

Modern Corporate Finance



























Assignment – 2 (BE21B037)



Market Risks

1) Select a Market index (refer the index portfolio slide for illustration): Briefly discuss the selection of securities representing the index. Ensure that the selection of market index is not replicated by others in the class (2 marks)

The Market Index I have selected is **BSE Power**. The securities representing the index are given below

BSE POWER 7,593.44 ▲ 40.46 (0.54%) Market Snapshot ▾

Overview	Technical	Fundamental	Performance	Pivot Level	* choose your choice of data ⚙				
Name ▾	LTP ▾	%Chg ▾	Volume ▾	Buy Price ▾	Sell Price ▾	Buy Qty ▾	Sell Qty ▾	Analysis	Technical Rating ▾
ABB India	8,393.30	0.21	1,616	8,375.00	8,435.00	1	3	 Analysis >	
Adani Energy	1,043.10	0.77	2,507	1,043.10	1,045.00	14	25	 Analysis >	
Adani Green Ene	1,834.75	0.15	2,122	1,833.80	1,836.00	168	4	 Analysis >	
Adani Power	634.95	-0.13	20,243	634.95	636.50	39	1,144	 Analysis >	
BHEL	309.40	3.27	340,778	309.40	309.50	48	500	 Analysis >	
CG Power	664.00	1.47	12,851	662.00	664.00	1	2	 Analysis >	
JSW Energy	598.00	0.86	2,978	597.00	598.00	2	15	 Analysis >	
NHPC	97.80	-1.45	501,743	97.78	97.80	5	8,737	 Analysis >	
NTPC	366.00	0.16	40,792	365.80	366.70	5	45	 Analysis >	
Power Grid Corp	316.00	0.85	72,358	316.00	316.80	1,070	100	 Analysis >	
Siemens	7,205.00	0.24	1,660	7,188.35	7,202.00	15	6	 Analysis >	
Suzlon Energy	42.40	-0.33	153,746	42.40	42.50	876	1,159	 Analysis >	
Tata Power	441.00	1.08	192,681	441.00	441.50	6	162	 Analysis >	

The companies from the power sector listed on the Bombay Stock Exchange focus on those involved in electricity generation, distribution, transmission and other such related services. They are selected based on factors such as Market Capitalization, Liquidity, Listing History and Free-Float Market Capitalization.

The index is reviewed semi-annually to ensure that it remains representative with adjustments made based on updated market data. You can find the securities mentioned above [here](#).

2) Collect market data of securities constituting your index to identify the minimum variance portfolio. (5 marks)

The source used to collect all market data/history for all 13 companies listed in BSE Power is <https://www.bseindia.com/>. We use the opening and closing values to calculate the return for each day.

The following is the information about the data:

- 1) Time period: Daily (from March 1st 2023 to March 1st 2024)
- 2) Formula used to calculate return for each day

$$\text{Return} = (\text{Closing Price} - \text{Opening Price}) * 100 / \text{Opening Price}$$

*Note that this formula does not take into consideration stock splits or other factors.

Below is the python code (commented) to calculate the minimum variance portfolio for the given 13 stocks

Libraries to be imported are

```
# !pip install cvxpy
import numpy as np
import pandas as pd
import cvxpy as cp
import matplotlib.pyplot as plt
```

Input the returns data for one year in the given input section

```
df = pd.read_csv("C:\\Users\\shree\\Downloads\\MCF_Assignment 2.xlsx - Returns.csv") # input
the excel file
returns = df[df.columns[1:]]

cov_matrix = returns.cov().values # Calculate the covariance matrix
n = len(returns.columns) # Number of securities
w = cp.Variable(n) # Define the weights variable

portfolio_variance = cp.quad_form(w, cov_matrix) # Objective function: portfolio variance
constraints = [cp.sum(w) == 1] # Constraints: sum of weights is 1, negative weights allowed,
to short stocks as well

# Problem definition
problem = cp.Problem(cp.Minimize(portfolio_variance), constraints)

# Solve the problem
problem.solve()

# Optimal weights
optimal_weights = w.value

companies = list(df.columns.values[1:])
Final_weights = pd.DataFrame({'Company':companies, 'Optimal Weights':list(optimal_weights)})
```

The final optimum weights after running the code will be

Company	Optimal Weights
ABB	0.124281
AE	0.033918
AGE	0.056768
AP	-0.051031
BHEL	-0.048459
CGP	0.129567
JSW	-0.027256
NHPC	-0.058114
NTPC	0.181067
PG	0.257356
Sie	0.219924
SE	0.061310
TP	0.120670

Table: Minimum variance portfolio

3) Construct the Markowitz Market Index with the data applying your indifference curve. (5 marks)

Constructing the Markowitz Market Index involves creating a portfolio with an optimal balance of risk and return. We consider the investors preference, which is then expressed as an indifference curve, based on risk taking, averse and neutral investors.

We first calculate the efficient frontier, using quadratic function

```
expected_returns = returns.mean()
cov_matrix = returns.cov()

# Defined using quadratic programming
def get_efficient_frontier(expected_returns, cov_matrix, num_points=100):
    n = len(expected_returns)
    w = cp.Variable(n)
    mu = expected_returns.values
    Sigma = cov_matrix.values

    portfolio_return = mu.T @ w
    portfolio_variance = cp.quad_form(w, Sigma)

    returns = np.linspace(expected_returns.min(), expected_returns.max(), num_points)
    risks = []

    for r in returns:
        constraints = [cp.sum(w) == 1, portfolio_return == r, w >= 0]
        problem = cp.Problem(cp.Minimize(portfolio_variance), constraints)
```

```

problem.solve()
risks.append(np.sqrt(portfolio_variance.value))

return risks, returns

risks, returns = get_efficient_frontier(expected_returns, cov_matrix)

```

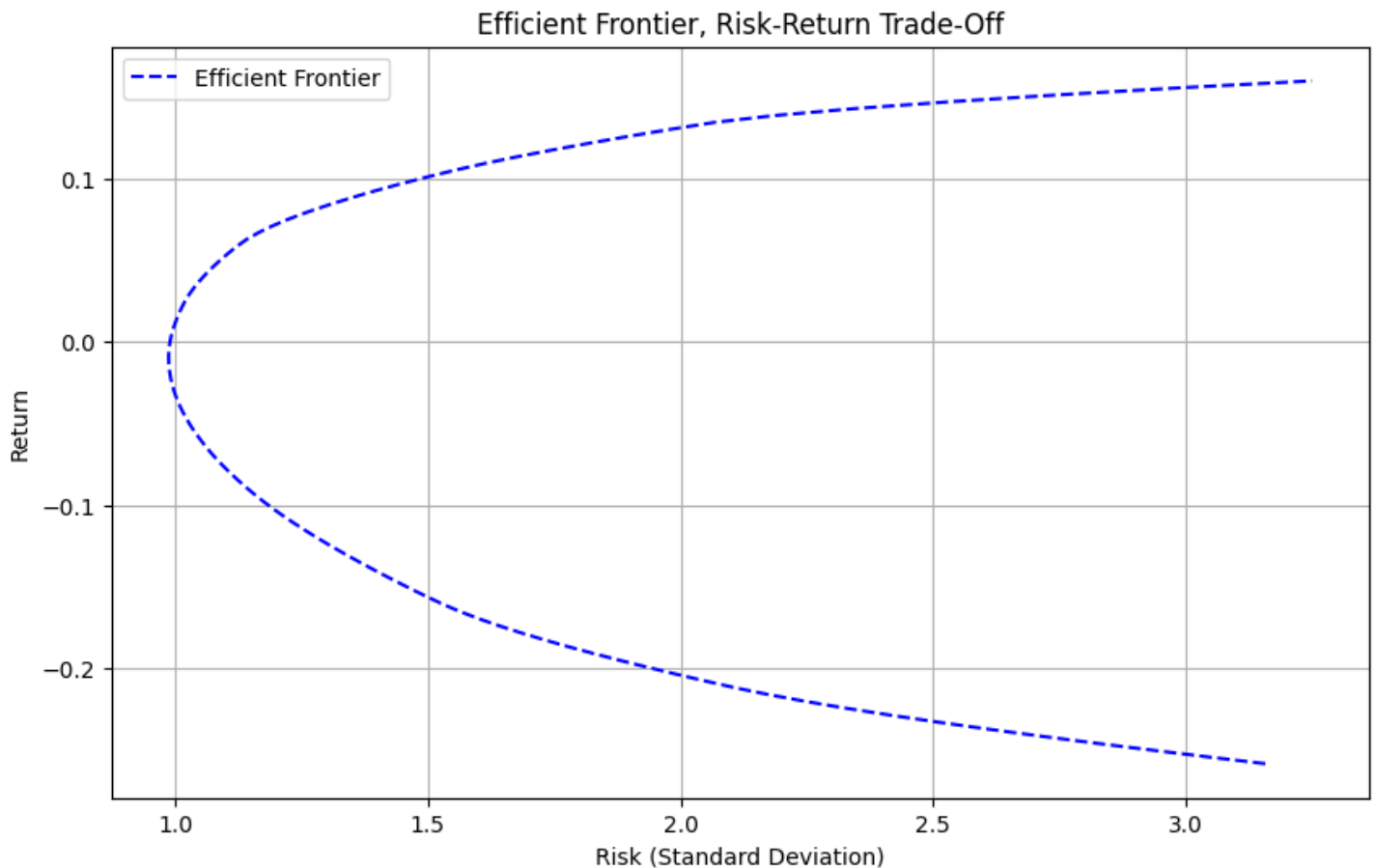
Plotting the efficient frontier curve gives us the visualization of risk-return trade-off

```

plt.figure(figsize=(10, 6))
plt.plot(risks, returns, 'b--', label='Efficient Frontier')
plt.xlabel('Risk (Standard Deviation)')
plt.ylabel('Return')
plt.title('Efficient Frontier, Risk-Return Trade-Off')
plt.legend()
plt.grid(True)
plt.show()

```

The output of which is:



We now plot the indifference curve, considering the investor(myself) as a risk-taking investor(moderate), with the utility function as $U = E[r] - 0.05 \cdot \sigma^2$. **(Assumption)**

The following is the code to plot for such a function

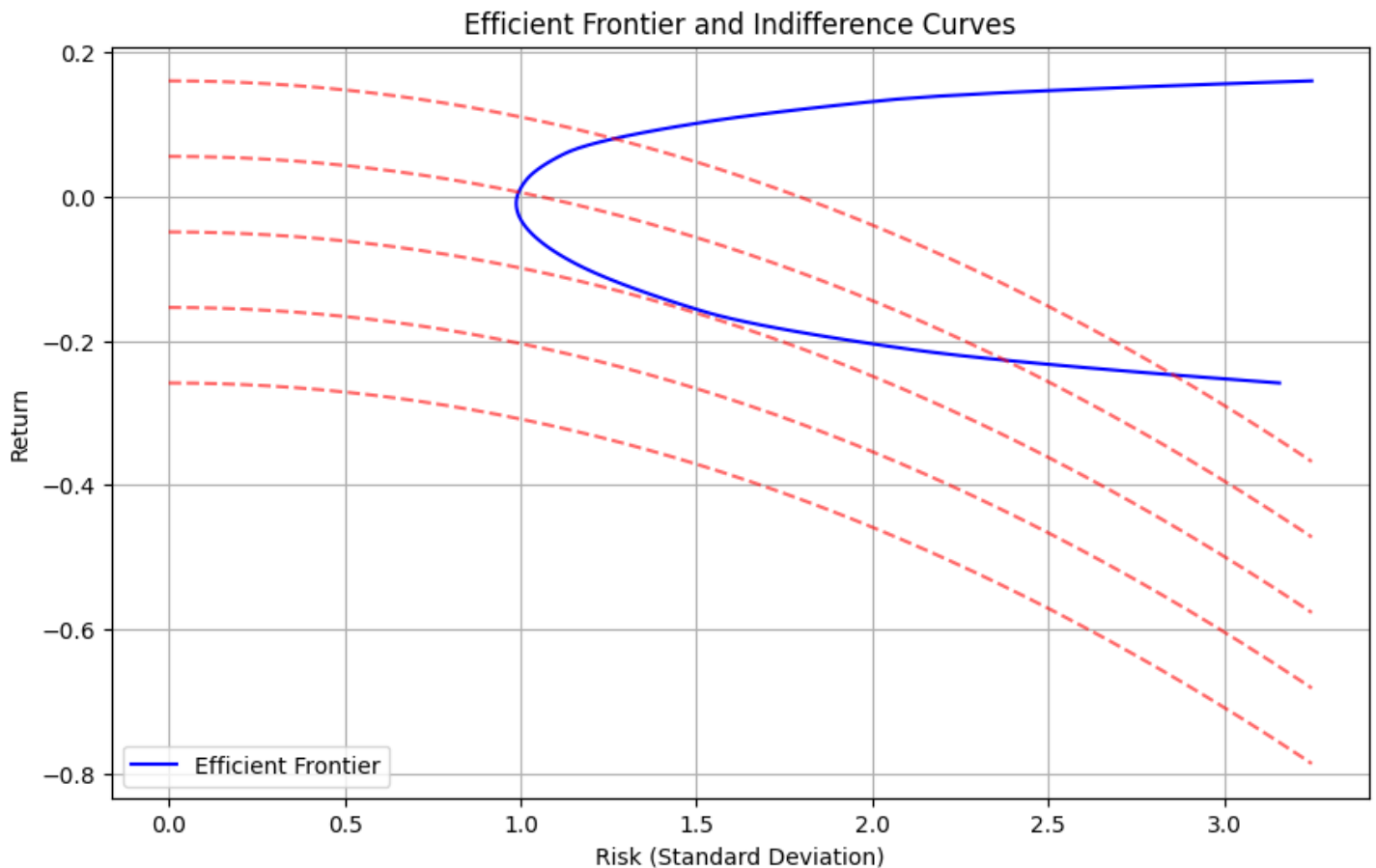
```
# Utility function used :  $U = E[r] - 0.05 \cdot \sigma^2$ 
def plot_indifference_curves(plt, num_curves=5, utility_levels=None):
    if utility_levels is None:
        utility_levels = np.linspace(min(returns), max(returns), num_curves)

    for U in utility_levels:
        risk = np.linspace(0, max(risks), 500)
        ret = U - (0.05 * risk**2)
        plt.plot(risk, ret, 'r--', alpha=0.6)

    plt.xlabel('Risk (Standard Deviation)')
    plt.ylabel('Return')
    plt.title('Efficient Frontier and Indifference Curves')
    plt.legend(['Efficient Frontier', 'Indifference Curves'])
    plt.grid(True)

plt.figure(figsize=(10, 6))
plt.plot(risks, returns, 'b-', label='Efficient Frontier')
plot_indifference_curves(plt)
plt.legend()
plt.show()
```

The output of which is



Constructing the Markowitz index for optimal weights

To do this we must find the optimal portfolio on the efficient frontier that maximizes my(investors) utility. This is where we use the indifference curve to find the same

```
# Identify optimal portfolio on the efficient frontier that maximizes the investors utility
def find_optimal_portfolio(expected_returns, cov_matrix, risk_aversion=0.05):
    n = len(expected_returns)
    w = cp.Variable(n)
    mu = expected_returns.values
    Sigma = cov_matrix.values

    portfolio_return = mu.T @ w
    portfolio_variance = cp.quad_form(w, Sigma)

    # Utility function
    utility = portfolio_return - risk_aversion * portfolio_variance

    constraints = [cp.sum(w) == 1, w >= 0]
    problem = cp.Problem(cp.Maximize(utility), constraints)
    problem.solve()

    return w.value, portfolio_return.value, cp.sqrt(portfolio_variance).value

optimal_weights, optimal_return, optimal_risk = find_optimal_portfolio(expected_returns,
cov_matrix, risk_aversion=0.05)

print("Optimal Weights:", optimal_weights)
print("Optimal Return:", optimal_return)
print("Optimal Risk:", optimal_risk)
```

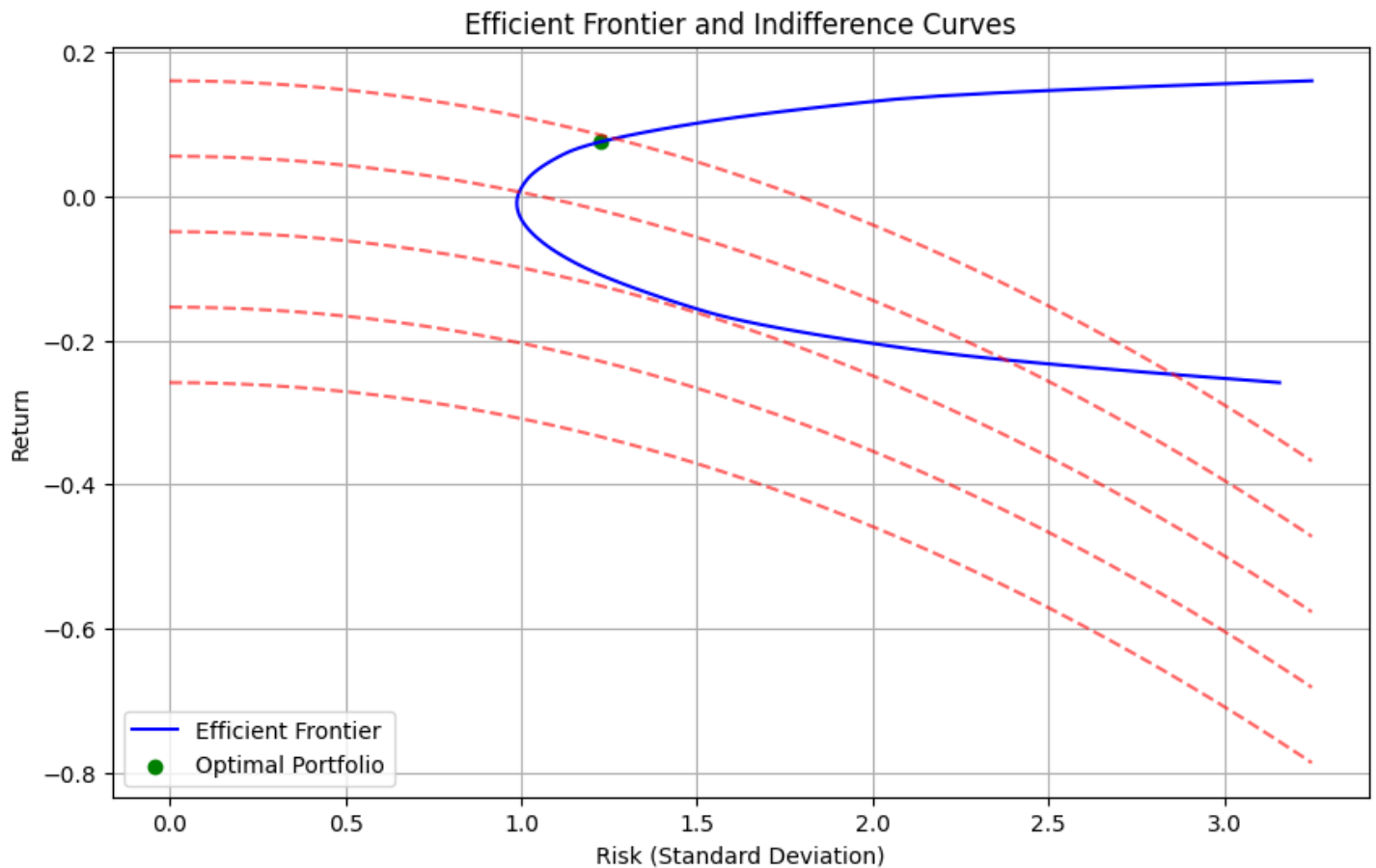
The output of which is given by

```
Optimal Weights: [ 6.07364645e-02 -2.93503231e-25  1.70285033e-22  1.08219930e-01
 8.58100053e-02  2.16077729e-23  2.23144315e-02  1.00472678e-23
 2.46094136e-24  3.89993531e-01  1.64542258e-01  1.68383380e-01
-4.71300098e-23]
Optimal Return: 0.07537917549278277
Optimal Risk: 1.2250581852189608
```

Plotting the optimal risk and return on the curve gives us the optimal point/portfolio

```
plt.figure(figsize=(10, 6))
plt.plot(risks, returns, 'b-', label='Efficient Frontier')
plot_indifference_curves(plt)
plt.scatter([optimal_risk], [optimal_return], color='g', marker='o', label='Optimal
Portfolio')
plt.legend()
plt.show()
```

The output of which is



4) Compare and discuss the weights assigned for the theoretical portfolio that you have identified with the weights of the Index. (3 marks)

The weights for the minimum variance portfolio and the Markowitz index are as follows

Company	Min Variance	Markowitz
ABB	0.124281	6.07364645e-02
AE	0.033918	-2.93503231e-25
AGE	0.056768	1.70285033e-22
AP	-0.051031	1.08219930e-01
BHEL	-0.048459	8.58100053e-02
CGP	0.129567	2.16077729e-23
JSW	-0.027256	2.23144315e-02
NHPC	-0.058114	1.00472678e-23
NTPC	0.181067	2.46094136e-24
PG	0.257356	3.89993531e-01
Sie	0.219924	1.64542258e-01
SE	0.061310	1.68383380e-01
TP	0.120670	-4.71300098e-23

We know that the Markowitz model seeks to balance the risk-return according to the investors specific risk aversion level. While the minimum variance portfolio is designed to minimize the risk without considering the returns explicitly.

Comparison of Weights:

The weights for Min Variances are clearly larger and have the constraint of adding up to 1. The weights from the Markowitz or mean variance model are much smaller in comparison. The trend for securities is not the same for both. While the min variance method weighs heavily on NTPC and TP, this is not the same for Markowitz model which weighs it relatively lower.

Parallely, the weights that are low for Min Variance like AP and BHEL, are not the same in the Markowitz model. The choice between the two will be up to the preference of the investor to make.

Minimum

Variance

Weights:

The weights, as we clearly see, are high for ABB India (ABB), Power Grid Corporation of India (PG), NTPC and Tata Powers (TP) when it comes to the minimum variance method. The weights are lower for Adani Power (AP), BHEL and NHPC.

Since we allowed negative weights in the minimum variance portfolio, which means short selling of stock is allowed. That is selling an asset that I currently do not own, with the intention of buying it back later at a lower price. The short position on AP, BHEL and NHPC allows us to reduce the overall portfolio variance and give us an advantage of mispricing or hedging. But this opens us up to additional risks such as those associated with leverage and margin requirements.

Markowitz Weights:

The weights are high for Adani Power (AP), Power Grid Corp (PG) and Suzlon Energy (SE). While they are low for Adani Energy (AE) and Tata Powers (TP).

The difference between the weighing for Min Variance and Markowitz weights tells us which securities to bet on, based on lowest possible risk (min Var) and an optimal balance between risk and return (Markowitz) for a moderately risk-taking investor.

Appendix:

Website to select the index – moneycontrol.com

Website for market price of securities - <https://www.bseindia.com/>

CSV files of the historical data and their individual returns – [Google drive link](#) (download the file **MCF_Assignment 2.xlsx - Returns** and use as input in the below code)

The code for all the above graphs, tables and models – [Drive link](#)