Chapter_1

March 2, 2025

1 Chapter 1

[1]:
```python
import requests_cache
import yfinance as yf
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy as sp
from statsmodels.tsa.stattools import acf
```

[2]:
```python
session = requests_cache.CachedSession('yfinance.cache')
session.headers['User-Leonardo'] = 'Chapter_1'
```

[3]:
```python
data = yf.download('^STOXX50E')
```

```
YF.download() has changed argument auto_adjust default to True
```

```
[*********************100%***********************]  1 of 1 completed
```

[4]:
```python
prices = data.Close['^STOXX50E']
prices
```

[4]:
```
Date
2007-03-30    4181.029785
2007-04-02    4189.549805
2007-04-03    4246.299805
2007-04-04    4261.830078
2007-04-05    4271.540039
                  …
2025-02-24    5453.759766
2025-02-25    5447.899902
2025-02-26    5527.990234
2025-02-27    5472.560059
2025-02-28    5463.540039
Name: ^STOXX50E, Length: 4493, dtype: float64
```

[5]:
```python
prices.describe()
```

[5]:
```
count    4493.000000
mean     3387.685643
std       709.582645
min      1809.979980
25%      2875.939941
50%      3324.860107
75%      3781.790039
max      5533.839844
Name: ^STOXX50E, dtype: float64
```

[6]:
```python
prices = prices.to_numpy()
```

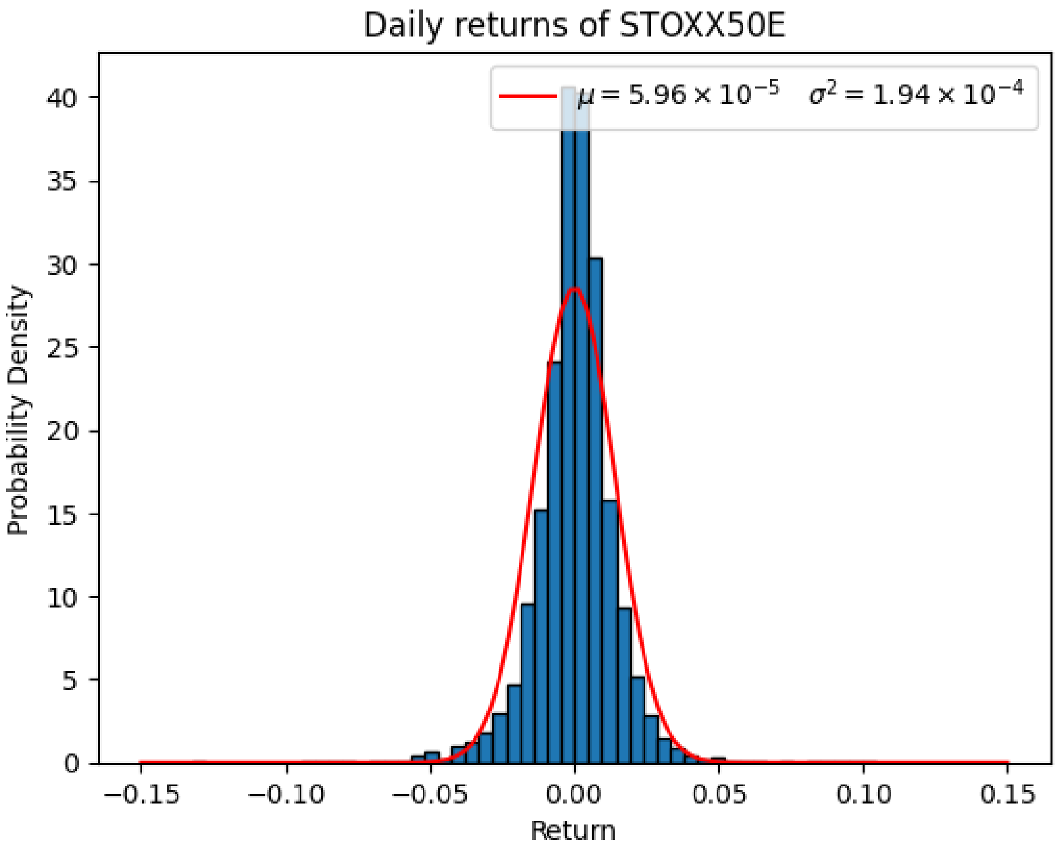
[7]:
```python
dates = data.Close['^STOXX50E'].index
```

1.1 Returns

[8]:
```python
returns = np.log(prices[1:])- np.log(prices[:-1])
mean = returns.mean()
variance = returns.var()
print('average = ',mean, '\nvariance = ', variance)
```

```
average =  5.955907439569843e-05
variance =  0.00019448892995148706
```

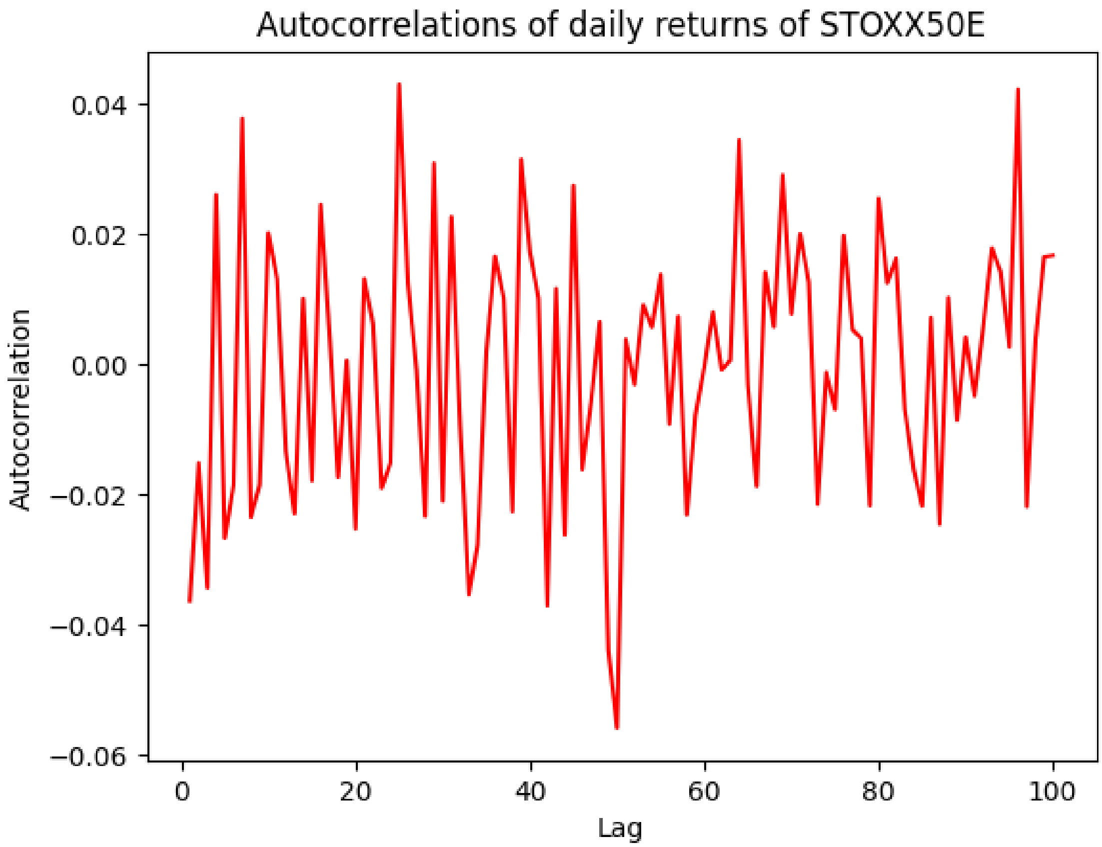
[9]:
```python
# Generate a range of x values for the normal distribution
x = np.linspace(-0.15, 0.15, 100)
# Evaluate the normal distribution at the x values
y = sp.stats.norm(loc =mean, scale = np.sqrt(variance)).pdf(x)

# Plot the histogram
plt.hist(returns, bins=50, edgecolor='black', density=True)
# Plot the fitted distribution
plt.plot(x, y, 'r-', label=r'$\mu = 5.96\times 10^{-5} \quad \sigma^2 = 1.
 ↪94\times 10^{-4}$')
plt.xlabel('Return')
plt.ylabel('Probability Density')
plt.title('Daily returns of STOXX50E')
plt.legend()
plt.show()
```



1.2 Autocorrelations

[10]:
```
max_lag = 100
# Calculate autocorrelation for lags 1 to 100
autocorr_values = acf(returns, nlags=max_lag)
```

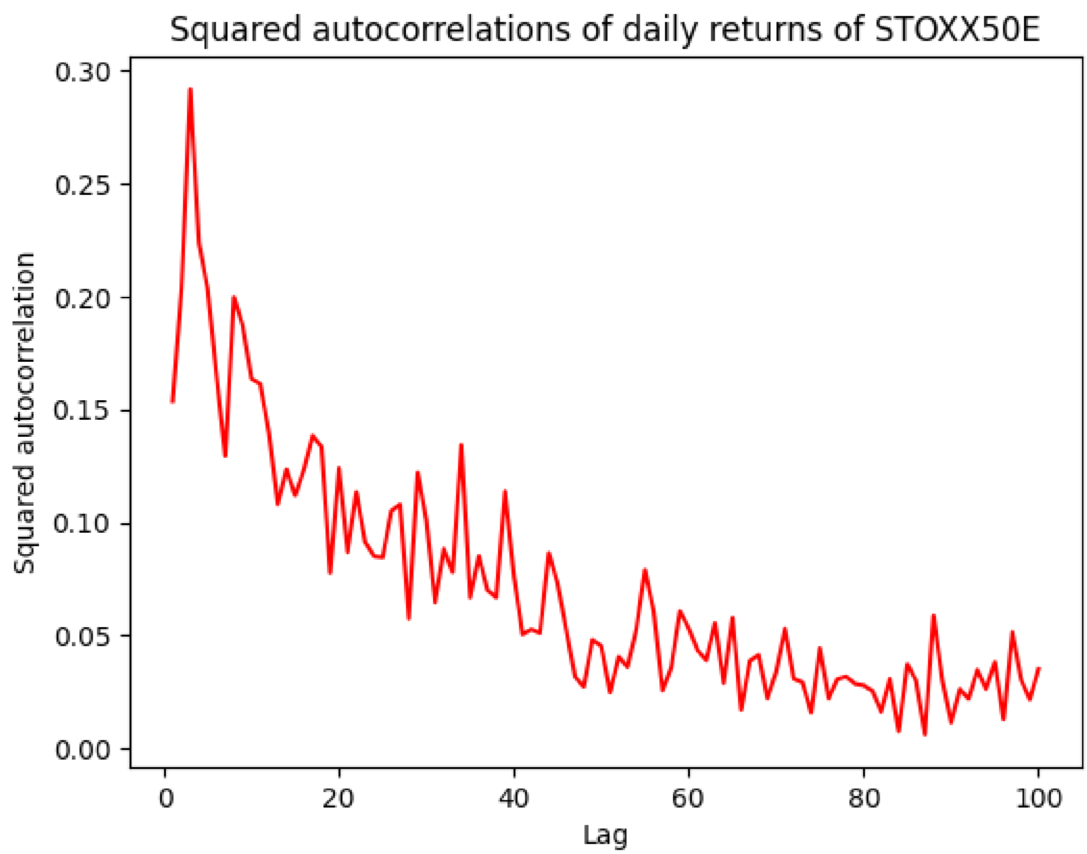
[11]:
```
x = np.arange(max_lag+1)

plt.plot(x[1:], autocorr_values[1:], 'r-')
plt.xlabel('Lag')
plt.ylabel('Autocorrelation')
plt.title('Autocorrelations of daily returns of STOXX50E')
plt.show()
```



1.3 Squared Autocorrelations

[12]:
```
# Calculate autocorrelation for lags 1 to 100
sq_autocorr_values = acf(returns**2, nlags=max_lag)
```

[13]:
```
plt.plot(x[1:], sq_autocorr_values[1:], 'r-')
plt.xlabel('Lag')
plt.ylabel('Squared autocorrelation')
plt.title('Squared autocorrelations of daily returns of STOXX50E')
plt.show()
```

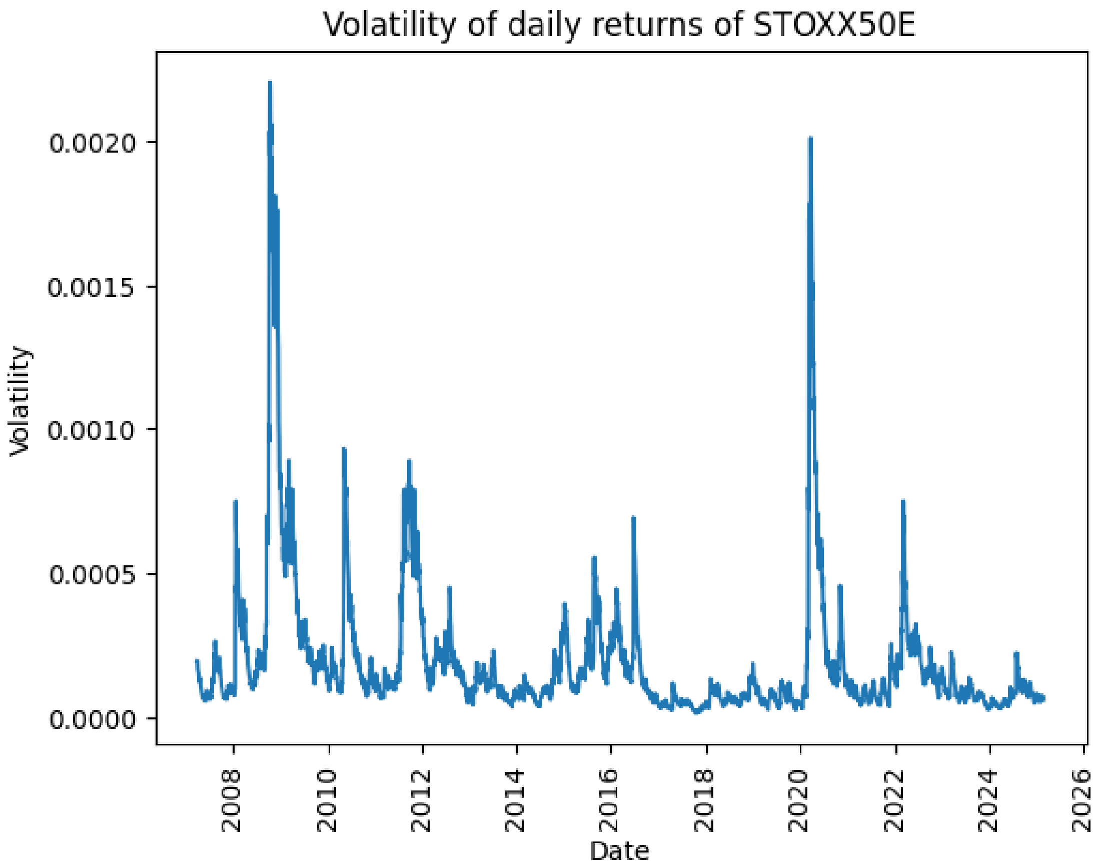


1.4 Riskmetrics

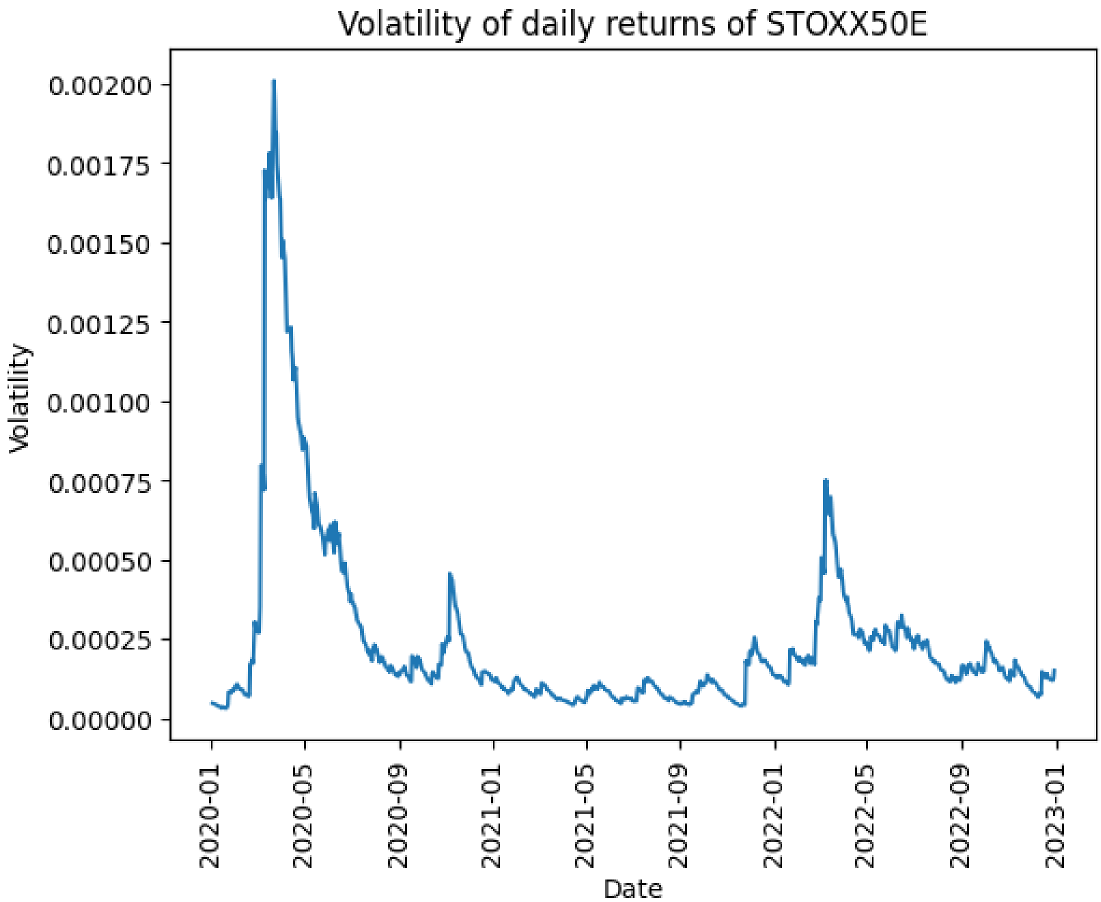
[14]:
```
def risk_metrics(returns, persistence):
    t_max = len(returns)
    volatility = np.zeros(t_max)
    volatility[0] = returns.var()
    for t in range(1, t_max):
        volatility[t] = persistence*volatility[t-1] +␣
 ↪(1-persistence)*returns[t]**2
    return volatility
```

[15]:
```
volatility = risk_metrics(returns, 0.94)
```

[16]:
```
plt.plot(dates[:-1],volatility)
# rotate x-tick-labels by 90°
plt.tick_params(axis='x',rotation=90)
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.title('Volatility of daily returns of STOXX50E')
plt.show()
```



[17]:
```
start_date = np.datetime64('2020-01-01')
end_date = np.datetime64('2022-12-31')

mask = (dates[:-1] >= start_date) & (dates[:-1] <= end_date)

plt.plot(dates[:-1][mask], volatility[mask])
# rotate x-tick-labels by 90°
plt.tick_params(axis='x', rotation=90)
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.title('Volatility of daily returns of STOXX50E')
plt.show()
```



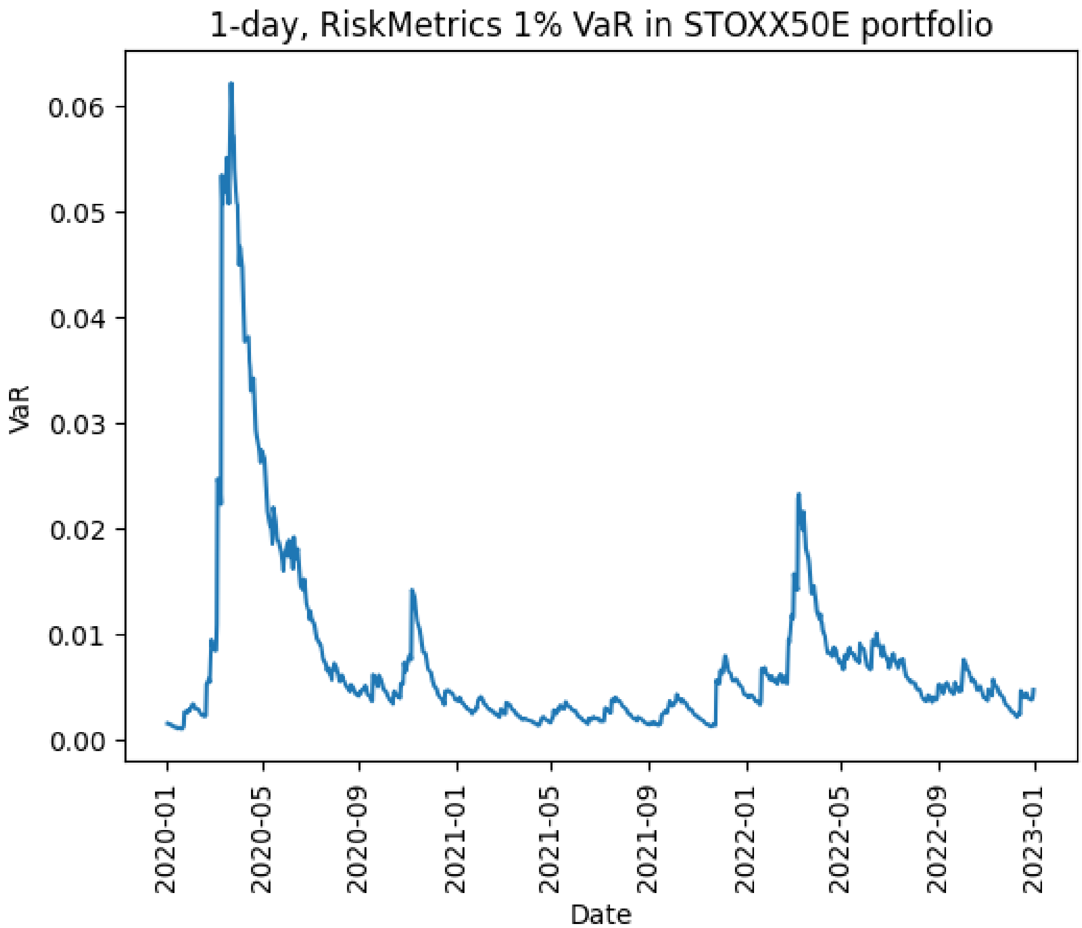
1.5 VaR

Assuming gaussian distribution of portfolio returns

[18]:
```
p = 0.01
quantile = sp.stats.norm.ppf(p, loc=mean, scale=np.sqrt(variance))
print(-quantile)
```

```
0.032383522729066964
```

[19]:
```
plt.plot(dates[:-1][mask], -quantile ** (-1) * volatility[mask])
# rotate x-tick-labels by 90°
plt.tick_params(axis='x', rotation=90)
plt.xlabel('Date')
plt.ylabel('VaR')
plt.title('1-day, RiskMetrics 1% VaR in STOXX50E portfolio')
plt.show()
```


1-day, RiskMetrics 1% VaR in STOXX50E portfolio
VaR
0.06
0.05
0.04
0.03
0.02
0.01
0.00
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01
2022-05
2022-09
2023-01
Date

[ ]: