MTH 500 Assignment 1

Mustafif Khan | 501095413 | Section 11

Mustafif Khan is solely responsible for its content.

1 Introduction

In this comprehensive report, I will expound on my solutions to the seven challenging questions in our Mathematical Statistics assignment. These questions delve into a broad range of topics we have studied in our course, including but not limited to random variables, probability distributions, time series, Markov chains, and Poisson processes.

To tackle each question, I employed different approaches and utilized various tools such as MATLAB, R, Python, or Excel, performing calculations, simulations, estimations, comparisons, and interpretations as deemed necessary.

I have included detailed supporting materials such as graphs, tables, formulas, and explanations whenever required to ensure that my solutions are easy to grasp. These materials serve to exemplify the steps taken in solving the problems and the reasoning behind my solutions.

2 Objectives

The main goals of this report are threefold:

2.1 Demonstrate Understanding

The first objective is to demonstrate my understanding of the concepts and methods we've learned in our Mathematical Statistics course. I aim to show how I've applied these concepts and methods to solve complex problems by presenting my solutions to the assignment questions.

2.2 Showcase Problem-Solving Skills

The second objective is to showcase my problem-solving skills. This involves using different tools and techniques to solve problems involving random variables, probability distributions, time series, Markov chains, and Poisson processes. By detailing the steps I took in solving each problem, I aim to show how I approached each problem and how I used the tools at my disposal.

2.3 Communicate Effectively

The third objective is communicating my results and reasoning effectively. This involves using graphs, tables, formulas, and explanations to support my answers. I aim to make my solutions clear and easily understood by presenting these supporting materials.

Question 1

$$X_t = \sin(\omega t + \Theta) + 3\varepsilon_t$$

a) $E[X_t] = E[\sin(\omega t+\Theta) + 3\varepsilon_t]$

$$= E[\sin(\omega t + \Theta)] + \underbrace{E[3\varepsilon_t]}_{0}$$

$$E[\sin(\omega t+\Theta)] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sin(\omega t+\Theta)\,d\Theta$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} \underbrace{\cos(\Theta)\sin(\omega t)}_{0} + \underbrace{\cos(\omega t)\sin(\Theta)}_{0}\,d\Theta$$

$$= 0$$

$$= 0 + 0$$

$\therefore m(t) = 0$

b) $\sigma^2(t) = E(X^2)$

$$= E[\sin^2(\omega t+\Theta) + 6\sin(\omega t+\Theta) + 9\varepsilon_t^2]$$

$$= E[\sin^2(\omega t+\Theta)] + \underbrace{E[6\sin(\omega t+\Theta)]}_{0} + E[9\varepsilon_t^2]$$

$\because E[\varepsilon_t^2] = 1$, then $E[9\varepsilon^2] = 9$

$$E[\sin^2(\omega t+\Theta)] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sin^2(\omega t+\Theta)\,d\Theta$$

$$= \frac{1}{4\pi}\int_{-\pi}^{\pi} 1 - \underbrace{\cos(2\omega t + 2\Theta)}_{0}\,d\Theta$$

$$= \frac{1}{4\pi}\left(\Theta\Big|_{-\pi}^{\pi}\right)$$

$$= \frac{1}{4\pi}\cdot 2\pi$$

$$= \frac{1}{2}$$

$$\sigma^2(t) = \frac{1}{2} + 0 + 9$$

$$= \frac{19}{2}$$

c) $\gamma(t,t+h) = E[(X_t - \mu_t)(X_{t+h} - \mu_{t+h})]$

$= E[(X_t)(X_{t+h})]$

$= E[(\sin(wt+\theta) + h\varepsilon)(\sin(wt+wh+\theta) + h\varepsilon)]$

$= E[\sin(wt+\theta)\sin(wt+wh+\theta) + h^2\varepsilon^2]$

$= \frac{1}{4\pi}\int_{-\pi}^{\pi} \cos(wh) - \cos(2wt+2\theta+wh)\, d\theta + h^2$

$= \frac{1}{4\pi}\cos(wh) \underbrace{\int_{-\pi}^{\pi} - \cos(2wt+2\theta+wh)\, d\theta}_{2\pi} + h^2$

$= \frac{1}{2}\cos(wh) + h^2$

$= \frac{1}{2}\cos(wh) + 9$

$$\rho(t,t+h) = \frac{\gamma(t,t+h)}{\sigma(t)\,\sigma(t+h)}$$

$$= \left(\frac{1}{2}\cos(wh) + 9\right) \cdot \frac{2}{19}$$

$$= \cos(wh) \cdot \frac{9}{19}$$

Q2

a) $F_{x,y}(0.5,-1)$

$$F_{XY}(0.5,-1) = \begin{cases} \frac{1}{8} & \text{for } x=-1 \text{ \& } y=-2 \\ 0 & \text{for } x=-1 \text{ \& } y=-1 \\ 0 & \text{for } x=0.5 \text{ \& } y=-2 \\ \frac{1}{4} & \text{for } x=0.5 \text{ \& } y=-1 \end{cases}$$

$= \frac{1}{8} + \frac{1}{4}$

$= \frac{3}{8}$

b) $P_X(-1) = \frac{1}{8} + 0 + 0$

$= \frac{1}{8}$

$P_X(0.5) = \frac{1}{4} + \frac{1}{2}$

$= \frac{3}{4}$

$P_X(1) = \frac{1}{8} + 0 + 0$

$= \frac{1}{8}$

c) $P_{Y/X}(-2/0.5) = \dfrac{P_{x,y}(0.5,-2)}{P_x(0.5)}$

$= 0 \cdot \frac{4}{3}$

$= 0$

$P_{Y/X}(-1/0.5) = \dfrac{P_{x,y}(0.5,-1)}{P_x(0.5)}$

$= \frac{1}{\cancel{4}} \cdot \frac{\cancel{4}\,1}{3}$

$= \frac{1}{3}$

$P_{Y/X}(1/0.5) = \dfrac{P_{x,y}(0.5,1)}{P_x(0.5)}$

$= \frac{1}{\cancel{2}} \cdot \frac{\cancel{4}\,2}{3}$

$= \frac{2}{3}$

d) $E[Y | X = 0.5] = (-2)(0) + (-1)(\frac{1}{3}) + (1)(\frac{2}{3})$

$= \frac{1}{3}$

$\therefore$ The average value of Y when fixed at $X = 0.5$ is $1/3$.

e)

$E(X) = \frac{3}{8}$ $\quad Var(X) = \frac{11}{64}$

$E(Y) = \frac{1}{8}$ $\quad Var(Y) = \frac{87}{64}$

$$E(XY) = (-1)(-2)(\tfrac{1}{8}) + \tfrac{1}{8} + (-1)(0.5)(\tfrac{1}{4}) + \tfrac{1}{4}$$

$$= \tfrac{1}{4} + \tfrac{1}{8} - \tfrac{1}{8} + \tfrac{1}{4}$$

$$= \tfrac{1}{2}$$

$$Cov(X,Y) = E(XY) - E(X)E(Y)$$

$$= \tfrac{1}{2} - (\tfrac{3}{8})(\tfrac{1}{8})$$

$$= \tfrac{32}{64} - \tfrac{3}{64}$$

$$= \tfrac{29}{64}$$

$$\rho_{X,Y} = \frac{Cov(X,Y)}{\sqrt{Var(X)\,Var(Y)}}$$

$$= \frac{\frac{29}{64}}{\sqrt{(\frac{11}{64})(\frac{87}{64})}}$$

$$= \frac{29}{\cancel{64}} \cdot \frac{\cancel{64}}{\sqrt{957}}$$

$$= \frac{29}{\sqrt{957}}$$

$$= \frac{29\sqrt{957}}{957}$$

Q3

$$f(x,y) = 4e^{-2y} \qquad 0 < 4x < y$$

$$f_X(x) = \int_{4x}^{\infty} 4e^{-2y}\, dy$$

$$= \left[-2e^{-2y}\right]_{4x}^{\infty}$$

$$= 0 - (-2e^{-24x}]$$

$$= 2e^{-24x}$$

$$f_Y(y) = \int_0^{y/4} 4e^{-2y}\, dx$$

$$= 4e^{-2y}\left[x\right]_0^{y/4}$$

$$= \frac{4}{4} y e^{-2y}$$

$$f_{Y/X}(y/x) = \frac{f(x,y)}{f_X(x)} = \frac{4e^{-2y}}{2e^{-24x}}$$

$$f_{Y/X}(x,y) = 2e^{24x-2y}$$

$$E[Y/X] = \int_{4x}^{\infty} y\, 2e^{24x-2y}\, dy$$

Use IBP with

$u = y \qquad du = dy$

$dv = 2e^{24x-2y} \qquad v = -e^{24x-2y}$

$$= uv - \int v\, du$$

$$= -ye^{24x-2y} + \int_{4x}^{\infty} e^{24x-2y}\, dy$$

$$= -ye^{24x-2y} - \frac{1}{2}\left[e^{24x-2y}\right]_{4x}^{\infty}$$

$$= -ye^{24x-2y} - \frac{1}{2}\left[0-(-1)\right]$$

$$= \frac{1}{2} - ye^{24x-2y}$$

Question 4

A)

To simulate a moving average process with $\theta = \frac{k}{10}$ driven by a Gaussian white noise, I created the R code `q4a.r` where defined our parameters, `n` to define the number of observations which I set to 1000, `theta` which is set to 0.3 since $k = 3$. Lastly as defined in the question, we set the variable `sigma` to 1. To generate the Gaussian white noise, I use the `rnorm` function which uses the normal distribution and is randomized to simulate the white noise. To intialize the moving average process, we use the `rep` function which replicates elements of a list, we set this from 0 to `n`. After we create a loop to define each `t` in the moving average process using the formula given. After we plot our function.

q4a.r

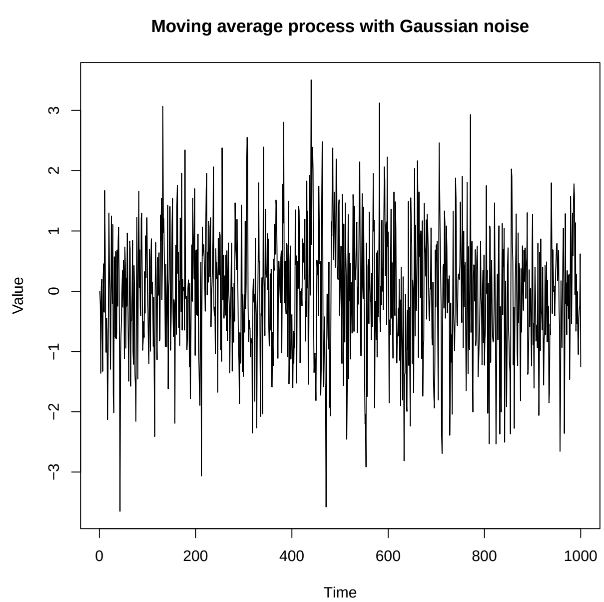
```
# Set the parameters
n <- 1000 # number of observations
theta <- 0.3 # value of theta (k/10) k = 3
sigma <- 1 # standard deviation of white noise

# Generate a Gaussian white noise
Z <- rnorm(n, mean = 0, sd = sigma)

# Initialize the moving average process
X <- rep(0, n)

# Define the moving average process
for (t in 2:n) {
    X[t] <- Z[t] + theta * Z[t - 1]
}

# Plot the simulated values
plot(X, type = "l",
main = "Moving average process with Gaussian noise",
xlab = "Time", ylab = "Value")
```



B)

To simulate the moving average process with t-distribution with 4 degrees of freedom, we will set similar parameters like A), with $n = 1000$, $theta = 0.3$, and $df = 4$, After to generate the randomized t-distribution, we will use the `rt` function with `n` and `df=df`. After we will intialize and define the moving average process like we had done before, and after we will plot our simulated values.

q4b.r

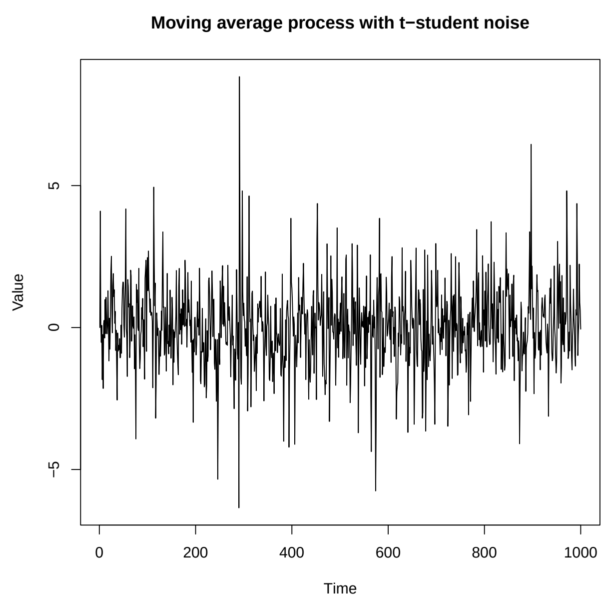
```
# Set the parameters
n <- 1000 # number of observations
theta <- 0.3 # value of theta
df <- 4 # degrees of freedom for t-student distribution

# Generate a t-student white noise
Z <- rt(n, df = df)

# Initialize the moving average process
X <- rep(0, n)

# Define the moving average process
for (t in 2:n) {
  X[t] <- Z[t] + theta * Z[t - 1]
}

# Plot the simulated values
plot(X,
  type = "l",
  main = "Moving average process with t-student noise",
  xlab = "Time", ylab = "Value"
)
```

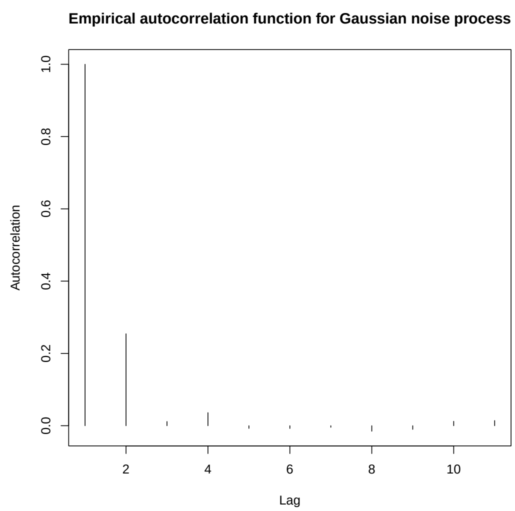


C)

q4c.r

```r
# Set the parameters
n <- 1000 # number of observations
theta <- 0.3 # value of theta
sigma <- 1 # standard deviation of white noise
# Generate a Gaussian white noise
Z <- rnorm(n, mean = 0, sd = sigma)
# Initialize the moving average process
X <- rep(0, n)
# Define the moving average process
for (t in 2:n) {
    X[t] <- Z[t] + theta * Z[t - 1]
}
# Define a function to calculate the empirical autocorrelation
autocorr <- function(x, h) {
    # x is the input series, h is the lag
    n <- length(x) # length of the series
    x_mean <- mean(x) # mean of the series
    num <- sum((x[(h + 1):n] - x_mean) * (x[1:(n - h)] - x_mean)) # numerator
    den <- sum((x - x_mean)^2) # denominator
    return(num / den)
}
# Set the maximum lag to plot
max_lag <- 10
# Plot the empirical autocorrelation
# function for Gaussian noise process
plot(sapply(0:max_lag, function(h) autocorr(X, h)), type = "h",
main = "Empirical autocorrelation function
for Gaussian noise process",
xlab = "Lag", ylab = "Autocorrelation")
```

The provided R code sets up a moving average process of order 1 with Gaussian noise, calculates the empirical autocorrelation of the process at different lags, and plots the autocorrelation function. The empirical autocorrelation function is calculated as the ratio of the covariance between the series at time t and $t + h$ and the variance of the series. The plot shows the autocorrelation values for each lag from 0 to 10. The term $\rho X(1)$ represents the autocorrelation of the process at lag 1, indicating how much the current value of the process depends on its previous value. For a moving average process of order 1.



Question 5

For question 5, I solved the problem by utilizing Python classes to handle finding each subsection of this question so I can present the final answers in a complete table. To answer each subsection I will provide the snippet of the python file `q5.py` that corresponds with finding the said problem. At the end once the code is shown, I will present the results in a neat table. The reason I chose to use a Python class is because all of the poisson files were identical, which means I would be able to handle the repetitive tasks effectively.

i)

First we need to be able to load the data and estimate our λ value. To begin in our code, we will be using the `pandas` library to read an excel file and use the second column which contains the arrivals. We will need to be able to calculate the poisson distribution, so we will need to make use of the `scipy` library, and lastly for better numerical capabilities we will be using the `numpy` library.

We will be creating a class called `PoissonData` and to create a new instance of it, we will define a constructor which will take in a file name, we will read it use the first column which will be assigned to the `arrivals` field, then we will take the mean of the arrivals and that will be our λ estimate and used later we will set μ to 30. This can all be seen below:

q5.py i)

```
import pandas as pd
from scipy.stats import poisson
import numpy as np

# A class to represent each of the Poisson Data in the files
class PoissonData:
    def __init__(self, filename):
        self.arrivals = pd.read_excel(filename,usecols=[1])
        self.lambdaHat = self.arrivals.mean().values[0]
        self.mu = 30
```

ii)

To calculate the probability of arrivals that occur in the interval of 5 hours, we will first calculate the rate which is found by $\lambda * \frac{5}{24}$ then to get the probability we use the Poisson PMF from 0 to our rate. This is all calculated in our function `no_arrivals_5_hours_prob` in the `PoissonData` class:

q5.py ii)

```
    # Q5 ii)
    def no_arrivals_5_hours_prob(self):
        lambda_5_hours = self.lambdaHat * (5 / 24)
        return poisson.pmf(0, lambda_5_hours)
```

iii)

To calculate the probability of at least one arrival occurs, we will be needing to find $1 - Poisson(0, \lambda)$ as shown in the function `at_least_one_arrival_prob` in the `PoissonData` class:

q5.py iii)
```
    # Q5 iii)
    def at_least_one_arrival_prob(self):
        prob = 1 - poisson.pmf(0, self.lambdaHat)
        return prob
```

iv)

To calculate the mean and variance of the arrival in a day, we need to remember that in a Poisson distribution, $E(X) = \lambda$ and $Var(X) = \lambda$, so in our `mean_arrivals` and `variance_arrivals` functions, we return our `lambdaHat` field.

q5.py
```
    # Q5 iv)
    def mean_arrivals(self):
        return self.lambdaHat
    # Q5 iv)
    def variance_arrivals(self):
        return self.lambdaHat
```

v)

To calculate the average and standard deviation of the total time it takes for a patient in a single day using $\mu = 30$ is by multiplying `self.mu` by the mean of arrivals and standard deviation respectively. This can be seen below:

q5.py
```
    # Q5 v)
    def average_total_time(self):
        return self.arrivals.mean() * self.mu
    # Q5 v)
    def std_dev_total_time(self):
        return np.sqrt(self.arrivals.var()) * self.mu
```

Results

To get the results for each file, I did add an extra method to turn our results into strings where each result from i)...v) would be rounded to 4 decimal points. I created a `results.txt` file where each of the results were appended and in a `for` loop all files from $0 \ldots 9$ were calculated.

```
q5.py
    def to_string(self):
        return f"""
File: {filename}
Estimated lambda: {self.lambdaHat:.4f}
Probability no arrivals in 5 hours:
{self.no_arrivals_5_hours_prob():.4f}
Probability at least one arrival in a day:
{self.at_least_one_arrival_prob():.4f}
Mean arrivals per day: {self.mean_arrivals():.4f}
Variance of arrivals per day: {self.variance_arrivals():.4f}
Average total time patients assisted per day:
{self.average_total_time().values[0]:.4f} minutes
Standard deviation of total time patients assisted per day:
{self.std_dev_total_time().values[0]:.4f} minutes
"""
# will store all the results
result_file = open('results.txt', 'a')
# Each poisson data will be read from the directory `poissn`
# and get the data from the `poissn.xlsx` file
for i in range(10):
    result_file.write('---------------------------')
    filename = f'poissn/poissn{i}.xlsx'
    pdata = PoissonData(filename)
    string = pdata.to_string()
    result_file.write(string)
    result_file.write('---------------------------')

result_file.close()
```

Instead of showing a 100 line text file, I opted to neatly format it into a table where we get our results for each file:

File	λ	Prob. No 5 Hrs	Prob. At Least One	Mean	Variance	Avg. Time (mins)	Std. Dev. (mins)
poissn0	3.0606	0.5285	0.9531	3.0606	3.0606	91.8182	58.4208
poissn1	2.9596	0.5398	0.9482	2.9596	2.9596	88.7879	49.5958
poissn2	2.9293	0.5432	0.9466	2.9293	2.9293	87.8788	42.4810
poissn3	2.9394	0.5421	0.9471	2.9394	2.9394	88.1818	54.1797
poissn4	3.1111	0.5230	0.9554	3.1111	3.1111	93.3333	47.5094
poissn5	3.2121	0.5121	0.9597	3.2121	3.2121	96.3636	54.5941
poissn6	3.1515	0.5186	0.9572	3.1515	3.1515	94.5455	51.8489
poissn7	3.0000	0.5353	0.9502	3.0000	3.0000	90.0000	50.1630
poissn8	3.3131	0.5015	0.9636	3.3131	3.3131	99.3939	52.0757
poissn9	3.2121	0.5121	0.9597	3.2121	3.2121	96.3636	51.1192

Table 1: Poisson Data Results

Question 6

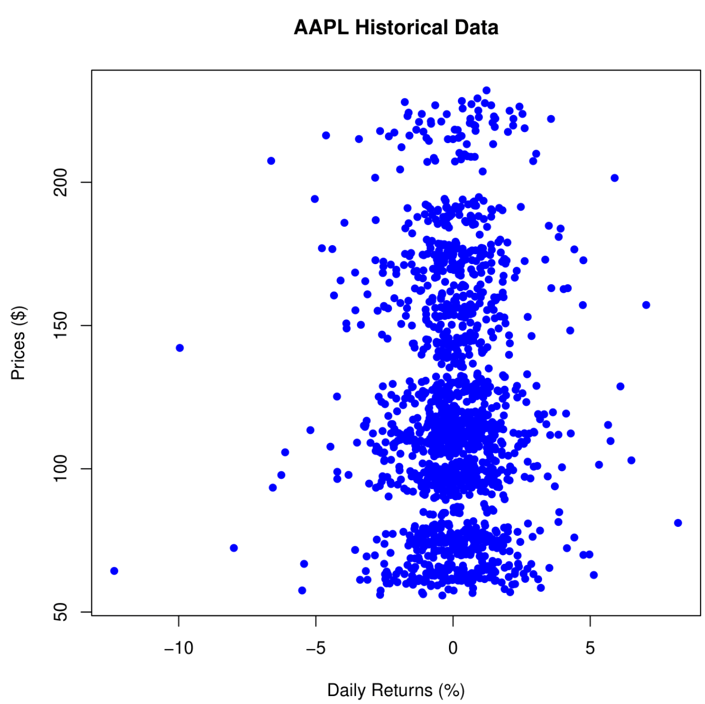
The timeseries data we will be analyzing is the Apple stock price found in `AAPL.csv`, to solve each part we will be using R in the file `q6.r`.

i)

To plot the Apple time series we will need to read the `csv` file, get the daily returns from the `Change` column, then we need to convert each of the strings into numerical numbers, we also get the stock prices from the Price column. With the daily_returns and prices variables, we will create a dataframe with the daily returns as the x-axis and prices as the y-axis. Lastly we plot the prices as a scatterplot:

q6.r i)

```r
# i)
########################################################################
# Install the moments package if not already installed
if (!requireNamespace("moments", quietly = TRUE)) {
  install.packages("moments")
}
# Load the moments package
library(moments)
# Read the CSV file
data <- read.csv("AAPL.csv")
# Extract daily returns and prices
dailyReturnStr <- data$Change
dailyReturns <- as.numeric(sub("%", "", dailyReturnStr))
prices <- data$Price
# Create a data frame
df <- data.frame(DailyReturns = dailyReturns, Prices = prices)
# Create a boxplot
plot(df,
  main = "AAPL Historical Data",
  xlab = "Daily Returns (%)",
  ylab = "Prices ($)",
  pch = 16, # Set the point character (16 for solid circles)
  col = "blue" # Set point color
)
```



ii)

To calculate the first four moments (mean, standard deviation, skewness and kurtosis) for the prices, and daily returns, we will need to use the functions `mean`, `sd`, `skewness` and `kurtosis` respectively for the variables `prices` and `dailyReturns` that were calculated previously.

q6.r ii)

```r
# ii)
########################################################################
# First four moments for prices series
prices_mean <- mean(prices)
cat("Mean of prices:", prices_mean, "\n")
prices_std <- sd(prices)
cat("Standard deviation of prices:", prices_std, "\n")
prices_skew <- skewness(prices)
cat("Skewness of prices:", prices_skew, "\n")
prices_kurt <- kurtosis(prices)
cat("Kurtosis of prices:", prices_kurt, "\n")
cat("\n")
# First four moments for daily returns
dr_mean <- mean(dailyReturns)
cat("Mean of daily returns:", dr_mean, "\n")
dr_std <- sd(dailyReturns)
cat("Standard deviation of daily returns:", dr_std, "\n")
dr_skew <- skewness(dailyReturns)
cat("Skewness of daily returns:", dr_skew, "\n")
dr_kurt <- kurtosis(dailyReturns)
cat("Kurtosis of daily returns:", dr_kurt, "\n")
```

When we run our script we get the following result:

```
Mean of prices: 120.4879
Standard deviation of prices: 41.64585
Skewness of prices: 0.5805278
Kurtosis of prices: 2.611525

Mean of daily returns: 0.06097975
Standard deviation of daily returns: 1.58414
Skewness of daily returns: -0.4705426
Kurtosis of daily returns: 8.834479
```