PYCON KOREA 2023

Quantitative Trading using Python

Stream Big Data into Money Maker

Jung, Jin Hyuk, Ph.D.

BACK TO US,
BACK TO PYTHON

목차

- Data collection
  - pandas, etc
- Making signals
  - Types of Signals and an Example
- Simulation and backtest of signals
- Combining many signals to make a strong signal
  - Portfolio optimization on signals - cvxpy
- Transforming a strong signal to a tradable strategy
  - Trade optimizer - cvxpy

About the Speaker

- 2023/01~Present, Chief Research Officer, Presto Labs, Pte. Ltd, Singapore
- 2014/07~2022/05, Quantitative Portfolio Manager/Advisor PM, Millennium Capital Management/WorldQuant, Pte. Ltd, Singapore
- 2012/10~2014/07, Quantitative Portfolio Manager, WorldQuant, LLC, Old Greenwich, CT, USA
- 2012/05~2012/09, Quantitative Portfolio Manager, Millennium Partners, London, UK
- 2008/08~2011/11, Quant Trader/Front Derivatives Quant, Dept. of Financial Engineering, Korea Investment & Securities, Co. Ltd., Seoul, S. Korea
- 2008/04~2008/10, Senior Quant Researcher, Research Center, Daishin Securities, Co. Ltd., Seoul, S. Korea
- 2003/09~2008/02, Teaching and Research Assistant during Ph.D. program, Dept. of Computer Science, School of Science and Mathematics, University of Maryland, College Park, MD, USA
    - Ph.D. and M.S, at Scientific Computing (Applied Mathematics) in Computer Science with Optimization as the Research Specialization. Minor research areas include Machine Learning and GPGPU (General Purpose GPU).
- 1996/03~2000/02, B.Eng in Computer Engineering, Seoul National University, Seoul, S. Korea
- 1993/03~1996/02, Seoul Science High School, Seoul, S. Korea

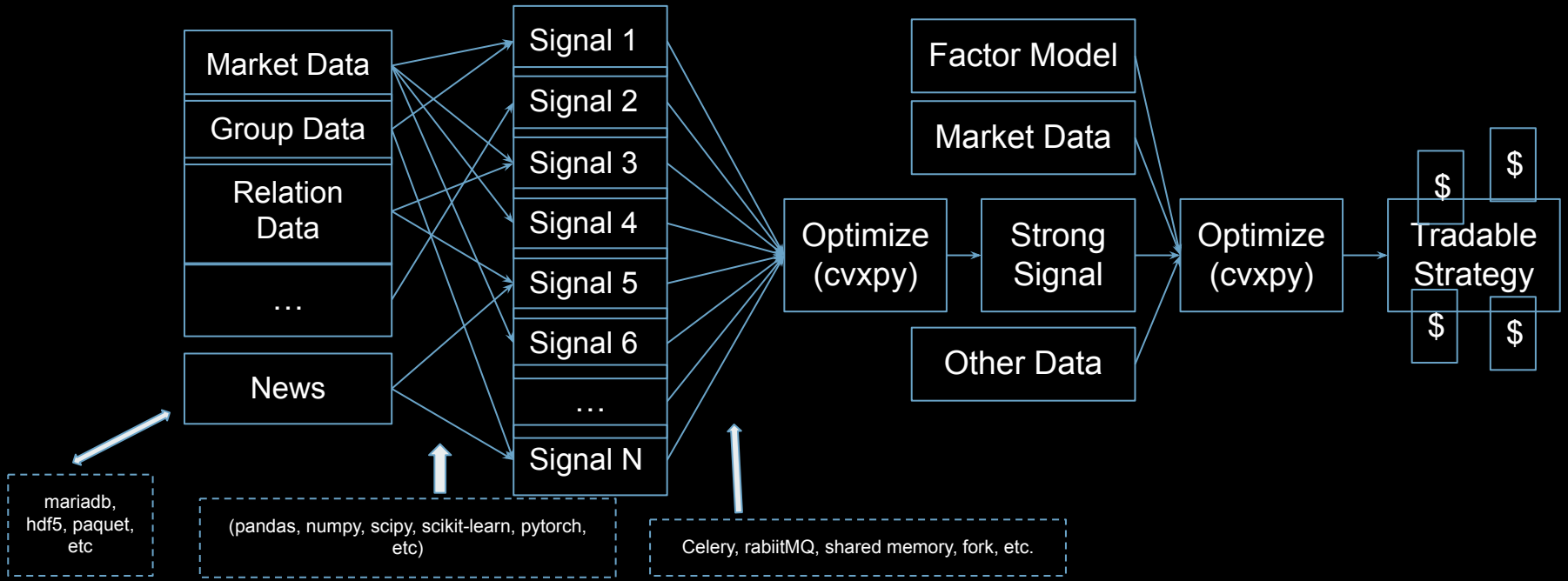
Data Collection

- Data is the root source of signals
- Types of data
  - Market data
    - Price, Volume, Order book
    - Short Interest
  - Group data
    - Sector, Industry, Subindustry
  - Relation Data
    - Revere (Supply Chain, Competition, etc)
  - Factor Models
    - MSCI Barra, etc.
  - Fundamental Data
    - FnGuide, FactSet, CompuStat, etc
  - Analyst Forecast
    - IBES
  - News
    - Bloomberg, Reuter
  - Social Media
    - Tweets, Reddit, StockTwits, etc
  - Satellite Images

Data Collection

- Data cleaning process should be done after collection
- Timestamp should be kept at the time of collection to be used in backtest
- The data retrieved from a third-party should be differentiated
  - Historical data is already cleaned
  - Subscribed data could be dirty and often amended later.
  - Data insample v.s. Data out of sample

Stream of Making Money out of Big Data



Data

- Market Data
  - Historical daily data from Yahoo! Finance

```
import pandas as pd
import yfinance as yf
from yahoofinancials import YahooFinancials
ticker = yf.Ticker('AAPL')
aapl_df = ticker.history(period="5y")
aapl_df['Close'].plot(title="APPLE's stock price")
```

  - Another free source for market data is openbb terminal
    - https://my.openbb.co/app/terminal/
  - For real trading, it is better to rely on a reliable real time and historical data vendor
    - Bloomberg, Refinitive, Direct feed from the exchanges/brokers, etc.

Signals

- Also known as Alphas
- Types of signal
  - Price reversion
  - Trade volume
  - Group momentum
  - Fundamental
  - Analyst
  - Earnings surprise
  - Short interest
  - Pairs trading
  - ...
- Target Universe
  - Examples:
    - Top N most liquid stocks
    - Top N biggest market cap stocks
  - When making universe, delisted (including halted) stocks should be included.
    - Otherwise, survivorship bias will be induced and the result is useless.
- More Breadth than Depth
  - Machines can handle more number of instruments than humans.
  - For a single instrument, humans have more insight than machines. (This may change in the future. Who know?)

An Example of Momentum Signal

```
alpha = basedata.close_mid                # Mid price of bid and ask
alpha = ts_zscore(alpha, days=2016)       # Time series z-score for 2016 intervals (of 5min) - 7 days
alpha = ewm(alpha, halflife=42)           # Exponential moving average
alpha = neutralize(alpha)                 # Make dollar neutral
alpha = scale(alpha, 20e6)                # Scale to book size of 20e6 (20M) GMV
```

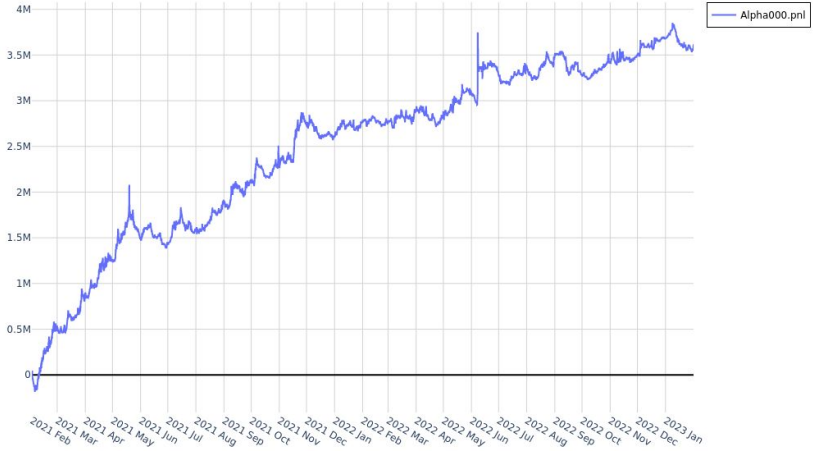
Examples:

101 Formulaic Alphas

https://arxiv.org/ftp/arxiv/papers/1601/1601.00991.pdf

Backtest and Performance Evaluation

Tested on Binance TOP Liquid 50 USDT perpetual contract symbols with 5min data



test_binance_20230609_081353/Alpha000.pnl

index	Booksize	Return	Sharpe(ir)	DailySharpe	Max.DD	Win/lose	Turnover	ReturnBP
20210201-20210701	2.00E+06	174.66	3.87(0.01)	0.220	34.89	50.22	134.59	35.55
20210701-20220101	2.00E+06	116.24	3.36(0.01)	0.163	15.34	49.54	131.23	24.27
20220101-20220701	2.00E+06	67.72	1.71(0.01)	0.068	25.42	48.91	135.32	13.71
20220701-20230101	2.00E+06	39.85	1.43(0.00)	0.076	15.81	49.13	132.27	8.25
20230101-20230131	2.00E+06	-36.55	-1.46(-0.00)	-0.072	16.10	49.48	161.33	-6.21
Total	2.00E+06	90.40	2.48(0.01)	0.119	34.89	49.42	134.48	18.42

Portfolio Optimization on Signals

- Make strong signal by combining multiple signals
  - We can use **convex optimization**, especially **SOCP** (Second Order Conic Programming) or **machine learning** for this purpose
- An example is **Sharpe ratio maximization**

$$\max_{\mathbf{w}} \frac{\boldsymbol{\mu}^\top \mathbf{w}}{\sqrt{\mathbf{w}^\top \boldsymbol{\Sigma} \mathbf{w}}} \xrightarrow{\mathbf{y} = \kappa \mathbf{w}} \max_{\mathbf{y}, \kappa} \frac{\boldsymbol{\mu}^\top \mathbf{y}/\kappa}{\sqrt{\mathbf{y}^\top \boldsymbol{\Sigma} \mathbf{y}/\kappa}} \longrightarrow \min_{\mathbf{y}, \kappa} \mathbf{y}^\top \boldsymbol{\Sigma} \mathbf{y}$$

$$\text{s.t. } \boldsymbol{\mu}^\top \mathbf{y} = 1 \qquad \text{s.t. } \boldsymbol{\mu}^\top \mathbf{y} = 1$$

$$\kappa > 0 \qquad \kappa \geq 0$$

Example of Sharpe Ratio Maximization

With **size limit** (Long & Short)

$$\|\mathbf{w}\|_1 = 1 \xrightarrow{\mathbf{y} = \kappa\mathbf{w}} \|\mathbf{y}\|_1 = \kappa \xrightarrow{\text{Convexity}} \|\mathbf{y}\|_1 \leq \kappa$$

```
import cvxpy as cp
import numpy as np
...
def max_sharpe(R, turnover, w_L, w_U, diversity_coefficient, max_turnover):
        m, n = R.shape
        mu = np.mean(R, axis=0)
        # Initialization
        constraints = []
        objective = 0
        y = cp.Variable(n)
        kappa = cp.Varianble(1)
```

Example of Sharpe Ratio Maximization

```
R_bar = R - mu[None, :]
f = R_bar @ y # Note that 1 / (m - 1) * f.T @ f is y.T @ Sigma @ y
 constraints += [
            cp.norm(y, 1) <= kappa,
            mu.T @ y == 1,
]
objective += 0.5 / (m - 1) * f.T @ f          ... # More constraints and objective terms
prob = cp.Problem(cp.Minimize(objective), constraints)
prob.solve(solver="ECOS")
w = y.value
w /= np.norm(w, ord=1)
return w
```

For more detail:
https://github.com/salbang/QuantTrading/blob/main/weight/sharpe_ratio_maximization/adding_constraints_and_more_objective_terms_for_socp_solver.md

Strong Signal

- A very simple example is an equally averaged signals
  - combo = np.mean(signals, axis=0, keepdims=False)
- A simple example is a weighted average of signals, where weights are obtained from the portfolio optimization process
  - combo = np.einsum('st,sti->ti', w, signals)
- You may have more creative idea to combine signals using non-linear models

Strong Signal to Strategy

- Why can’t we just trade the strong signal directly?
  - It does not consider liquidity and cost.
    - Market impact and slippage might be more than the margin we could achieve.
      - Not profitable enough
      - Maximum tradable size might be limited.
  - It may have much more factor exposure or variance than we can allow.
- How to resolve?
  - Build an optimization model - Trade Optimizer
    - Consider factor exposure
    - Consider liquidity of assets
    - Consider market impact
    - etc.

Trade Optimizer

- Making target as close as possible to the ideal combo alpha
  - This can be achieved with either maximizing the similarity or minimizing the distance between the target alpha and the ideal combo alpha

$$\min_{\boldsymbol{\alpha}} \frac{1}{2} \|\boldsymbol{\alpha} - \boldsymbol{\alpha}^o\|_2^2$$

Code:

```
n = len(alpha_0)
alpha = cp.Variable(n)
objective = 0.5 * (alpha - alpha_0) @ (alpha - alpha_0)
```

Trade Optimizer - Size Constraints

- Basic size constraints with dollar neutrality (for long-short dollar neutral portfolio)

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^o\|_2^2$$
$$\text{s.t. } \|\boldsymbol{\alpha}\|_1 \leq \|\boldsymbol{\alpha}^o\|_1$$
$$\mathbf{1}^\top \boldsymbol{\alpha} = 0$$

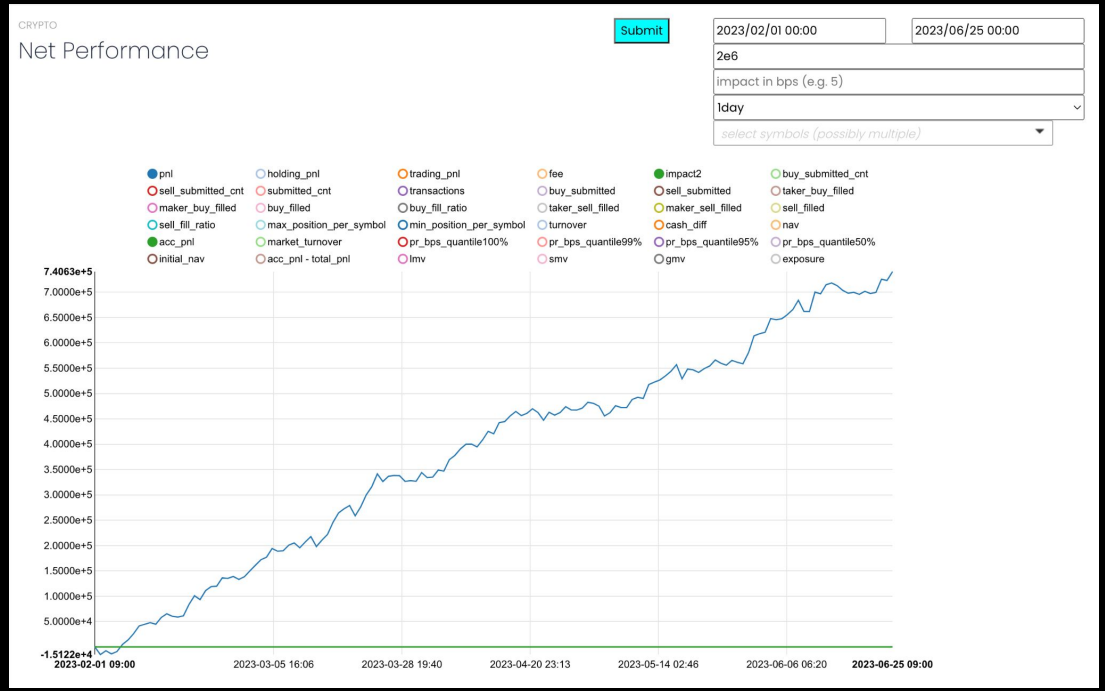
Code:

```
constraints = []
constraints += [
        cp.norm(alpha, 1) <= np.linalg.norm(alpha_0, ord=1),
        cp.sum(alpha) == 0
]
```

Trade Optimizer - Things that can be added

- Risk minimization using a factor model such as Barra equity model (USE4, GME3, etc)
- Limiting risk factor exposure on a set of selected factors (Momentum, Growth, etc)
- Limiting systematic risk
- Turnover control
- Impact minimization
- Limiting holding position
- Limiting trade over market liquidity
- Minimizing transaction costs (such as slippage, impact , etc.)
- etc.

Final Strategy (Example)



Interval	Return	Sharpe Ratio	MDD	Return Per Trade	Win Ratio	Average Turnover	GMV (Book Size)	Exposure
2023-02-01 ~ 2023-06-25	37.03%	8.79	1.40%	8.65 bp	65.97%	5.95e+06 USDT	2.00e+06 USDT	0.33%

More things to go live

- Execution engine
  - Order management
- Account management
- Event management
  - Corporate actions
- Strategy monitor
  - Event monitor
  - PNL monitor
  - Risk monitor
- Etc.

Thank you

- Github page:
  - https://github.com/salbang/QuantTrading/
- Questions?