Efficient Frontier - Documentation

1. Expected Returns μ and Covariance Matrix Σ:

- **Expected Returns μ:**
  - Represents the mean or expected returns of individual assets.
- **Covariance Matrix Σ:**
  - Represents the covariances between the returns of different assets.
  - Captures how the returns of one asset move concerning the returns of another.

2. Portfolio Returns R_p and Volatility σ_p:

- **Portfolio Returns R_p:**
  - The expected return of a portfolio is the weighted sum of the expected returns of its constituent assets.
  - $R_p = \sum_i w_i \mu_i$
- **Portfolio Volatility (Standard Deviation) σ_p:**
  - The standard deviation of a portfolio measures its risk or volatility.
  - $\sigma_p = \sqrt{\sum_{i,j} w_i w_j \sigma_{ij}}$
  - Involves the covariance matrix Σ weights w.

3. Efficient Frontier Optimization:

- The Efficient Frontier is constructed by solving an optimization problem.
- The goal is often to maximize the Sharpe ratio $\frac{R_p - R_f}{\sigma_p}$
- The `EfficientFrontier` class in PyPortfolioOpt provides functions for optimization.

4. PyPortfolioOpt Functions:

- `EfficientFrontier` **Class:**
  - Core class for constructing the Efficient Frontier.
  - Takes expected returns and covariance matrix as inputs.
- `.min_volatility()` **Method:**
  - Minimizes the portfolio volatility for a given level of expected return.
- `.max_sharpe()` **Method:**
  - Maximizes the Sharpe ratio to find the tangency portfolio on the Efficient Frontier.
- `.efficient_return(target_return)` **Method:**
  - Finds the portfolio with the minimum volatility for a specified target return.
- `.efficient_risk(target_volatility)` **Method:**
  - Finds the portfolio with the maximum Sharpe ratio for a specified target volatility.

5. Constraints:

- Constraints can be applied to the optimization problem, such as budget constraints, minimum/maximum weights, and sector constraints.

Example Code:

```python
from pypfopt.efficient_frontier import EfficientFrontier

# Create an EfficientFrontier object with expected returns and covariance matrix
ef = EfficientFrontier(expected_returns, cov_matrix)

# Optimize for maximum Sharpe ratio
weights = ef.max_sharpe()

# Optimize for minimum volatility at a target return
weights = ef.efficient_return(target_return)
```

Example Scenario:

Suppose we have a portfolio with three assets: Stock A, Stock B, and Stock C.

Step 1: Expected Returns μ and Covariance Matrix Σ :

Let's assume the following expected returns and a simplified covariance matrix:

- Expected Returns μ:
  - Stock A: 8%
  - Stock B: 12%
  - Stock C: 15%
- Covariance Matrix Σ:

$$\Sigma = \begin{bmatrix} 0.0004 & 0.0002 & 0.0003 \\ 0.0002 & 0.0009 & 0.0004 \\ 0.0003 & 0.0004 & 0.001 \end{bmatrix}$$

Step 2: Portfolio Returns R_p and Volatility σ_p:

Now, let's create a few hypothetical portfolios with different weights for the three assets.

1. Portfolio 1 (Equally Weighted):
   - $w_A = 0.33, , w_B = 0.33, , w_C = 0.33$
2. Portfolio 2 (More Weight on Stock C):
   - $w_A = 0.2, , w_B = 0.2, , w_C = 0.6$

For each portfolio, calculate the expected return and volatility:

$R_p = \sum_i w_i \mu_i$

$\sigma_p = \sqrt{\sum_{i,j} w_i w_j \sigma_{ij}}$

Step 3: Efficient Frontier Optimization:

Now, we can use the EfficientFrontier module to find portfolios that optimize certain criteria. For simplicity, let's consider two scenarios:

1. **Maximize Sharpe Ratio:**
   - Find weights that maximize the Sharpe ratio $\frac{R_p - R_f}{\sigma_p}$
2. **Minimize Volatility:**
   - Find weights that minimize portfolio volatility.

These optimisation problems are solved under constraints on the weights as:

$\Sigma w_i = 1$
$w_i 0$; weights cannot be negative.

Step 4: Plotting the Efficient Frontier:

Finally, we can plot the Efficient Frontier to visualize the trade-off between risk and return.

Resources:

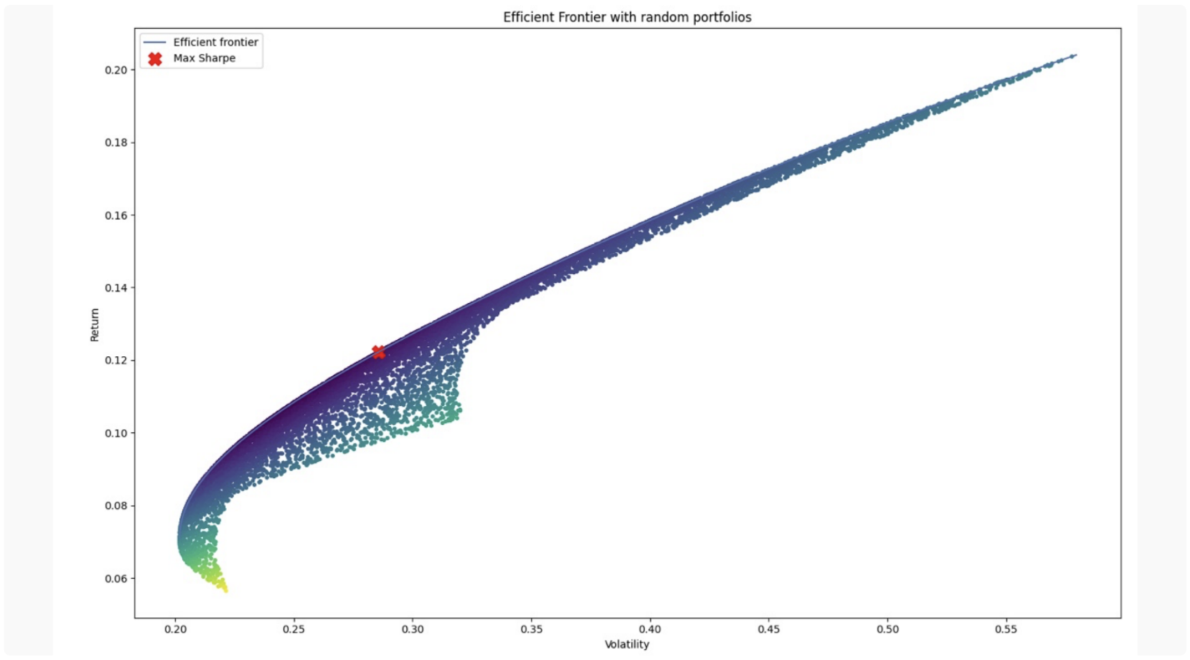
- PyPortfolioOpt Documentation
- PyPortfolioOpt GitHub Repository

Efficient Frontier Curve:

The Efficient Frontier is a set of optimal portfolios that offer the highest expected return for a given level of risk or the lowest risk for a given level of expected return. The Efficient Frontier curve is a graphical representation of these portfolios in the risk-return space.

- **X-Axis (Risk/Volatility):**
  - Represents the standard deviation or volatility of portfolio returns.
  - Portfolios to the left have lower volatility.
- **Y-Axis (Return):**
  - Represents the expected return of the portfolio.
  - Portfolios higher up have higher expected returns.

The Efficient Frontier curve typically slopes upwards, indicating the positive relationship between risk and return. The curve represents the optimal trade-off between risk and return, and portfolios lying on the curve are considered efficient because they provide the best possible combination of risk and return.



Efficient Frontier Plot with max sharpe ratio

Efficient Frontier with Monte Carlo Simulation:

Monte Carlo Simulation can be integrated into the process of constructing the Efficient Frontier. By simulating various scenarios of asset returns, investors can explore a more comprehensive set of possible portfolios. This can lead to a more robust understanding of risk and return trade-offs, especially in situations where historical data may not fully capture the complexity of future market dynamics.

In summary, the Efficient Frontier curve represents the optimal portfolios in terms of risk and return. Monte Carlo Simulation is a powerful tool to enhance the construction of the Efficient Frontier by introducing a probabilistic element, considering a wide range of potential future scenarios for asset returns, and providing a more realistic view of portfolio possibilities.

Something rather interesting about the Efficient frontier graph is that, moving below a certain point creates a non-ideal situation i.e. point where low return start yielding higher risk. This is known as the inefficient region, where portfolios have higher risk for lower expected returns compared to portfolios on the Efficient Frontier. Hence, investors want to stick to the upper half of the curve and not move into this region.