LCI, BDSI, STX, CNX, APOL, ARRS, XPER, WLL, NRG, AGO,
ACAD, GNV, ALNY, DRH, HPY, IHS, CROX, ZIOP, SFLY, LDOS, ACHN, GSAT, AFSI, TWC, SMCI,
CZZ, DFT, CATM, SEM, PEB, PPC, FNGN, TSLA, SWFT, ZG, ZLTQ, GEVA, IMPV, HLSS, NSM,
PBYI, FRGI, WAGE, BLMN, RGLS, FLTJ, ENTA, CSTM, PTCT, GOGO, AGIO, COMM, NMBL, RARE,
TRUE, ANET, TMST, ECR, KITE, MIK, SYF, MBLY, KEYS, VA, LC, QRVO
```

# Quantopian

## Sample Mean Reversion Algorithm

Save Build Algorithm

Enter Contest

Collaborate

API Reference



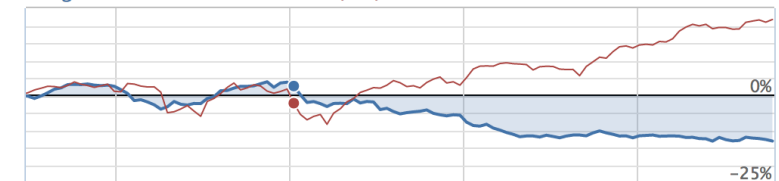
Exit Fullscreen

```
1 """
2 This is a sample mean-reversion algorithm on Quantopian for you to test and adapt.
3 This example uses a dynamic stock selector, pipeline, to select stocks to trade.
4 It orders stocks from the top 1% of the previous day's dollar-volume (liquid
5 stocks).
6
7 Algorithm investment thesis:
8 Top-performing stocks from last week will do worse this week, and vice-versa.
9
10 Every Monday, we rank high dollar-volume stocks based on their previous 5 day returns.
11 We long the bottom 10% of stocks with the WORST returns over the past 5 days.
12 We short the top 10% of stocks with the BEST returns over the past 5 days.
13
14 This type of algorithm may be used in live trading and in the Quantopian Open.
15 """
16
17 # Import the libraries we will use here.
18 from quantopian.algorithm import attach_pipeline, pipeline_output
19 from quantopian.pipeline import Pipeline
20 from quantopian.pipeline.data.builtin import USEquityPricing
21 from quantopian.pipeline.factors import Returns
22 from quantopian.pipeline.filters.morningstar import Q1500US
23
24
25 def initialize(context):
26     """
27     Called once at the start of the program. Any one-time
28     startup logic goes here.
29     """
30     # Define context variables that can be accessed in other methods of
31     # the algorithm.
32     context.long_leverage = 0.5
33     context.short_leverage = -0.5
34     context.returns_lookback = 5
35
36     # Rebalance on the first trading day of each week at 11AM.
37     schedule_function(rebalance,
38                     date_rules.week_start(days_offset=0),
39                     time_rules.market_open(hours=1, minutes=30))
40
41     # Record tracking variables at the end of each day.
42     schedule_function(record_vars,
43                     date_rules.every_day()
```

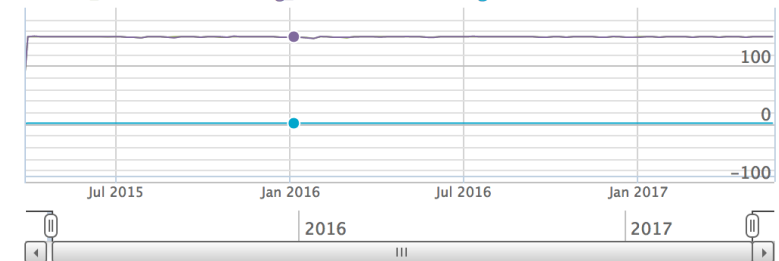
03/27/2015 to 05/23/2017 \$ 1000000 US Equities Run Full Backtest >

| RETURNS | ALPHA | BETA | SHARPE | DRAWDOWN |
|---------|-------|------|--------|----------|
| -13.2%  | -0.08 | 0.13 | -0.81  | -17.3%   |

Algorithm 2.64% Benchmark (SPY) -2.28% Week of Jan 4, 2016



short\_count 149.6 long\_count 150 leverage 0.97



Logs

Runtime Errors

More ^

```
2015-03-30 23:00 rebalance:146 INFO This week's longs: ARCB, ACXM, ARW, AVT, AXP,
BBY, CAMP, CDE, CMCS_A, CY, ATGE, S, HL, HMSY, HPQ, IDCC, BIIB, IDTI, IMGN, IONS,
JCP, KLAC, LRCX, MAT, MBI, MGM, MTG, REGN, RGLD, SMT, SONC, TDS, TDW, TER, TRMB,
WDC, WERN, WOR, YRCW, ZBRA, CREE, RIG, GMC, HST, BCRX, KNX, SSYS, FOXA, VECO, MYGN,
SNDK, NVAX, SCCO, ANDE, HUBG, CR, NBIX, GWR, OCN, CIEN, TIVO, POWI, WAC, BRM, AMKR,
NVDA, BRCD, LPNT, JNPR, VIAV, FNSR, ON, SGM, CYH, ARNA, KERX, MDCO, ARRY, UTII,
EXAS, SGMS, BTU, JOY, GME, LCI, BDSI, STX, CNX, APOL, ARRS, XPER, WLL, NRG, AGO,
ACAD, GNW, ALNY, DRH, HPY, IHS, CROX, ZIOP, SFLY, LDOS, ACHN, GSAT, AFSI, TWC, SMCI,
CZZ, DFT, CATM, SEM, PEB, PPC, FNGN, TSLA, SWFT, ZG, ZLTQ, GEVA, IMPV, HLSS, NSM,
PBYI, FRGI, WAGE, BLMN, RGLS, FLT, ENT, CST, PTCT, GOGO, AGIO, COMM, NMBL, RARE,
TRUE, ANET, TMST, ECR, KITE, MIK, SYF, MBL, KEYS, VA, LC, QRO
```

# Writing and Backtesting an Algorithm on Quantopian



# What is a Trading Algorithm?

**On Quantopian,  
a trading algorithm  
is a Python program  
that defines two special functions:  
`initialize()` and `handle_data()`**

# An example of an algorithm that allocates 100% of its portfolio in AAPL

```
def initialize(context):  
    # Reference to AAPL  
    context.aapl = sid(24)  
  
def handle_data(context, data):  
    # Position 100% of our portfolio to be long in AAPL  
    order_target_percent(context.aapl, 1.00)
```

# Moving Average

```
def initialize(context):
    context.security = symbol('AAPL')
    schedule_function(myfunc, date_rules.every_day(), time_rules.market_open(minutes = 15))

def handle_data(context, data):
    MovingAvg1 = data[context.security].mavg(20)
    MovingAvg2 = data[context.security].mavg(60)

    current_positions = context.portfolio.positions[symbol('AAPL')].amount

    if (MovingAvg1 > MovingAvg2) and current_positions == 0:
        order_target_percent(context.security, 0.25)

    elif (MovingAvg1 < MovingAvg2) and current_positions != 0:
        order_target(context.security, 0)
```

# Quantopian

## WSJ Example Algorithm

Q

Capital

Research

Community

Learn

Help



< Cloned from "WSJ Example Algorithm"

< All Backtests

Algorithm

Backtest

Settings: From 2009-01-01 to 2011-01-01 with \$1,000,000 initial capital

Live Trade Algorithm

Share Results



Calendar: US Equities

Status:  Backtest complete

### Results Overview

Transaction Details

Daily Positions & Gains

Log Output

### RISK METRICS

Returns

Benchmark Returns

Alpha

Beta

Sharpe

Sortino

Volatility

Benchmark Volatility

Total Returns  
46.3%

Benchmark Returns  
45.4%

Alpha  
0.16

Beta  
0.16

Sharpe  
1.82

Sortino  
2.89

Volatility  
0.11

Max Drawdown  
-12.1%

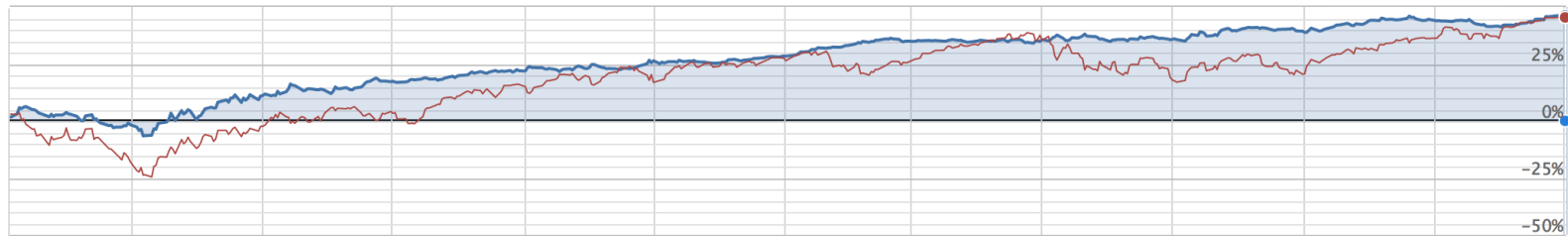
Cumulative performance: ■ Algorithm 46.3% ■ Benchmark (SPY) 45.4%

Jan 1, 2011

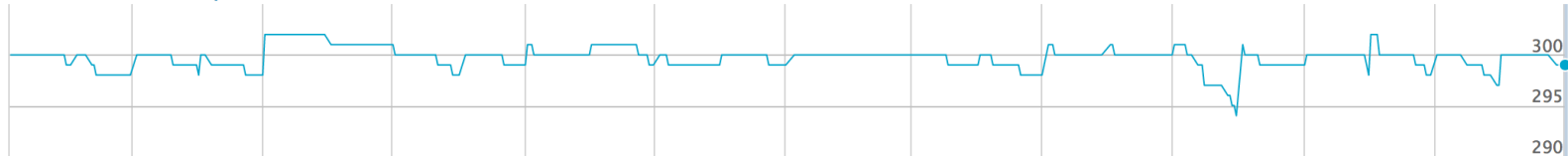
Week

Month

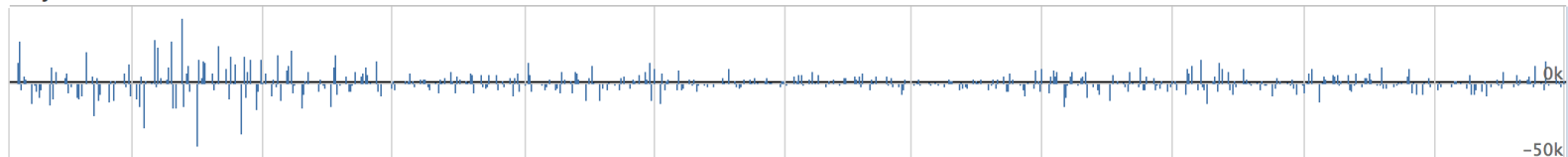
All



Custom data: ■ num\_positions 299

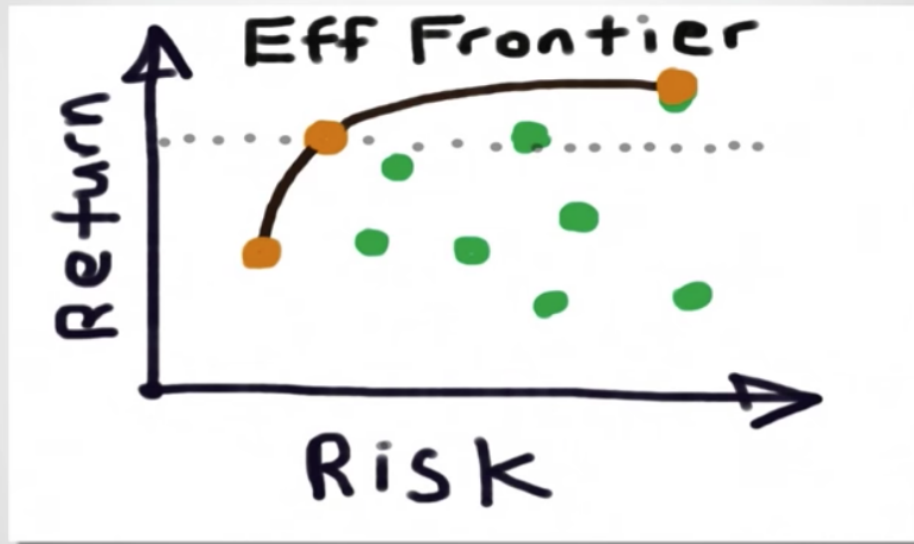


Daily returns \$963



# Investment Science: Portfolio Optimization

➔ How to Combine Them?



Lucena Research LLC | www.lucenaresearch.com | 404-907-1702

Three important portfolios on the Efficient Frontier

Full screen (f)

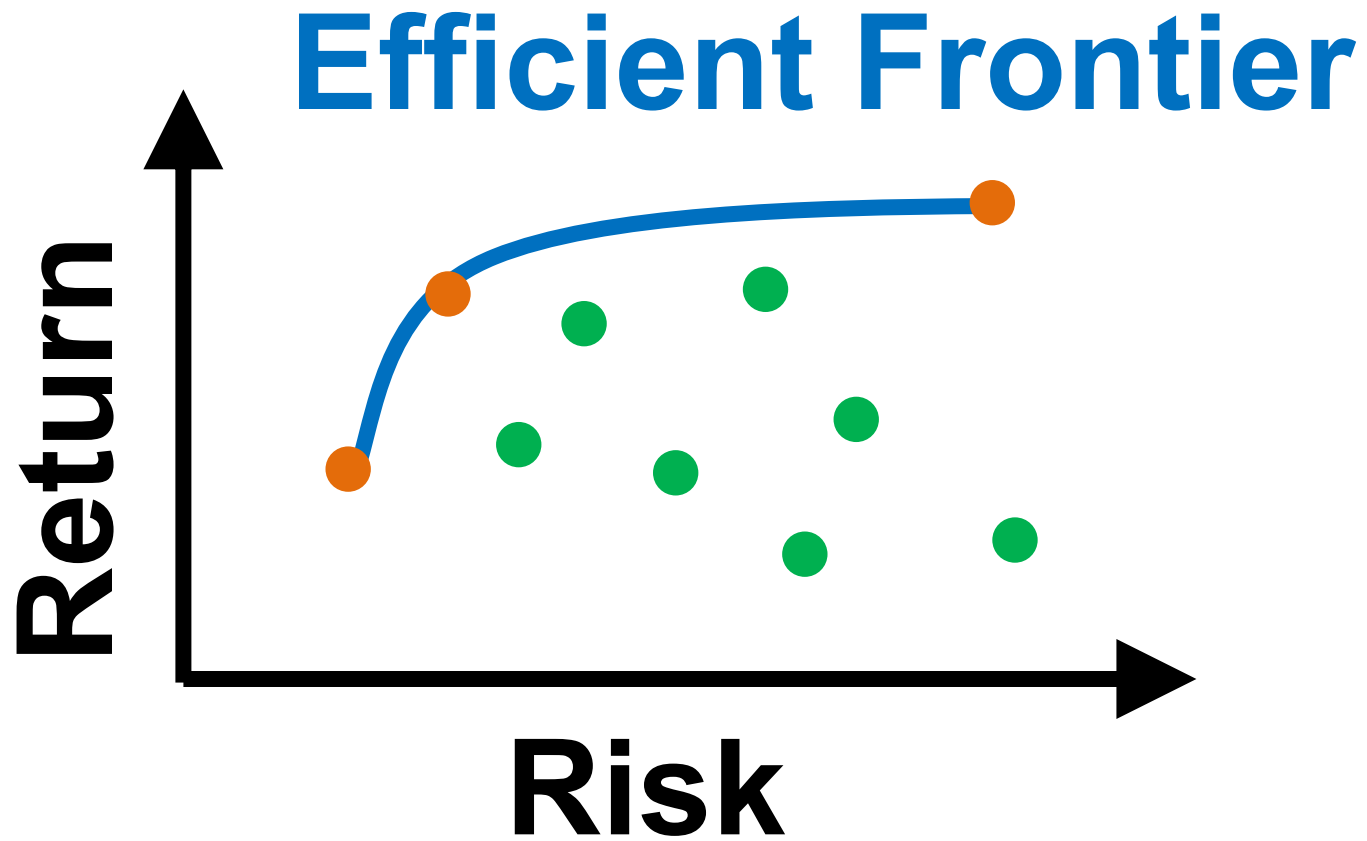
▶ ▶ 🔊 11:42 / 18:08



Source: Tucker Balch (2012), Investment Science: Portfolio Optimization,  
<https://www.youtube.com/watch?v=5qbMhXXq0vI>

# Portfolio Optimization

## Efficient Frontier



# Portfolio Optimization and Algorithmic Trading

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings Profile

RAM Disk Editing

- Table of contents
- Python101
  - Python File Input / Output
  - OS, IO, files, and Google Drive
  - Python Programming
  - Pythong String and Text
  - Python Numpy
  - Python Pandas
  - Deep Learning for Financial Time Series Forecasting
  - Portfolio Optimization and Algorithmic Trading
    - Investment Portfolio Optimisation with Python
    - Efficient Frontier Portfolio Optimisation in Python**
    - Investment Portfolio Optimization
  - Text Analytics and Natural Language Processing (NLP)
    - Python for Natural Language Processing
      - spaCy Chinese Model
    - Open Chinese Convert (OpenCC, 開放中文轉換)
    - Jieba 結巴中文分詞
    - Natural Language Toolkit (NLTK)
    - Stanza: A Python NLP Library for Many Human Languages

```
Annualised Return: 0.19
Annualised Volatility: 0.18
```

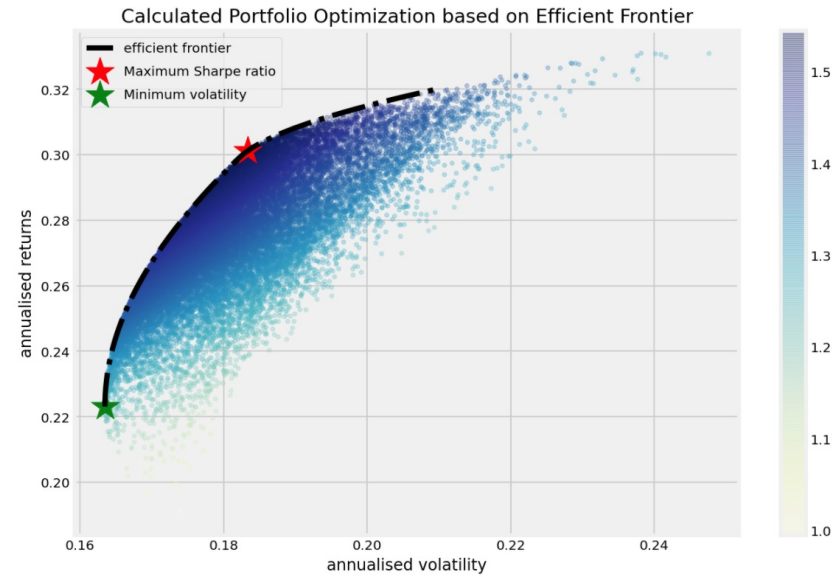
```
      AAPL  AMZN  FB  GOOGL
allocation 44.67 29.05 26.28 0.0
```

-----

Minimum Volatility Portfolio Allocation

```
Annualised Return: 0.22
Annualised Volatility: 0.16
```

```
      AAPL  AMZN  FB  GOOGL
allocation 34.02 0.73 6.98 58.26
```

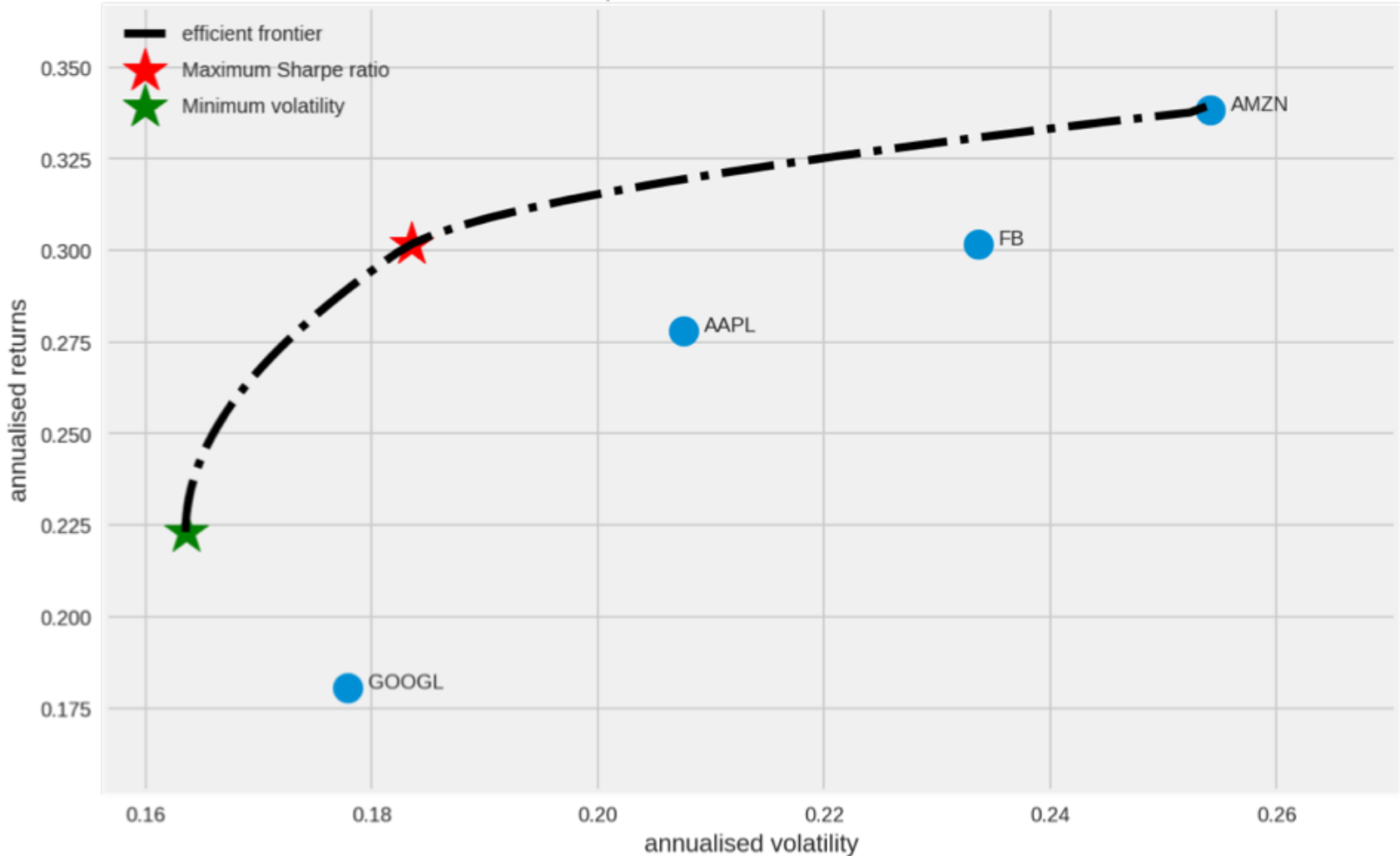


<https://tinyurl.com/imtkupython101>

# Portfolio Optimization

## Efficient Frontier

Portfolio Optimization with Individual Stocks





## 优矿，您的私人量化平台

打破金融量化的壁垒，为量化研究者提供媲美华尔街专业机构的研究装备

新手指引

开始研究

[了解专业版>>>](#)



### 产品动态

持续更新，为你提供更好的研究体验

- 2017-5-16 客户端优化信号库因子分类，增强按照因子分类查看因子表现；  
风险模型数据接口支持调用截面数据；  
客户端知识库界面改版。

### 热门讨论

策略/研究方法/代码分享，一网打尽

- 风险模型应用之归因分析：以“长信量化先锋混合”...  jiang.wei 2017-05-24
- 克隆！测算近期的最强因子  投资七日谈 2017-04-11
- 事件驱动策略研究2——员工持股计划，近年来alp... Paul333 2017-04-27

# UQER

## 海量金融大数据

高质量的海量金融数据支撑，轻松实现大数据时代的交易策略



## 云端平台，高效研究，极速回测

稳定、安全、高可扩展的云平台，零门槛获得华尔街专业级别量化研究装备



# UQER



模拟交易，赢取基金管理权

---

一键实盘模拟，云端托管，更有机会赢取500万实盘资金管理收益

# JoinQuant

JoinQuant 聚宽 [了解企业版](#)

[首页](#) [策略擂台](#) [投资研究](#) [我的策略](#) [我的交易](#) [数据](#) [帮助](#) [量化课堂](#) [社区](#) [登录](#) | [注册](#)

## 十行代码，玩转聚宽



JOINQUANT  
与7万+宽客一起  
玩转量化投资

免费试用

立即登录

## 策略广场

赢率季胜季

鏖战先觉者

模拟实盘

基于SVM的机器学习策略

走得很慢的海龟

策略回测

稳增长爆组合

阴吹思婷

模拟实盘

## 策略广场

### 赢率季胜季

模拟实盘

鏖战先觉者

■ 策略收益 ■ 基准收益



年化收益 **305.71%** | 最大回撤 **13.31%** | 初始资金 **¥ 50000**

已有**540**人订阅

免费订阅

### 基于SVM的机器学习策略

策略回测

走得很慢的海龟

■ 策略收益 ■ 基准收益



年化收益 **10.05%** | 最大回撤 **20.49%** | 初始资金 **¥ 1000000**

已有**633**人获取源码

获取源码

### 稳增长爆组合

模拟实盘

阴吹思婷

■ 策略收益 ■ 基准收益



年化收益 **142.52%** | 最大回撤 **15.88%** | 初始资金 **¥ 1000000**

已有**585**人订阅

免费订阅

### 银行日内

模拟实盘

囚徒之爱

■ 策略收益 ■ 基准收益



年化收益 **78.64%** | 最大回撤 **4.51%** | 初始资金 **¥ 30000**

### 【量化课堂】股指期货跨期套... 策略回测

JoinQuant量化课堂

■ 策略收益 ■ 基准收益



年化收益 **56.11%** | 最大回撤 **17.13%** | 初始资金 **¥ 1000000**

### 分级A轮动策略

策略回测

囚徒

■ 策略收益 ■ 基准收益



年化收益 **18.58%** | 最大回撤 **1.08%** | 初始资金 **¥ 300000**

# RiceQuant

竞赛 ▾

社区

学院

研究

我的策略

策略英雄榜 <sup>NEW</sup>

数据

帮助 ▾

注册

登录

RiceQuant

# RiceQuant

一个为你量身打造的量化策略平台

灵感 · 策略 · 代码 · 交易

编写您的算法

新手入门



策略研究



历史回测

强大、易用的量化接口API，易于编写交易策略  
免费提供10年+的日、分钟级历史数据以及400多项指标的财务数据  
极速、精准的回测体验，快速开发和验证投资策略

# RiceQuant

## 策略研究



免费提供IPython Notebook研究平台以及强大的金融、数学等工具库  
免费提供10年+的日、分钟级历史数据以及400多项指标的财务数据  
灵活的文本编辑和绘图功能，提供无与伦比的交互式体验

# RiceQuant



## 历史回测

强大、易用的量化接口API，易于编写交易策略

免费提供10年+的日、分钟级历史数据以及400多项指标的财务数据

极速、精准的回测体验，快速开发和验证投资策略



# RiceQuant



## 实时模拟交易

一键部署，云端永久运行

微秒级别实时数据推送计算

将会提供微信、邮件等交易信号推送

# RiceQuant



复制到新策略

保存

编译策略



2016-01-01 至

2017-01-01

股票

¥ 1000000

每日

更多

运行回测

回测收益

6.236%

回测年化收益

6.289%

基准收益

-4.579%

基准年化收益

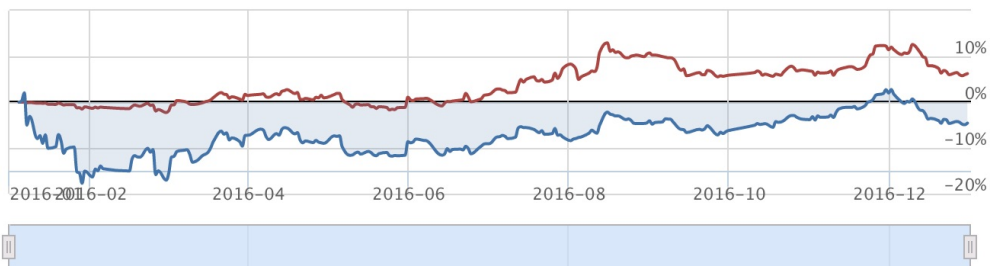
-4.616%

Sharpe

0.4057

最大回撤

6.573%



日志

运行时错误

- 2016-01-04 WARN [Deprecated]在before\_trading函数中, 第二个参数bar\_dict已经不再使用了。
- 2016-01-04 INFO Interested at stock: 000001.XSHE
- 2016-06-27 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8610.00。
- 2016-06-28 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8630.00。
- 2016-06-29 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8690.00。
- 2016-06-30 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8700.00。
- 2016-07-01 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8710.00。
- 2016-07-04 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8810.00。
- 2016-07-05 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8810.00。
- 2016-07-06 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8789.90。
- 2016-07-07 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8780.00。
- 2016-07-08 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8740.00。
- 2016-07-11 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8750.00。
- 2016-07-12 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8880.00。
- 2016-07-13 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8990.00。
- 2016-07-14 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8940.00。
- 2016-07-15 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 8990.00。
- 2016-07-18 WARN 订单被拒单: 可用资金不足。当前资金: 6756.93, 000001.XSHE 下单所需资金: 9039.90。

< 快捷键 ctrl+i / cmd+i 开启股票代码搜索功能 >

# RiceQuant

RiceQuant

竞赛

社区

学院

研究

我的策略

策略英雄榜

数据

帮助



二八轮动简单版

股票

编辑策略

回测结果

历史回测

复制到新策略

保存

编译策略



2015-01-04 至 2017-01-04

股票 ¥ 1000000

每日

更多

运行回测

```
1 # 在这个方法中编写任何的初始化逻辑。context对象将会在你的算法策略的任何方法之间做传递。
2 def init(context):
3     # 沪深300指数、中证500指数和国债指数
4     context.stocks = ["000300.XSHG", "000905.XSHG", "000012.XSHG"]
5     # before_trading此函数会在每天交易开始前被调用，当天只会被调用一次
6     # 你选择的证券的数据更新将会触发此段逻辑，例如日或分钟历史数据切片或者是实时数据切片更新
7 def handle_bar(context, bar_dict):
8     # 开始编写你的主要的算法逻辑
9     hs300 = history_bars(context.stocks[0], 20, "1d", "close")
10    zz500 = history_bars(context.stocks[1], 20, "1d", "close")
11    hsIncrease = hs300[19] - hs300[0]
12    zzIncrease = zz500[19] - zz500[0]
13    p = context.portfolio.positions
14    hsQuality = p[context.stocks[0]].quantity
15    zzQuality = p[context.stocks[1]].quantity
16    gzQuality = p[context.stocks[2]].quantity
17    if hsIncrease < 0 and zzIncrease < 0:
18        if hsQuality > 0:
19            order_target_percent(context.stocks[0], 0)
20            logger.info("卖出沪深300")
21        if zzQuality > 0:
22            order_target_percent(context.stocks[1], 0)
23            logger.info("卖出中证500")
24        if gzQuality <= 0.001:
25            order_target_percent(context.stocks[2], 1)
26            logger.info("买入国债")
27    elif hsIncrease < zzIncrease:
28        if hsQuality > 0:
29            order_target_percent(context.stocks[0], 0)
30            logger.info("卖出沪深300")
31        if gzQuality > 0:
32            order_target_percent(context.stocks[2], 0)
33            logger.info("卖出国债")
34    if zzQuality <= 0.001:
```

< 快捷键 ctrl+i/cmd+i 开启股票代码搜索功能 >



| 日期         | 日志   | 运行时错误   |
|------------|------|---------|
| 2015-01-05 | INFO | 买入沪深300 |
| 2015-01-19 | INFO | 卖出沪深300 |
| 2015-01-19 | INFO | 买入国债    |
| 2015-01-20 | INFO | 卖出国债    |
| 2015-01-20 | INFO | 买入中证500 |
| 2015-05-05 | INFO | 卖出中证500 |
| 2015-05-05 | INFO | 买入沪深300 |
| 2015-05-06 | INFO | 卖出沪深300 |
| 2015-05-06 | INFO | 买入中证500 |
| 2015-05-07 | INFO | 卖出中证500 |
| 2015-05-07 | INFO | 买入沪深300 |
| 2015-05-08 | INFO | 卖出沪深300 |
| 2015-05-08 | INFO | 买入中证500 |
| 2015-06-19 | INFO | 卖出中证500 |
| 2015-06-19 | INFO | 买入国债    |
| 2015-06-25 | INFO | 卖出国债    |
| 2015-06-25 | INFO | 买入中证500 |

# MultiCharts



+1 888 340 6572

MULTICHARTS

MULTICHARTS .NET

SUPPORT

COMPANY

## MultiCharts 12

Advanced market analysis features for expert traders

- Native chart type for Time Price Opportunity analysis
- Matrix Optimization for the strategy re-optimization
- Evaluate Strategy Robustness with greater ease
- More Cryptocurrency data providers are now available
- Local order emulation has been enhanced
- New Kase Bar custom resolution plugin has been added
- Completely new optimization GUI and extended Report metrics
- Flush Cached Data to Database manually
- You can now view optimization reports on-the-go

More exciting features & improvements

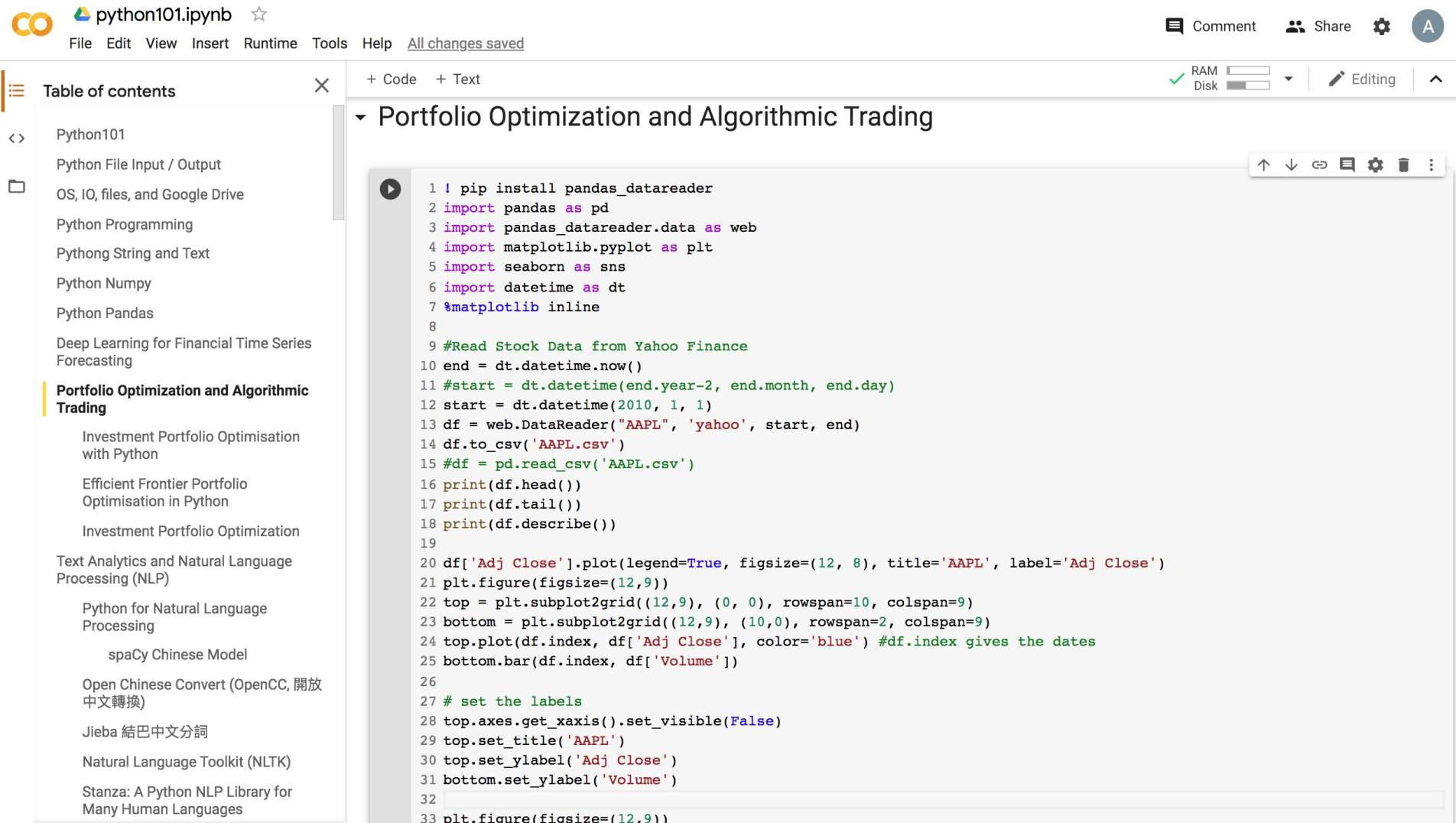
LEARN MORE

TRY IT FOR FREE

<https://www.multicharts.com/>

# Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



The screenshot shows a Google Colab notebook titled "python101.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A "Table of contents" sidebar is open on the left, listing various topics under "Python101", with "Portfolio Optimization and Algorithmic Trading" selected. The main workspace displays a code cell with the following Python code:

```
1 ! pip install pandas_datareader
2 import pandas as pd
3 import pandas_datareader.data as web
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 import datetime as dt
7 %matplotlib inline
8
9 #Read Stock Data from Yahoo Finance
10 end = dt.datetime.now()
11 #start = dt.datetime(end.year-2, end.month, end.day)
12 start = dt.datetime(2010, 1, 1)
13 df = web.DataReader("AAPL", 'yahoo', start, end)
14 df.to_csv('AAPL.csv')
15 #df = pd.read_csv('AAPL.csv')
16 print(df.head())
17 print(df.tail())
18 print(df.describe())
19
20 df['Adj Close'].plot(legend=True, figsize=(12, 8), title='AAPL', label='Adj Close')
21 plt.figure(figsize=(12,9))
22 top = plt.subplot2grid((12,9), (0, 0), rowspan=10, colspan=9)
23 bottom = plt.subplot2grid((12,9), (10,0), rowspan=2, colspan=9)
24 top.plot(df.index, df['Adj Close'], color='blue') #df.index gives the dates
25 bottom.bar(df.index, df['Volume'])
26
27 # set the labels
28 top.axes.get_xaxis().set_visible(False)
29 top.set_title('AAPL')
30 top.set_ylabel('Adj Close')
31 bottom.set_ylabel('Volume')
32
33 plt.figure(figsize=(12,9))
```

<https://tinyurl.com/imtkupython101>

# Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share


RAM Disk Editing

### Table of contents

- Python101
- Python File Input / Output
- OS, IO, files, and Google Drive
- Python Programming
- Python String and Text
- Python Numpy
- Python Pandas
- Deep Learning for Financial Time Series Forecasting
- Portfolio Optimization and Algorithmic Trading**
  - Investment Portfolio Optimisation with Python
  - Efficient Frontier Portfolio Optimisation in Python
  - Investment Portfolio Optimization
  - Text Analytics and Natural Language Processing (NLP)
    - Python for Natural Language Processing
      - spaCy Chinese Model
    - Open Chinese Convert (OpenCC, 開放中文轉換)
    - Jieba 結巴中文分詞
    - Natural Language Toolkit (NLTK)
    - Stanza: A Python NLP Library for Many Human Languages

```
2 !pip install plotly
3 import plotly.graph_objects as go
4
5 import pandas as pd
6 from datetime import datetime
7 df = pd.read_csv('AAPL.csv')
8 fig = go.Figure(data=[go.Candlestick(x=df['Date'],
9                                     open=df['Open'],
10                                    high=df['High'],
11                                    low=df['Low'],
12                                    close=df['Close'])])
13
14 fig.show()
```

Requirement already satisfied: plotly in /usr/local/lib/python3.6/dist-packages (4.4.1)  
Requirement already satisfied: retrying>=1.3.3 in /usr/local/lib/python3.6/dist-packages (from plotly) (1.3.3)  
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from plotly) (1.12.0)



<https://tinyurl.com/imtkupython101>

# Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk Editing

### Table of contents

- Python101
- Python File Input / Output
- OS, IO, files, and Google Drive
- Python Programming
- Pythong String and Text
- Python Numpy
- Python Pandas
- Deep Learning for Financial Time Series Forecasting
- Portfolio Optimization and Algorithmic Trading

#### Investment Portfolio Optimisation with Python

- Efficient Frontier Portfolio Optimisation in Python
- Investment Portfolio Optimization
- Text Analytics and Natural Language Processing (NLP)
  - Python for Natural Language Processing
    - spaCy Chinese Model
  - Open Chinese Convert (OpenCC, 開放中文轉換)
  - Jieba 結巴中文分詞
  - Natural Language Toolkit (NLTK)
  - Stanza: A Python NLP Library for Manv Human Lanuaaces

```
51 max_sharpe_port = results_frame.iloc[results_frame['sharpe'].idxmax()]
52 #locate positon of portfolio with minimum standard deviation
53 min_vol_port = results_frame.iloc[results_frame['stdev'].idxmin()]
54
55 #create scatter plot coloured by Sharpe Ratio
56 plt.figure(figsize=(10,6))
57 plt.scatter(results_frame.stdev,results_frame.ret,c=results_frame.sharpe,cmap='RdYlBu')
58 plt.xlabel('Volatility')
59 plt.ylabel('Returns')
60 plt.colorbar()
61 #plot red star to highlight position of portfolio with highest Sharpe Ratio
62 plt.scatter(max_sharpe_port[1],max_sharpe_port[0],marker=(5,1,0),color='r',s=1000)
63 #plot green star to highlight position of minimum variance portfolio
64 plt.scatter(min_vol_port[1],min_vol_port[0],marker=(5,1,0),color='g',s=500)
```

<matplotlib.collections.PathCollection at 0x7f13132a01d0>



The figure is a scatter plot showing the relationship between Volatility (x-axis, ranging from 0.24 to 0.34) and Returns (y-axis, ranging from 0.16 to 0.30). The data points are colored based on the Sharpe Ratio, with a color bar on the right ranging from 0.6 (red) to 1.1 (blue). A red star highlights the portfolio with the highest Sharpe Ratio, located at approximately (0.25, 0.28). A green star highlights the minimum variance portfolio, located at approximately (0.24, 0.23).

<https://tinyurl.com/imtkupython101>

# Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk Editing

Table of contents

- Python101
- Python File Input / Output
- OS, IO, files, and Google Drive
- Python Programming
- Pythong String and Text
- Python Numpy
- Python Pandas
- Deep Learning for Financial Time Series Forecasting
- Portfolio Optimization and Algorithmic Trading
  - Investment Portfolio Optimisation with Python
  - Efficient Frontier Portfolio Optimisation in Python**
  - Investment Portfolio Optimization
- Text Analytics and Natural Language Processing (NLP)
  - Python for Natural Language Processing
    - spaCy Chinese Model
  - Open Chinese Convert (OpenCC, 開放中文轉換)
  - Jieba 結巴中文分詞
  - Natural Language Toolkit (NLTK)
  - Stanza: A Python NLP Library for Many Human Languages

```
Annualised Return: 0.18
Annualised Volatility: 0.18

      AAPL  AMZN  FB  GOOGL
allocation 44.67 29.05 26.28 0.0
-----
Minimum Volatility Portfolio Allocation

Annualised Return: 0.22
Annualised Volatility: 0.16

      AAPL  AMZN  FB  GOOGL
allocation 34.02 0.73 6.98 58.26
```

Calculated Portfolio Optimization based on Efficient Frontier

Legend:

- efficient frontier
- Maximum Sharpe ratio
- Minimum volatility

Y-axis: annualised returns (0.20 to 0.32)

X-axis: annualised volatility (0.16 to 0.24)

Color scale: 1.0 to 1.5

<https://tinyurl.com/imtkupython101>



# Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings A

RAM Disk Editing

Table of contents

- Python101
- Python File Input / Output
- OS, IO, files, and Google Drive
- Python Programming
- Pythong String and Text
- Python Numpy
- Python Pandas
- Deep Learning for Financial Time Series Forecasting
- Portfolio Optimization and Algorithmic Trading
  - Investment Portfolio Optimisation with Python
  - Efficient Frontier Portfolio Optimisation in Python**
  - Investment Portfolio Optimization
- Text Analytics and Natural Language Processing (NLP)
  - Python for Natural Language Processing
    - spaCy Chinese Model
  - Open Chinese Convert (OpenCC, 開放中文轉換)
  - Jieba 結巴中文分詞
  - Natural Language Toolkit (NLTK)
  - Stanza: A Python NLP Library for Many Human Languages

```
Annualised Return: 0.22
Annualised Volatility: 0.16

allocation  AAPL  AMZN   FB  GOOGL
34.02  0.73  6.98  58.26
```

-----

Individual Stock Returns and Volatility

```
AAPL : annuaised return 0.28 , annualised volatility: 0.21
AMZN : annuaised return 0.34 , annualised volatility: 0.25
FB : annuaised return 0.3 , annualised volatility: 0.23
GOOGL : annuaised return 0.18 , annualised volatility: 0.18
```

-----

Portfolio Optimization with Individual Stocks

| Stock | Annualised Return | Annualised Volatility |
|-------|-------------------|-----------------------|
| AAPL  | 0.28              | 0.21                  |
| AMZN  | 0.34              | 0.25                  |
| FB    | 0.30              | 0.23                  |
| GOOGL | 0.18              | 0.18                  |

<https://tinyurl.com/imtkupython101>

# Summary

- **Portfolio Optimization**
- **Algorithmic Trading**

# References

- Paolo Sironi (2016), “FinTech Innovation: From Robo-Advisors to Goal Based Investing and Gamification”, Wiley.
- Ernie Chan (2008), “Quantitative Trading: How to Build Your Own Algorithmic Trading Business”, Wiley
- Ernie Chan (2013), “Algorithmic Trading: Winning Strategies and Their Rationale”, Wiley
- Ernest P. Chan (2017), “Machine Trading: Deploying Computer Algorithms to Conquer the Markets”, Wiley
- Aurélien Géron (2019), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition, O’Reilly Media, 2019, <https://github.com/ageron/handson-ml2>
- Yves Hilpisch (2018), "Python for Finance: Mastering Data-Driven Finance", 2nd Edition, O'Reilly Media, <https://github.com/yhilpisch/py4fi2nd>
- Michael Heydt (2015) , Mastering Pandas for Finance, Packt Publishing
- Tucker Balch (2012), Investment Science: Portfolio Optimization, <https://www.youtube.com/watch?v=5qbMhXXq0vI>
- Quantopian, <https://www.quantopian.com/>
- Zipline, <https://github.com/quantopian/zipline>
- Pyfolio, <https://github.com/quantopian/pyfolio>
- UQER, <https://uqer.io/>
- Joinquant, <https://www.joinquant.com/>
- Ricequant, <https://www.ricequant.com/>
- MultiCharts, <https://www.multicharts.com/>