# Perbandingan Metode *Mean-Semivariance* dan *Mean Absolute Deviation* untuk Menentukan Portofolio Optimal Menggunakan *Python*

**Bilqis Khairun Nisa*, Onoy Rohaeni, Erwin Harahap**

Prodi Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Bandung, Indonesia.

*bilqiskhairun123@gmail.com, onoyrohaeni@gmail.com, erwin2h@unisba.ac.id

**Abstract.** Investment is an investment activity with the aim of making a profit. One of the investments in financial assets is stock investment. Stocks are a high-risk investment because stock prices fluctuate. To avoid risks that will affect returns when investing, it is necessary to form an optimal portfolio. Optimal portfolio is a portfolio that provides maximum return and has minimum risk. This study discusses the formation of an optimal portfolio using the Mean Semivariance and *Mean Absolute Deviation* methods. From the calculation results in this study the Mean Semivariance method obtained a return of 0.0035% and a risk of 0.080518%. Meanwhile, using the *Mean Absolute Deviation* method, a return of 0.000273% and a risk of 0.022276% are obtained.

**Keywords:** *Optimal Portfolio, Mean Semivariance, Mean Absolute Deviation.*

**Abstrak.** Investasi merupakan kegiatan menanamkan modal dengan tujuan mendapatkan keuntungan. Salah satu investasi pada aset keuangan yaitu investasi saham. Saham merupakan investasi yang berisiko tinggi karena harga saham yang fluktuatif. Untuk menghindari risiko yang akan mempengaruhi return saat berinvestasi, maka perlu membentuk portofolio optimal. Portofolio optimal merupakan portofolio yang memberikan return maksimum dan memiliki risiko minimum. Pada penelitian ini dibahas mengenai pembentukan portofolio optimal dengan menggunakan metode Mean Semivariance dan *Mean Absolute Deviation.* Dari hasil perhitungan pada penelitian ini metode Mean Semivariance memperoleh return sebesar 0.0035% dan risiko sebesar 0.080518%. Sedangkan dengan menggunakan metode *Mean Absolute Deviation* diperoleh return sebesar 0.000273% dan risiko sebesar 0.022276%.

**Kata Kunci:** *Portofolio Optimal, Mean Semivariance, Mean Absolute Deviation.*

Corresponding Author
Email: onoyrohaeni@gmail.com

## A. Pendahuluan

Investasi artinya menaruh dana atau melakukan komitmen dengan tujuan untuk memperoleh pengembalian ekonomi atau memperoleh hasil dari dana tersebut selama periode waktu tertentu [1][2]. Tujuan investasi umumnya yaitu untuk memperoleh tambahan kemanfaatan ekonomis dari jumlah uang yang telah dialokasikan pada suatu aset [3][4]. Salah satu investasi pada aset keuangan dengan risiko tinggi yaitu investasi saham. Saham merupakan salah satu komoditas keuangan yang sangat populer diperdagangkan di pasar modal [5].

Saham adalah salah satu instrumen investasi di pasar modal dengan return dan risiko yang cukup tinggi [6]. Saham merupakan investasi yang berisiko tinggi karena harga saham yang fluktuatif sehingga memungkinkan investor dapat memperoleh keuntungan dengan jumlah besar, tetapi dapat pula mengalami kerugian dengan jumlah yang tidak kecil. Untuk menghindari risiko yang akan mempengaruhi return saat berinvestasi, maka perlu membentuk portofolio yang optimal.

Untuk mewujudkan pembentukan portofolio yang optimal seorang investor perlu mempertimbangkan kinerja portofolionya. Seorang investor yang rasional tentu akan memilih portofolio yang optimal [7]. Portofolio optimal merupakan portofolio yang memberikan return maksimum dan memiliki risiko minimum. Pembentukan portofolio optimal diawali dengan melakukan analisis portofolio terlebih dahulu. Analisis portofolio digunakan untuk menentukan return yang akan diterima dan seberapa kecil risiko yang akan didapatkan. Salah satu metode untuk menganalisis portofolio optimal adalah metode *Mean-Semivariance* dan *Mean Absolute Deviation* (MAD).

Portofolio optimal diperoleh dimulai dengan menghitung nilai *realized return* dan *expected return*. Adapun rumus dari *realized return* adalah:

$$R_i = \frac{P_t - P_{t-1}}{P_{t-1}} \quad (1)$$

dimana $R_i$ merupakan return realisasi saham *i*, $P_t$ merupakan harga penutupan saham *i* pada bulan ke *t* dan $P_{t-1}$ merupakan harga penutupan saham *i* pada bulan ke *t-1*.

*Expected return* $\left(E(R_i)\right)$ atau tingkat keuntungan yang diharapakan adalah persentase rata-rata realisasi saham *i* [4]. Adapun rumus untuk menghitung *expected return* yaitu:

$$E(R_i) = \frac{\sum_{i=1}^{n} R_i}{n} \quad (2)$$

Dimana $E(R_i)$ merupakan *expected return* saham *i*, $R_i$ merupakan *realized return* saham *i* dan *n* banyak aset investasi. Pada perhitungan return portofolio digunakan persamaan:

$$E(R_p) = \sum_{i=1}^{n} \omega_i E(R_i) \quad (3)$$

Dimana $E(R_p)$ merupakan *expected return* portofolio, *n* merupakan banyak aset investasi, $\omega_i$ merupakan bobot investasi aset ke *i* dan $E(R_i)$ merupakan *expected return* saham *i*. Standar Deviasi digunakan untuk menghitung risiko saham perusahaan yang memiliki *expected return* positif. Rumus untuk menghitung standar deviasi yaitu:

$$\sigma_{SD} = \sqrt{\frac{\sum_{i=1}^{n} |R_i - E(R_i)|^2}{n}} \quad (4)$$

Dimana $R_i$ merupakan *realized return*, $E(R_i)$ merupakan *expected return* saham *i*, dan *n* banyak aset investasi.

Markowitz memperkenalkan model *mean-variance* pada tahun 1952. Metode *mean-variance* menggunakan mean untuk mempresentasikan nilai return dan variance untuk ukuran risiko. Metode *mean-variance* banyak dikritik oleh para peneliti karena asumsi pada model tersebut. Metode *mean-variance* mempunyai dua asumsi yang harus dipenuhi yaitu distribusi data return aset berdistribusi normal dan investor mempunyai fungsi utilitas kuadrat. Karena kelemahan yang dimiliki model *mean-variance*, para peneliti menawarkan metode lain dalam pembuatan portofolio optimal, yaitu metode *Mean-Semivariance*. Metode *Mean-Semivariance* menggunakan matriks *semivariance-semicovariance* yang memiliki sifat asimetrik dan endogen. Matriks dengan sifat tersebut lebih susah untuk dihitung dengan menggunakan algoritma numerik dalam penyelesaiannya [8]. Agar dapat menyelesaikan permasalahan tersebut, Estrada (2008) menawarkan solusi dengan mengubah matriks *semivariance-*

*semicovariance* menjadi simetrik dan eksogen dengan menggunakan pendekatan heuristic [9].

Konno & Yamazaki (1991) memperkenalkan optimasi portofolio *Mean Absolute Deviation* (MAD) sebagai alternatif metode Markowitz. Perhitungan nilai risiko menggunakan metode *Mean Absolute Deviation* (MAD) adalah menentukan rata-rata nilai mutlak penyimpangan MAD dari tingkat *realized return* terhadap *expected return* [10]. Permasalahan pembentukan portofolio metode *Mean Absolute Deviation* merupakan masalah linear programming.

Pada dasarnya perhitungan pada portofolio optimal saham dapat dihitung dengan menggunakan aplikasi Microsoft Excel, namun pada penelitian ini digunakan bahasa pemrograman *Python*. Bahasa pemrograman *Python* dibuat untuk membantu menyelesaikan permasalahan terkait portofolio optimal saham. Bahasa pemrograman *Python* dapat dikembangkan oleh siapa saja karena bersifat *Open Source*, atau bahasa pemrograman ini dapat digunakan secara gratis, tanpa lisensi dan dapat dikembangkan semampu yang dapat dilakukan.

Berdasarkan latar belakang yang telah diuraikan, maka perumusan masalah dalam penelitian ini sebagai berikut: "Bagaimana menentukan portofolio optimal menggunakan metode *Mean-Semivariance* dan *Mean Absolute Deviation* (MAD) dengan bantuan *Python* serta membandingkan portofolio optimal dari hasil yang telah diperoleh?". Selanjutnya, tujuan dalam penelitian ini diuraikan dalam pokok-pokok sbb.

1. Mengetahui dan membandingkan portofolio optimal dengan menggunakan metode *Mean-Semivariance* dan *Mean Absolute Deviation* (MAD).
2. Menggunakan bahasa pemrograman *Python* pada proses perhitungan portofolio optimal.

## B. Metodologi Penelitian

Peneliti menggunakan metode *Mean-Semivariance* dan *Mean Absolute Deviation* (MAD). Metode *Mean-Semivariance* diawai dengan menghitung nilai *semivariance return* dengan persamaan [7]:

$$S_{ij}^{B} = \frac{1}{T}\sum_{t=1}^{T}[min(R_{it} - B, 0)]^2 \tag{5}$$

dimana $S_{ij}^{B}$ merupakan *semivariance*, $T$ merupakan jumlah observasi, $R_{it}$ merupakan *return* saham *i* pada periode *t*, dan $B$ merupakan *return benchmark*.

*Semicovariance return* dihitung dengan menggunakan persamaan:

$$S_{iB}^{2} = \frac{1}{T}\sum_{t=1}^{T}[min(R_i - B, 0) \cdot min(R_j - B, 0)] \tag{6}$$

dimana $S_{iB}^{2}$ merupakan semicovariance, $R_i$ merupakan *return* saham *i*, dan $R_j$ merupakan *return* saham *j* [3]. Selanjutnya matriks *semivariance-semicovariance* yang dibentuk dari nilai *semivariance* dan nilai *semicovariance* sebagai berikut:

$$\Delta_{msv} = \begin{bmatrix} S_{1\,B}^{2} & S_{12B} & S_{13B} & S_{14B} \\ S_{21B} & S_{2\,B}^{2} & S_{23B} & S_{24B} \\ S_{31B} & S_{32B} & S_{3\,B}^{2} & S_{34B} \\ S_{41B} & S_{42B} & S_{43B} & S_{4\,B}^{2} \end{bmatrix} \tag{7}$$

Kemudian matriks *semivariance-semicovariance* dihitung invers matriksnya. Berikutnya, menghitung bobot masing-masing saham dengan persamaan sebagai berikut:

$$\omega = \frac{\underline{\Delta m}^{-1}1}{\underline{1}^{T}\underline{\Delta m}^{-1}1} \tag{8}$$

Perhitungan *return* portofolio pada metode *mean semivariance* dengan persamaan (3) dan risiko portofolio dihitung menggunakan persamaan:

$$\sigma_p{}^2 = [\omega_1 \dots \omega_n] \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_2 & \cdots & \sigma_{nn} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_n \end{bmatrix} \tag{9}$$

Metode *Mean Absolute Deviation* merupakan perhitungan risiko yang menggunakan nilai MAD, dengan menggunakan persamaan:

$$MAD = \frac{\sum_{t=1}^{T} |x_i - \bar{x}|}{T} \tag{10}$$

Metode *Mean Absolute Deviation* menggunakan linear programming dalam menghitung bobot portofolio. Pada perhitungan bobot portofolio metode *Mean Absolute Deviation* membutuhkan nilai *expected return*, *return* minimal, dan nilai MAD. Nilai MAD yang telah diperoleh akan dibentuk menjadi fungsi tujuan [9], maka diperoleh fungsi tujuan sebagai berikut:

$$\sigma(x) = MAD_1 x_1 + MAD_2 x_2 + \cdots + MAD_n x_n \tag{11}$$

Portofolio metode *Mean Absolute Deviation* memiliki tiga fungsi kendala, dimana kendala pertama merupakan representasi dari nilai *return* portofolio yang akan dibentuk berdasarkan niliai *return* minimalnya. Fungsi kendala pertama dapat dituliskan sebagai berikut:

$$\bar{R}_1 x_1 + \bar{R}_2 x_2 + \cdots + \bar{R}_n x_n \geq R \tag{12}$$

dimana $\bar{R}_n$ merupakan rata-rata *expected return* masing-masing saham dan $R$ merupakan *return* minimal. *Return* minimal diperoleh dari nilai *expected return* masing-masing saham dibagi dengan banyaknya saham [9], sebagai berikut:

$$R = \frac{r_1 + r_2 + \cdots + r_n}{n} \tag{13}$$

Kendala kedua diperoleh melalui representasi dari seluruh perusahaan yang sahamnya akan diinvestasikan, seperti pada persamaan berikut :

$$x_1 + x_2 + \cdots + x_n = 1 \tag{14}$$

Kendala ketiga diperoleh dengan merepresentasikan bobot invetasi masing-masing aset yang tidak bernilai negatif dan tidak lebih dari nilai yang sudah ditentukan $(\mu_i)$. Pada bobot maksimal yang diinvestasikan $(\mu_i)$ nilai disesuaikan dengan banyak sahan yang diinginkan [9]. Kendala yang ketiga dapat dituliskan sebagai berikut :

$0 \leq x_i \leq u_i$, dimana $i = 1,2, \dots, n$

dimana $u_i$ merupakan nilai bobot maksimal yang diinvestasikan investor. Masalah program linear pada *Mean Absolute Deviation* ini diselesaikan dengan metode simpleks. *Return* portofolio pada metode *Mean Absolute Deviation* dihitung menggunakan persamaan (2.3). untuk menghitung risiko portofolio pada metode *Mean Absolute Deviation* digunakan persamaan sebagai berikut:

$$\sigma_p = \sum_{i=1}^{n} w_i \, . \, \sigma_{SD}$$

Dimana $w_i$ merupakan bobot saham dan $\sigma_{SD}$ merupakan standar deviasi.

Pada penelitian ini mengambil data penutupan harga saham yang terdaftar di IDX30

yang bersumber dari https://www.idx.co.id/. Teknik pengambilan data pada penelitian ini menggunakan metode *purposive sampling*. Adapun kriteria-kriteria yang digunakan dalam penentuan sampel pada penelitian ini adalah sebagai berikut:

1. Perusahaan yang tergabung di BEI dan terdaftar sebagai anggota IDX30 yang tidak pernah keluar dari daftar selama periode 01 Januari 2020 – 31 Desember 2022.
2. Perusahaan tersebut menerbitkan laporan keuangan pada situs resmi Bursa Efek Indonesia (www.idx.co.id).
3. Perusahaan yang tidak melakukan *Corporate Action* (*Stock, Split, Merger,* dan Akuisisi) selama periode 01 Januari 2020 - 31 Desember 2022.
4. Perusahaan tersebut memiliki data yang lengkap mengenai *earning per share* (EPS) dan *return on equity* (ROA) dan melakukan pembayaran deviden secara rutin selama periode 01 Januari 2020-31 Desember 2022.

Dari 30 saham yang sudah diteliti menggunakan metode *purposive sampling*, didapat 5 saham yang akan dijadikan sampel pada penelitian ini, yang diuraikan pada Tabel 1. Data yang telah diperoleh kemudian diolah dengan bantuan *Python* sehingga menghasilkan *return* dan risiko portofolio dari masing-masing saham perusahaan yang telah dipilih.

**Tabel 1.** Sampel Penelitian

| No | Kode Saham | Nama Saham |
|---|---|---|
| 1 | BBNI | Bank Negara Indonesia (Persero) Tbk. |
| 2 | TLKM | Telkom Indonesia (Persero) Tbk. |
| 3 | ASII | Astra Internasional Tbk |
| 4 | ICBP | Indofood CBP Sukses Makmur Tbk. |
| 5 | KLBF | Kalbe Farma Tbk. |

## C. Hasil Penelitian dan Pembahasan

### *Import Library*

Sebelum mengolah data dengan bantuan *Python*, terlebih dahulu dilakukan *import* beberapa *library* yang dibutuhkan yaitu *Pandas*, *NumPy*, dan *SciPy*, seperti di bawah ini.

```
#Import library
import pandas as pd
import numpy as np
import math
from scipy.optimize import minimize
```

**Gambar 1**. *Import Library*

### Input Data Harga Saham

Input data harga saham dilakukan dengan menggunakan *library Pandas* yang telah diinstal. Gunakan fungsi *"pd.read_excel()"* untuk membaca data dari file Excel. Fungsi ini menerima argumen berupa *path* file Excel yang ini dibaca. Pada penelitian ini file Excel bernama "SAHAM.xlsx" dan berada di direktori yang sama dengan skrip *Python*. Data yang telah diinput kemudian diseleksi untuk memilih kolom yang akan digunakan dengan menggunakan fungsi *"selected_stocks"* yang berisi data yang akan digunakan. Cara input data dan memilih kolom yang akan digunakan pada penelitian ini, seperti di bawah ini.

```python
#Membaca data harga penutupan saham harian
data = pd.read_excel('SAHAM.xlsx')

#memilih kolom
selected_stocks = ['BBNI', 'TLKM', 'ASII', 'ICBP', 'KLBF']
df = data[selected_stocks]
print(df)
```

```
     BBNI  TLKM  ASII   ICBP  KLBF
0    7775  3910  6875  11150  1615
1    7800  3980  6950  11250  1635
2    7625  3960  6750  11500  1640
3    7550  3940  6775  11500  1635
4    7425  3900  6775  11550  1585
..    ...   ...   ...    ...   ...
730  9325  3750  5650  10150  2070
731  9300  3810  5675  10075  2030
732  9225  3730  5625  10175  2030
733  9225  3780  5650  10175  2090
734  9225  3750  5700  10000  2090

[735 rows x 5 columns]
```

**Gambar 2.** Input Data Harga Saham

**Menghitung *realized return***

Perhitungan *realized return* pada harga penutupan saham dari 5 perusahaan yang terdaftar di IDX30 periode 01 Januari 2020 sampai 31 Desember 2022, dengan bantuan *Python* menggunakan metode *".pct_change()"*. Metode ".pct_change memiliki fungsi untuk menghitung perubahan antara elemen-elemen yang menghasilkan seri baru *"returns"*. Kemudian digunakan metode *".dropna()"* untuk menghapus baris dengan nilai NaN dari seri *"returns"*. Perhitungan *realized return* dengan menggunakan *Python* seperti di bawah ini.

```python
#Menghitung realized return
returns = pd.DataFrame()
returns = df.pct_change()
returns.dropna(inplace=True)
print(returns)
```

```
          BBNI      TLKM      ASII      ICBP      KLBF
1     0.003215  0.017903  0.010909  0.008969  0.012384
2    -0.022436 -0.005025 -0.028777  0.022222  0.003058
3    -0.009836 -0.005051  0.003704  0.000000 -0.003049
4    -0.016556 -0.010152  0.000000  0.004348 -0.030581
5     0.037037  0.015385  0.014760 -0.002165  0.009464
..         ...       ...       ...       ...       ...
730   0.000000 -0.007937 -0.004405  0.012469  0.014706
731  -0.002681  0.016000  0.004425 -0.007389 -0.019324
732  -0.008065 -0.020997 -0.008811  0.009926  0.000000
733   0.000000  0.013405  0.004444  0.000000  0.029557
734   0.000000 -0.007937  0.008850 -0.017199  0.000000

[734 rows x 5 columns]
```

**Gambar 3.** *Realized return*

**Menghitung *Expected return* dan Standar Deviasi**

Perhitungan *expected return* dengan *Python* menggunakan metode *".mean()"*. Metode *".mean()*" digunakan untuk menghitung rata-rata realisasi saham. Pada Perhitungan standar deviasi dilakukan dengan menggunakan metode *".std()"*, yang digunakan untuk menghitung simpangan baku dari seri *returns*. Perhitungan *expected return* dan stadar deviasi dengan menggunakan *Python* seperti di bawah ini.

```
#Menghitung Expected Return dan Standar Deviasi saham
hasil = pd.DataFrame()
expected_returns = returns.mean()
standar_deviasi = returns.std()
print("Expected Return")
print(expected_returns)
print("Standar Deviasi")
print(standar_deviasi)
```

```
Expected Return
BBNI    0.000548
TLKM    0.000160
ASII    0.000019
ICBP    0.000023
KLBF    0.000616
dtype: float64
Standar Deviasi
BBNI    0.025164
TLKM    0.020934
ASII    0.023475
ICBP    0.018607
KLBF    0.023199
dtype: float64
```

**Gambar 4.** *Expected return* dan Standar Deviasi

**Tabel 2.** *Expected return* dan standar deviasi

| | ***Expected return*** | **Standar Deviasi** |
|---|---|---|
| $\bar{R}_1$ | 0.000548 | 0.025164 |
| $\bar{R}_2$ | 0.000160 | 0.020934 |
| $\bar{R}_3$ | 0.000019 | 0.023475 |
| $\bar{R}_4$ | 0.000023 | 0.018607 |
| $\bar{R}_5$ | 0.000616 | 0.023199 |

**Pembentukan Portofolio Optimal *Mean Semivariance***

Metode *mean semivariance* dihitung dengan pendekatan heuristik menggunakan matriks *semivariance-semicovariance* yang dibentuk dari nilai *semivaraince* dan nilai *semicovariance* masing-masing saham. Untuk menghitung nilai *semivariance* pada *Python* dibantu dengan menggunakan fungsi ”np.sum” untuk membantu menghitung nilai *semivariance* dalam menjumlahkan elemen pada perhitungan nilai *semivariance*. Hasil perhitungan nilai semivariance dengan menggunakan *Python* seperti di bawah ini.

```
#Menghitung nilai semivariance
N = len(returns)
semivariance_bbni = np.sum((returns['BBNI'] - 0) ** 2)
semivariance_tlkm = np.sum((returns['TLKM'] - 0) ** 2)
semivariance_asii = np.sum((returns['ASII'] - 0) ** 2)
semivariance_icbp = np.sum((returns['ICBP'] - 0) ** 2)
semivariance_klbf = np.sum((returns['KLBF'] - 0) ** 2)
print("Semivariance BBNI:", semivariance_bbni)
print("Semivariance TLKM:", semivariance_tlkm)
print("Semivariance ASII:", semivariance_asii)
print("Semivariance ICBP:", semivariance_icbp)
print("Semivariance KLBF:", semivariance_klbf)
```

```
Semivariance BBNI: 0.46439018853886327
Semivariance TLKM: 0.3212549624166834
Semivariance ASII: 0.40394412286110726
Semivariance ICBP: 0.253778270586259
Semivariance KLBF: 0.39475769726875576
```

**Gambar 5.** Portofolio Optimal *Mean Semivariance*

Selanjutnya menghitung nilai *semicovarian* dengan bantuan *Python* menggunakan fungsi *”np.sum”*, seperti di bawah ini.

```
#Menghitung nilai semikovariance
semikovariance_bbni_tlkm = np.sum((returns['BBNI'] - 0) * (returns['TLKM'] - 0))
semikovariance_bbni_asii = np.sum((returns['BBNI'] - 0) * (returns['ASII'] - 0))
semikovariance_bbni_icbp = np.sum((returns['BBNI'] - 0) * (returns['ICBP'] - 0))
semikovariance_bbni_klbf = np.sum((returns['BBNI'] - 0) * (returns['KLBF'] - 0))
semikovariance_tlkm_asii = np.sum((returns['TLKM'] - 0) * (returns['ASII'] - 0))
semikovariance_tlkm_icbp = np.sum((returns['TLKM'] - 0) * (returns['ICBP'] - 0))
semikovariance_tlkm_klbf = np.sum((returns['TLKM'] - 0) * (returns['KLBF'] - 0))
semikovariance_asii_icbp = np.sum((returns['ASII'] - 0) * (returns['ICBP'] - 0))
semikovariance_asii_klbf = np.sum((returns['ASII'] - 0) * (returns['KLBF'] - 0))
semikovariance_icbp_klbf = np.sum((returns['ICBP'] - 0) * (returns['KLBF'] - 0))
print("Semikovariance BBNI_TLKM:", semikovariance_bbni_tlkm)
print("Semikovariance BBNI_ASII:", semikovariance_bbni_asii)
print("Semikovariance BBNI_ICBP:", semikovariance_bbni_icbp)
print("Semikovariance BBNI_KLBF:", semikovariance_bbni_klbf)
print("Semikovariance TLKM_ASII:", semikovariance_tlkm_asii)
print("Semikovariance TLKM_ICBP:", semikovariance_tlkm_icbp)
print("Semikovariance TLKM_KLBF:", semikovariance_tlkm_klbf)
print("Semikovariance ASII_ICBP:", semikovariance_asii_icbp)
print("Semikovariance ASII_KLBF:", semikovariance_asii_klbf)
print("Semikovariance ICBP_KLBF:", semikovariance_icbp_klbf)
```

```
Semikovariance BBNI_TLKM: 0.16419199298208 26
Semikovariance BBNI_ASII: 0.2405939548374089
Semikovariance BBNI_ICBP: 0.1353219979987118
Semikovariance BBNI_KLBF: 0.14441255119857366
Semikovariance TLKM_ASII: 0.1488287559640386
Semikovariance TLKM_ICBP: 0.09484762197054757
Semikovariance TLKM_KLBF: 0.10458446264974042
Semikovariance ASII_ICBP: 0.1041651152024059
Semikovariance ASII_KLBF: 0.12702146046061918
Semikovariance ICBP_KLBF: 0.09928679070793088
```

**Gambar 6.** Nilai *Semicovarians*

Setelah melakukan perhitungan nilai *semivariance* dan nilai *semicovariance*, kemudian dibentuk matriks *semivariance-semikovariance* dan dihitung invers matriksnya. Pembentukan matriks *semivariance-semicovariance* menggunakan *Python* dibantu dengan fungsi *"np.array"* yang terdapat dalam *library NumPy*. Fungsi *"np.array"* digunakan dalam membentuk matriks dengan menggunakan list sebagai argumen. List yang digunakan dalam matriks *semivariance-semicovariance* merupakan nilai *semivariance* dan nilai *semicovariance*. dan perhitungan matriksnya menggunakan *Python*. Penghitungan invers matriks *semivariance-semicovariance* digunakan fungsi *"np.linalg.inv"* yang disediakan *NumPy*. Hasil perhitungan matriks *semivaraince-semicovariance* dan inversnya dengan menggunkan *Python*, seperti di bawah ini.

```
#Membentuk Invers Matriks Semivaraince-Semikovaraince
matrix = np.array([[semivariance_bbni, semikovariance_bbni_tlkm, semikovariance_bbni_asii, semikovariance_bbni_icbp, semikovariar
                   [semikovariance_bbni_tlkm, semivariance_tlkm, semikovariance_tlkm_asii, semikovariance_tlkm_icbp, semikovariar
                   [semikovariance_bbni_asii, semikovariance_tlkm_asii, semivariance_asii, semikovariance_asii_icbp, semikovariar
                   [semikovariance_bbni_icbp, semikovariance_tlkm_icbp, semikovariance_asii_icbp, semivariance_icbp, semikovariar
                   [semikovariance_bbni_klbf, semikovariance_tlkm_klbf, semikovariance_asii_klbf, semikovariance_icbp_klbf, semi
matrix_invers = np.linalg.inv(matrix)
print('Matriks Invers')
print(matrix_invers)
```

```
Matriks Invers
[[ 3.5709727  -0.7436482  -1.50486691 -0.84734383 -0.4119965 ]
 [-0.7436482   4.19855353 -0.79822308 -0.68405171 -0.41139811]
 [-1.50486691 -0.79822308  3.87852574 -0.33391756 -0.40201622]
 [-0.84734383 -0.68405171 -0.33391756  5.04745283 -0.67084805]
 [-0.4119965  -0.41139811 -0.40201622 -0.67084805  3.09099572]]
```

**Gambar 7.** Matriks *Semivaraince-Semicovariance*

Berikutnya, untuk menghitung bobot masing-masing saham dengan menggunakan *Python*. Perhitungan bobot pada *Python* digunakan *"np.ones"* untuk membuat *array* dengan elemen yang semuanya bernilai 1, dan operator "@" digunakan untuk melakukan operasi perkalian matriks. Perhitungan bobot dengan *Python* seperti di bawah ini.

```
#Menghitung Bobot masing-masing saham
expected_return = 0.20
ones_vector = np.ones((5, 1))
weights = matrix_invers @ ones_vector*expected_return/(ones_vector.T @ matrix_invers @ ones_vector)
weights = np.round(weights.flatten(), 6)
print("Bobot Saham")
print("BBNI:", weights[0])
print("TLKM:", weights[1])
print("ASII:", weights[2])
print("ICBP:", weights[3])
print("KLBF:", weights[4])
```

```
Bobot Saham
BBNI: 0.002046
TLKM: 0.050608
ASII: 0.027213
ICBP: 0.081405
KLBF: 0.038728
```

**Gambar 8.** Bobot Matriks

Berdasarkan bobot yang telah diperoleh sebelumnya, maka dapat dilakukan perhitungan *return* masing-masing saham. Perhitungan *return* masing-masing saham dengan *Python* dibantu dengan menggunakan fungsi *"np.sum"* untuk menjumlahkan elemen dalam *array*. *Array* yang digunakan pada perhitungan *return* masing-masing saham merupakan bobot dan *expected return* yang sebelumnya sudah diperoleh.

```
#Menghitung return portofolio masing masing saham
returns_portofolio_bbni = np.sum(weights[0] * expected_returns['BBNI'])
returns_portofolio_tlkm = np.sum(weights[1] * expected_returns['TLKM'])
returns_portofolio_asii = np.sum(weights[2] * expected_returns['ASII'])
returns_portofolio_icbp = np.sum(weights[3] * expected_returns['ICBP'])
returns_portofolio_klbf = np.sum(weights[4] * expected_returns['KLBF'])
print("Return Portofolio")
print("BBNI:", returns_portofolio_bbni)
print("TLKM:", returns_portofolio_tlkm)
print("ASII:", returns_portofolio_asii)
print("ICBP:", returns_portofolio_icbp)
print("KLBF:", returns_portofolio_klbf)
```

```
Return Portofolio
BBNI: 1.120347192647079e-06
TLKM: 8.081720368331315e-06
ASII: 5.249231515418968e-07
ICBP: 1.8985899214114415e-06
KLBF: 2.3843268940565326e-05
```

**Gambar 9.** Bobot dan *Expected return*

Perhitungan *return* ekspektasi menggunakan *Python* dibantu dengan fungsi *"np.sum"* untuk menjumlahkan *return* portofolio yang telah diperoleh sebelumnya. Sehingga diperoleh *return* ekspektasi portofolio dengan menggunakan metode *mean semivariance* adalah 0.0035%, seperti di bawah ini.

```
#Menghitung Return Ekspektasi Portofolio
returns_portofolio = [returns_portofolio_bbni, returns_portofolio_tlkm, returns_portofolio_asii, returns_portofolio_icbp, returns
returns_ekspektasi = np.round(np.sum(returns_portofolio), 6)
returns_ekspektasi_persen = returns_ekspektasi * 100
print("Return:", f"{returns_ekspektasi_persen}%")
```

```
Return: 0.0034999999999999996%
```

**Gambar 10.** *Return* Ekspektasi Portofolio

Perhitungan risiko portofolio menggunakan *Python* dilakukan dengan menggunakan *"np.sqrt"* yang berfungsi menghitung akar kuadrat dan *"np.dot"* yang berfungsi menghitung operasi perkalian matriks. Hasil perhitungan risiko portofolio sebesar 0.080518%, seperti di bawah ini.

```
#Menghitung risiko portofolio
portofolio_risk = np.round(np.sqrt(np.dot(weights, np.dot(matrix, weights.T))), 6)
portofolio_risk_persen = portofolio_risk * 100
print("Risiko Portofolio:", f"{portofolio_risk}%")
```

```
Risiko Portofolio: 0.080518%
```

**Gambar 11.** Risiko Portofolio

**Pembentukan Portofolio Optimal *Mean Absolute Deviation*, Menghitung Return Minimal**

*Return* minimal diperoleh dari nilai rata-rata *expected return* seluruh saham. Pada perhitungan *return* minimal dengan menggunakan *Python* digunakan fungsi *".mean()"*. *Return* miniaml yang telah diperoleh kemudian dibulatkan dengan menggunakan fungsi *"np.round"*. Perhitungan *return* minimal dengan menggunakan *Python*, seperti di bawah ini.

```
#Menghitung nilai return minimal
returns_minimal = pd.DataFrame()
returns_minimal = np.round(expected_returns.mean(),6)
print(returns_minimal)
```

```
0.000273
```

**Gambar 12.** *Return* Minimal

**Membentuk Fungsi Kendala**

Pada metode *Mean Absolute Deviation* dibentuk tiga kendala. Kendala yang pertama dibentuk dari nilai *expected return* dan *return* minimal. Berdasarkan persamaan (2.11), $\bar{R}$ merupakan *expected return*, $x$ merupakan saham yang digunakan dan $R$ merupakan *return* minimal. Kendala pertama ditulis sebagai berikut:

$$0.000548x_1 + 0.000160x_2 + 0.000019x_3 + 0.000023x_4 + 0.000616x_5 \geq 0.000273$$

Kendala yang kedua akan dibentuk dari seluruh perusahaan yang sahamnya akan diinvestasika. $x_1$ untuk saham BBNI, $x_2$ untuk saham TLKM, $x_3$ untuk saham ASII, $x_4$ untuk saham ICBP, $x_5$ untuk saham KLBF. Kendala kedua ditulis sebagai berikut:

$$x_1 + x_2 + x_3 + x_4 + x_5 = 1$$

Kendala ketiga dibentuk berdasarkan banyaknya saham yang diinginkan. Karena pembentukan portofolio terdiri dari lima saham, maka diambil $\mu_i = 20\%$ berdasarkan persamaan (2.14) didapatkan kendala ketiga sebagai berikut:

$$x_i \leq 20\%, \qquad i = 1,2,3,4,5$$

**Menghitung Nilai *Mean Absolute Deviation***

Nilai *Mean Absolut Deviation* (MAD) digunakan untuk koefisien pada fungsi tujuan dalam pembentukan portofolio dengan menggunakan metode MAD. Pada perhitungan MAD dengan *Python* menggunakan *"np.abs"* yang berfungsi untuk menghitung nilai *absolute*. Kemudian dijumlahkan dengan menggunakan *"np.sum"*. Nilai MAD, seperti di bawah ini.

```
#Menghitung nilai MAD
deviation_bbni = returns['BBNI'] - expected_returns['BBNI']
deviation_tlkm = returns['TLKM'] - expected_returns['TLKM']
deviation_asii = returns['ASII'] - expected_returns['ASII']
deviation_icbp = returns['ICBP'] - expected_returns['ICBP']
deviation_klbf = returns['KLBF'] - expected_returns['KLBF']
mad_bbni = np.round(np.mean(np.abs(deviation_bbni)), 6)
mad_tlkm = np.round(np.mean(np.abs(deviation_tlkm)), 6)
mad_asii = np.round(np.mean(np.abs(deviation_asii)), 6)
mad_icbp = np.round(np.mean(np.abs(deviation_icbp)), 6)
mad_klbf = np.round(np.mean(np.abs(deviation_klbf)), 6)
print("Nilai MAD:")
print("MAD BBNI:", mad_bbni)
print("MAD TLKM:", mad_tlkm)
print("MAD ASII:", mad_asii)
print("MAD ICBP:", mad_icbp)
print("MAD KLBF:", mad_klbf)
```

```
Nilai MAD:
MAD BBNI: 0.017244
MAD TLKM: 0.014836
MAD ASII: 0.016904
MAD ICBP: 0.012585
MAD KLBF: 0.015811
```

**Gambar 13.** *Mean Absolute Deviation*

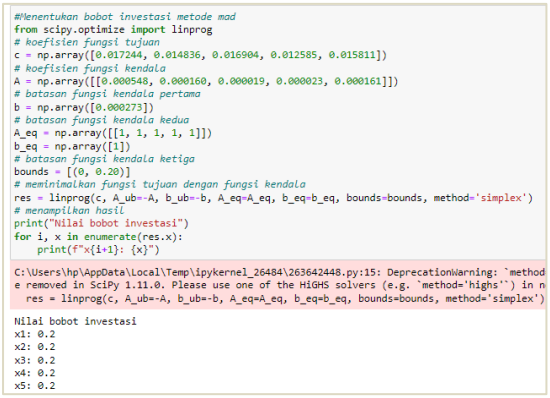
**Membentuk Fungsi Tujuan**

Fungsi tujuan dibentuk dari nilai MAD yang telah diperoleh sebelumnya. Berdasarkan persamaan (2.10), $MAD_1x_1$ merupakan nilai MAD BBNI, $MAD_2x_2$ merupakan nilai MAD TLKM, $MAD_3x_3$ merupakan nilai MAD ASII, $MAD_4x_4$ merupakan nilai MAD ICBP, $MAD_5x_5$ merupakan nilai MAD KLBF. Fungsi tujuan dapat ditulis sebagai berikut:

$$\sigma(x) = 0.017224x_1 + 0.014836x_2 + 0.016904x_3 + 0.012585x_4 + 0.015811x_5$$

**Bobot Investasi Metode *Mean Absolute Deviation***

Perhitungan bobot investasi diperoleh dengan meminimalkan fungsi tujuan dengan tiga kedala yang telah dibentuk. Perhitungan bobot investasi menggunakan *Python* dibantu dengan fungsi *"np.array"* dan fungsi *"linprog"* untuk membantu perhitungan dengan metode simpleks, seperti di bawah ini.



```
#Menentukan bobot investasi metode mad
from scipy.optimize import linprog
# koefisien fungsi tujuan
c = np.array([0.017244, 0.014836, 0.016904, 0.012585, 0.015811])
# koefisien fungsi kendala
A = np.array([[0.000548, 0.000160, 0.000019, 0.000023, 0.000161]])
# batasan fungsi kendala pertama
b = np.array([0.000273])
# batasan fungsi kendala kedua
A_eq = np.array([[1, 1, 1, 1, 1]])
b_eq = np.array([1])
# batasan fungsi kendala ketiga
bounds = [(0, 0.20)]
# meminimalkan fungsi tujuan dengan fungsi kendala
res = linprog(c, A_ub=-A, b_ub=-b, A_eq=A_eq, b_eq=b_eq, bounds=bounds, method='simplex')
# menampilkan hasil
print("Nilai bobot investasi")
for i, x in enumerate(res.x):
    print(f"x{i+1}: {x}")
```

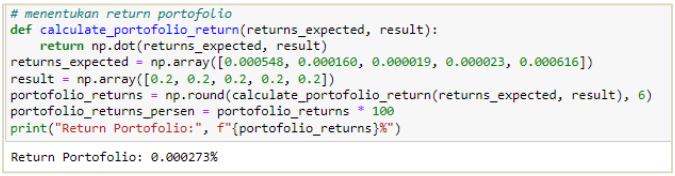
```
C:\Users\hp\AppData\Local\Temp\ipykernel_26484\263642448.py:15: DeprecationWarning: `method
e removed in SciPy 1.11.0. Please use one of the HiGHS solvers (e.g. `method='highs'`) in n
  res = linprog(c, A_ub=-A, b_ub=-b, A_eq=A_eq, b_eq=b_eq, bounds=bounds, method='simplex')
```

```
Nilai bobot investasi
x1: 0.2
x2: 0.2
x3: 0.2
x4: 0.2
x5: 0.2
```

**Gambar 14.** Bobot Investasi

**Return dan Risiko Portofolio Optimal Metode *Mean Absolute Deviation* (MAD)**

*Return* portofolio dihitung menggunakan *Python* dengan menggunakan *"def"* untuk mendefinisikan *expected return* dan bobot investas. Kemudian dengan menggunakan fungsi *"np.dot"* dilakukan perkalian antara *expected return* dan bobot. Digunakan *"np.array"* untuk mengubah nilai *expected return* dan bobot menjadi *array NumPy*, agar dapat menghitung *return* dari *expected return* dan bobot. Diperoleh *return* sebesar 0.000273%.

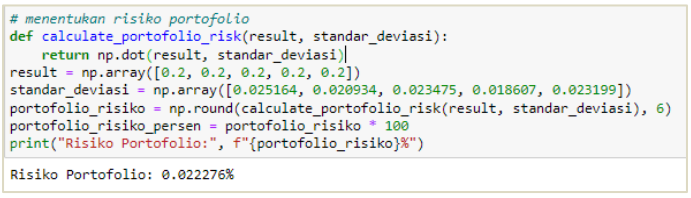


```
# menentukan return portofolio
def calculate_portofolio_return(returns_expected, result):
    return np.dot(returns_expected, result)
returns_expected = np.array([0.000548, 0.000160, 0.000019, 0.000023, 0.000616])
result = np.array([0.2, 0.2, 0.2, 0.2, 0.2])
portofolio_returns = np.round(calculate_portofolio_return(returns_expected, result), 6)
portofolio_returns_persen = portofolio_returns * 100
print("Return Portofolio:", f"{portofolio_returns}%")
```

```
Return Portofolio: 0.000273%
```

**Gambar 15.** *Return* dan Risiko Portofolio Optimal MAD

Risiko portofolio dihitung dengan menggunakan *Python* sama dengan cara mencari *return*, tetapi untuk mencari risiko portofolio dari nilai bobot dan standar deviasi. Diperoleh risiko sebesar 0.022276%.



```
# menentukan risiko portofolio
def calculate_portofolio_risk(result, standar_deviasi):
    return np.dot(result, standar_deviasi)
result = np.array([0.2, 0.2, 0.2, 0.2, 0.2])
standar_deviasi = np.array([0.025164, 0.020934, 0.023475, 0.018607, 0.023199])
portofolio_risiko = np.round(calculate_portofolio_risk(result, standar_deviasi), 6)
portofolio_risiko_persen = portofolio_risiko * 100
print("Risiko Portofolio:", f"{portofolio_risiko}%")
```

```
Risiko Portofolio: 0.022276%
```

**Gambar 16.** Risiko Portofolio

**Analisis Hasil Perhitungan Menggunakan *Python***

Berdasarkan perhitungan menggunakan metode *Mean Semivariance* dengan bantuan *Python* diperoleh hasil *return* sebesar 0.0035% dan risiko sebesar 0.080518%. Sedangkan dengan menggunakan metode *Mean Absolute Deviation* diperoleh hasil *return* sebesar 0.000273% dan risiko sebesar 0.022276%. Portofolio dengan metode Mean Semivariance memberikan return yang lebih tinggi jika dibandingkan dengan metode *Mean Absolute Deviation*. Untuk risiko yang diperoleh, metode Mean Semivariance memberikan risiko lebih tinggi dibandingkan metode *Mean Absolute Deviation*. Sesuai dengan hubungan antara return dan risiko, bahwa semakin tinggi return yang diperoleh maka semakin tinggi juga risiko yang diperoleh. Investor dengan tipe konservatif dianjurkan memilih metode *Mean Absolute Deviation*, karena menghasilkan risiko yang lebih kecil. Sedangkan investor dengan tipe agresif dianjurkan memilih metode *Mean Semivariance*.

## D. Kesimpulan

Berdasarkan hasil pengolahan data yang telah dilakukan, maka penelitia ini dapat disimpulkan yaitu pembentukan pirtofolio optimal dengan menggunakan metode metode *Mean Semivariance* menghasilkan *return* yang lebih besar dibandingkan dengan metode *Mean Absolute Deviation*, dan metode *Mean Absolute Deviation* menghasilkan risiko lebih kecil dibandingkan metode metode *Mean Semivariance*.

## Daftar Pustaka

[1] S. Zein and G. Gunawan, "Prediksi Hasil FIFA World Cup Qatar 2022 Menggunakan Machine Learning dengan Python," *Jurnal Riset Matematika*, pp. 153–162, Dec. 2022, doi: 10.29313/jrm.v2i2.1382.

[2] W. Hidayat, *Konsep Dasar Investasi Dan Pasar Modal*. Ponorogo: Uwais Inspirasi Indonesia, 2019.

[3] M. A. Alfiansyah and E. Kurniati, "Analisis Portofolio Saham Syariah di Masa Pandemi Covid-19 dengan Mengunakan Multi Indeks Model," *Jurnal Riset Matematika*, pp. 30–36, Jul. 2022, doi: 10.29313/jrm.v2i1.795.

[4] N. F. Nuzula dan F. Nurlaily, Dasar-Dasar Manajemen Investasi, Malang: UB Press, 2020.

[5] E. N. Vanti dan E. D. Supandi, "Pembentukan Portofolio Optimal Dengan Menggunakan Mean Absolurte Deviation dan Conditional Mean Variance," *Jurnal Fourier*, vol. 9, no. 1, pp. 25-34, 2020.

[6] L. A. Pratama, "Analisis Pembentukan Portofolio Saham Optimal Menggunakan Metode Single Index Model (Studi Empiris pada Saham Indeks LQ 45 di Bursa Efek Indonesia)," *Jurnal Ilmu Manajemen*, vol. 16, no. 1, pp. 48-60, 2019.

[7] I. F. Prasetyo dan A. G. Suarjaya, "Pembentukan Portofolio Optimal Dengan Menggunakan Model Indeks Tunggal," *E-Jurnal Manajemen*, vol. 9, no. 2, pp. 553-575, 2020.

[8] N. S. Suyasa, K. Dharmawan dan K. Sari, "Perhitungan Portofolio Optimal Dengan Metode Mean-Semivarians dan Mean Absoute Deviation (Studi Kasus: Indeks Harga Saham LQ45 Periode Februari 2017-Juli 2019)," *E-Jurnal Matematika*, vol. 10, no. 2, pp. 65-69, 2021.

[9] J. Estrada, "*Mean-Semivariance* Optimization: A Heuristic Approach," *Journal Of Applied Finance-Spring/Summer* , pp. 57-72, 2008.

[10] D. Wulandari, D. Ispriyanti dan A. Hoyyi, "Optimalisasi Portofolio Saham Menggunakan Metode *Mean Absolute Deviation* Dan Single Indeks Model Pada Saham Indeks LQ-45," *Jurnal Gaussian*, vol. 7, no. 2, pp. 119-131, 2018.

[11] J. Hartono , Teori Portofolio dan Analisis Investasi Edisi Ketujuh, Yogyakarta: BPFE, 2010.