Markowitz Portfolio Optimization

Zachary Brooks, Christopher Demirjian, Kaja Huruk

Introduction

Harry Markowitz, *Portfolio Selection*, 1952 - pioneered Modern Portfolio Theory

Models the rate of returns on assets as random variables, with the goal of choosing optimal portfolio weighting factors to maximize returns/minimize volatility

Core concept: Optimizes portfolio to maximize expected return for a given level of risk using mean-variance analysis

Key insights

- Focus on diversification: diversification of investments to reduce risks is more important than maximizing returns on individual stocks
- Combining uncorrelated assets into a portfolio can reduce its risk without sacrificing the returns
- Variance/standard deviation as a measure of risk: optimal portfolio maximizes returns for a given level of risk
- Efficient frontier: set of optimal portfolios offering highest expected returns

Inputs

- Predicted returns of n stocks (a good guess is the average of historical returns)
  - $\mu = r_1, \ldots, r_n$

- Covariance matrix of stocks
  - $$\text{Covariance matrix} = \underbrace{\begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \sigma_n \end{bmatrix}}_{\text{Diagonal matrix with standard deviations in the diagonal (and zeros in the other cells)}} \times \underbrace{\begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1n} \\ \rho_{21} & 1 & \cdots & \rho_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \cdots & \cdots & 1 \end{bmatrix}}_{\text{Correlation matrix}} \times \underbrace{\begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \sigma_n \end{bmatrix}}_{\text{Diagonal matrix with standard deviations in the diagonal (and zeros in the other cells) – same as the one before}}$$

Outputs

- Restrictions of weights
  - $(x_1, \ldots, x_n)$

- Weights of the nth stock
  - $(M_1, M_2, \ldots, M_n)$

Markowitz Portfolio

- Inputs:
  - Predicted returns on stocks: r1…rn
  - Covariance metrics of stocks
- Portfolio 1: minimize volatility
- Portfolio 2: minimize volatility given some target return
- Portfolio 3: maximize Sharpe ratio

Portfolio 1 Code (minimize volatility)

```
#collecting tickers and historical data

tickers = ['NORD', 'MSFT', 'SBUX']

df_list = []
for ticker in tickers:
  print(ticker)
  prices = yf.download(ticker, start = '2023-04-29')
  prices = prices[['Adj Close']].rename(columns = {'Adj Close': ticker})
  df_list.append(prices)

df_prices = pd.concat(df_list, axis=1)

df_return = df_prices.pct_change()

df = pd.merge(df_prices, df_return, how='left', left_index = True, right_index = True, suffixes = ('','_prc'))
df = df.dropna(axis=0, how = 'any')
```

Markowitz's Key Insight

Finding the optimal portfolio that maximizes returns subject to a given risk level:

$$\begin{aligned} \max_{\mathbf{x}} \mu_p &= \mathbf{x}'\boldsymbol{\mu} \text{ s.t.} \\ \sigma_p^2 &= \mathbf{x}'\Sigma\mathbf{x} = \sigma_{p,0}^2 \text{ and } \mathbf{x}'\mathbf{1} = 1. \end{aligned}$$

Has a dual representation of minimizing risk for a given target return:

$$\begin{aligned} \min_{\mathbf{x}} \ \sigma_{p,x}^2 &= \mathbf{x}'\Sigma\mathbf{x} \ \text{ s.t.} \\ \mu_p &= \mathbf{x}'\boldsymbol{\mu} = \mu_{p,0}, \text{ and } \mathbf{x}'\mathbf{1} = 1, \end{aligned}$$

https://sites.math.washington.edu/~burke/crs/408/fin-proj/mark1.pdf

Portfolio 1: Minimize volatility

- Using matrix notation, we want to find

$$\min_{\mathbf{m}} \ \sigma^2_{p,m} = \mathbf{m}'\Sigma\mathbf{m} \text{ s.t. } \mathbf{m}'\mathbf{1} = 1.$$

- In Markowitz’s original paper, constraint optimization problem was solved by hand using Lagrange's Method
    - With modern computational techniques, we can use quadratic programming optimization techniques to solve portfolio constraint problems

```python
def min_var_weights(Cov, lb, ub):
  num_assets = len(Cov)


  def get_var(w):
    var = np.dot(w.T,np.dot(Cov,w))
    return var


  def risk_function(w):
    return get_var(w)

  def check_sum(w):
    return 1-np.sum(w)

  constraints = ({'type':'eq', 'fun':check_sum})

  w0 = np.array(num_assets * [1.0 / num_assets])
  bounds = ((lb,ub),)*num_assets


  w_opt = minimize(risk_function, w0, method = 'SLSQP', bounds = bounds, constraints=constraints)
  return w_opt.x
```

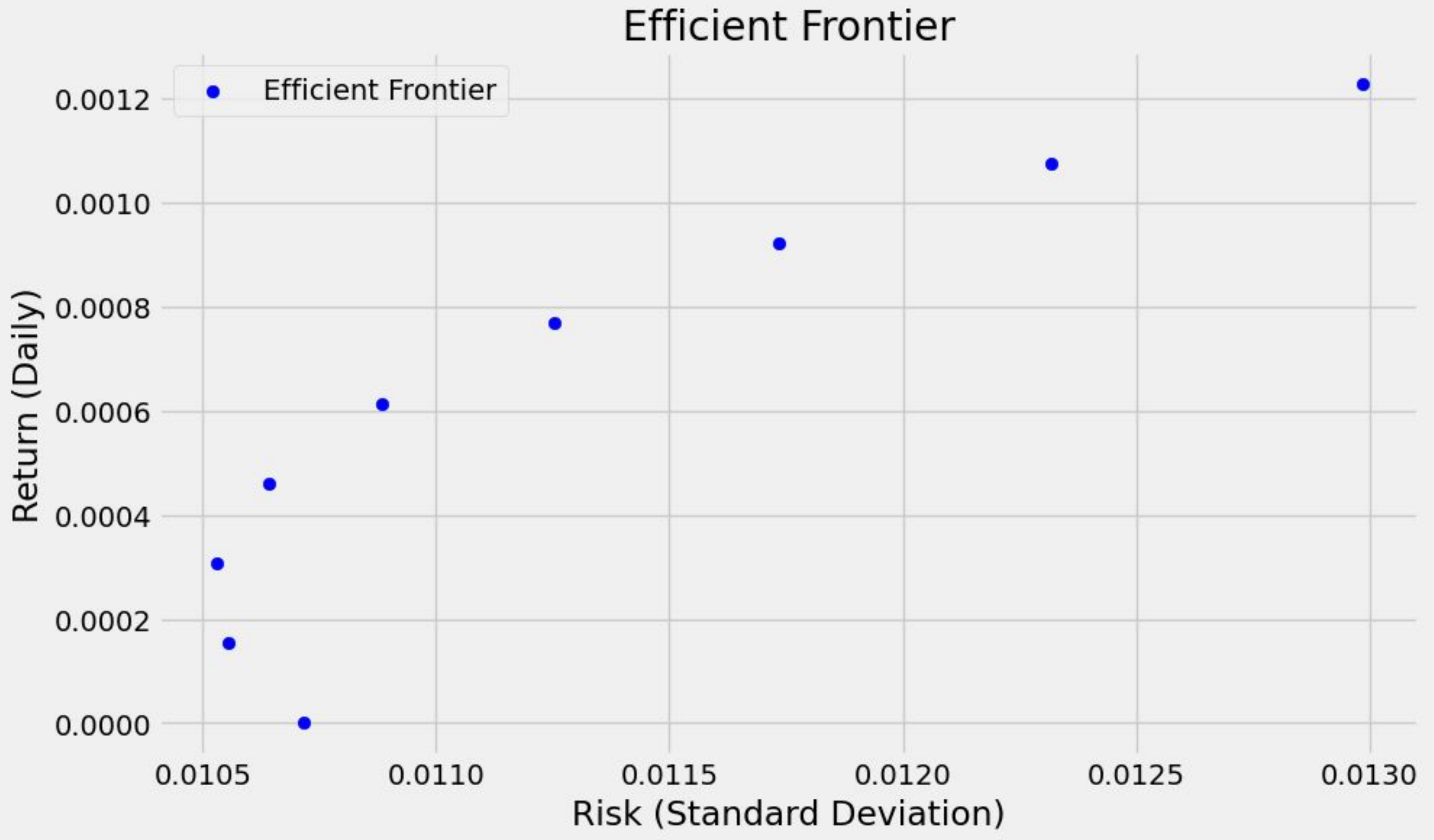
Portfolio 2: Minimize volatility for a given return target

- By adding an additional constraint to our optimization problem, we can now find the least risky portfolio for a given target return: μ_p

$$\min_{\mathbf{x}} \ \sigma^2_{p,x} = \mathbf{x}'\Sigma\mathbf{x} \ \text{ s.t.}$$
$$\mu_p = \mathbf{x}'\boldsymbol{\mu} = \mu_{p,0}, \text{ and } \mathbf{x}'\mathbf{1} = 1,$$

- The set of these solutions for every target return is called the **efficient frontier**. Points on this curve represent the best returns we can get for a given level of risk.

```python
def min_var_for_returns(Cov, target_return, lb, ub):
  num_assets = len(Cov)


  def get_var(w):
    var = np.dot(w.T,np.dot(Cov,w))
    return var


  def risk_function(w):
    return get_var(w)

  def check_sum(w):
    return 1-np.sum(w)

  def constraint_for_target_return(w):
    portfolio_return = np.dot(w, expected_returns)
    return portfolio_return - target_return

  constraints = [{'type': 'eq', 'fun': check_sum},
                 {'type': 'eq', 'fun': constraint_for_target_return}]

  w0 = np.array(num_assets * [1.0 / num_assets])
  bounds = ((lb,ub),)*num_assets


  w_opt = minimize(risk_function, w0, method = 'SLSQP', bounds = bounds, constraints=constraints)
  return w_opt.x
```


Efficient Frontier
Efficient Frontier
Return (Daily)
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0.0105
0.0110
0.0115
0.0120
0.0125
0.0130
Risk (Standard Deviation)

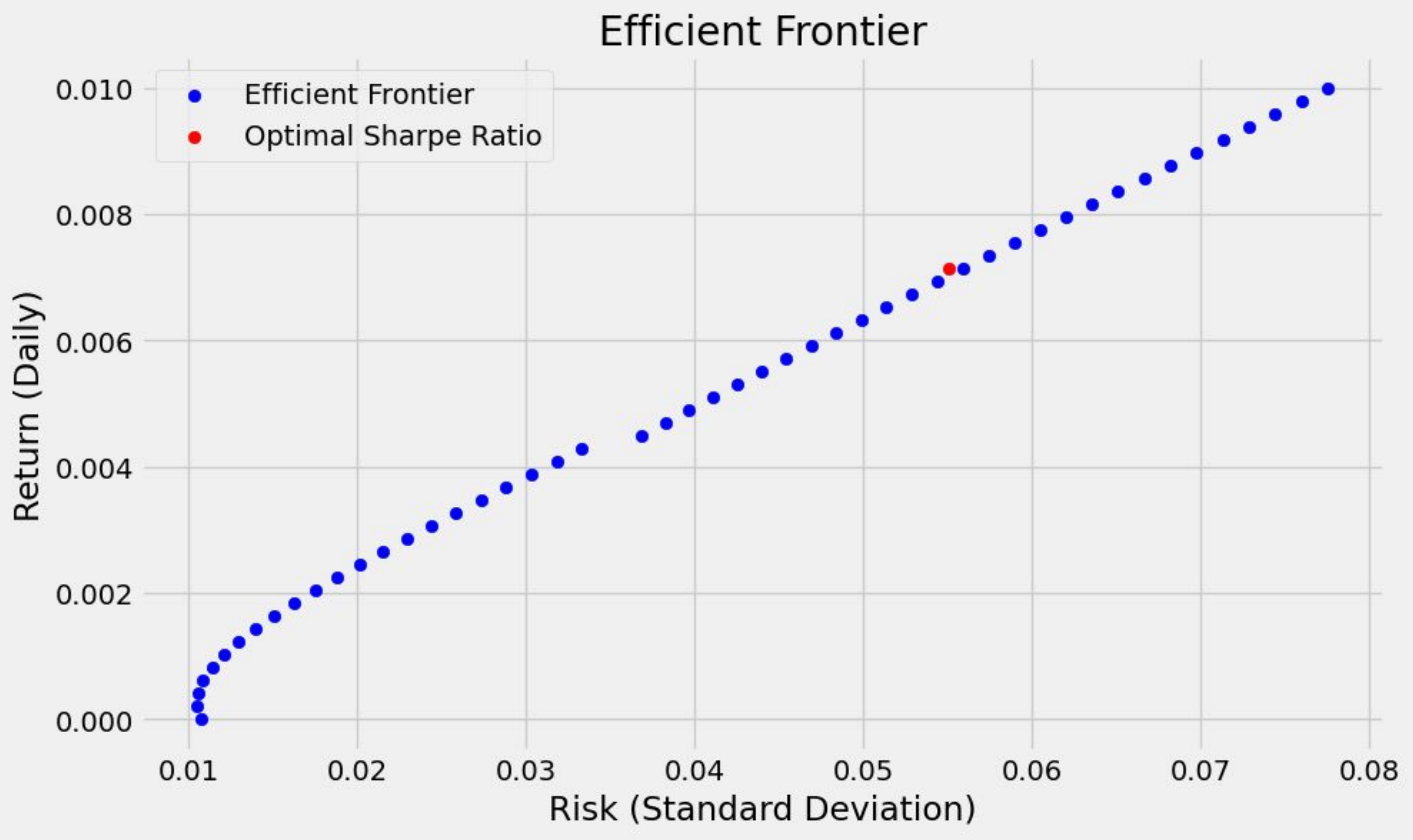
Portfolio 3: Maximize Sharpe ratio

- Of all the portfolios on the efficient frontier, which one is the "best"?
    - The one with the best return-to-risk ratio
- Now consider the following optimization problem:

$$\max_{\mathbf{t}} \frac{\mathbf{t}'\boldsymbol{\mu} - r_f}{(\mathbf{t}'\Sigma\mathbf{t})^{\frac{1}{2}}} = \frac{\mu_{p,t} - r_f}{\sigma_{p,t}} \text{ s.t. } \mathbf{t}'\mathbf{1} = 1.$$

- This is called the **tangency portfolio (optimal weights)**, is it the theoretical maximum Sharpe portfolio

```python
def get_max_sharpe(mu, Cov, rf, lb, ub):
  num_assets = len(mu)
  rf = np.exp(rf/252)-1

  def get_sharpe(w):
    sigma = np.sqrt((np.dot(w.T,np.dot(Cov,w))))
    r = np.sum(mu*w)
    return (r-rf)/sigma

  def risk_function(w):
    return -get_sharpe(w)

  def check_sum(w):
    return 1-np.sum(w)

  constraints = ({'type':'eq', 'fun':check_sum})

  w0 = np.array(num_assets * [1.0 / num_assets])
  bounds = ((lb,ub),)*num_assets


  w_opt = minimize(risk_function, w0, method = 'SLSQP', bounds = bounds, constraints=constraints)
  return w_opt.x
```


Efficient Frontier
Efficient Frontier
Optimal Sharpe Ratio
Return (Daily)
0.010
0.008
0.006
0.004
0.002
0.000
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Risk (Standard Deviation)

Result

Predicted Daily Sharpe of Tangency Portfolio: 0.1410

Predicted Daily Sharpe of Even Weights Portfolio: 0.0197

Applications in real life

- Asset allocation and portfolio construction
- Pension fund management
- Endowment and foundation investing
- Multi-asset and multi-strategy funds
- Risk management and portfolio stress testing

Limitations

- Reliance on historical data
- Assumption of normal distribution
- Sensitivity to input parameters
- Static and single-period framework