Actsc 445 Final Project

Prediction of Tesla Stock Price and Estimation of Risk Measures

Zheng, Xinran (Student ID: 20819287)
Zhai, Minghao (Student ID: 20825248)

Fall 2022

1 Introduction

According to chapter 1, one type of risks in QRM is market risk which refers to the risk of loss in a financial position due to changes in the underlying component like stock price. As two students majoring in Mathematical Finance, we are interested in stock price analysis. Among all the stocks in the financial market, we believe that stocks and shares related to new energy automobile are the most superior ones. Thus, this report will focus on Tesla's stock price. We will use historical Tesla's stock price data to predict future prices and then we will further analyze the maximum losses by creating a portfolio consisting of 10 shares of Tesla and 1 share of gold to estimate the value at risk(VaR) and expected shortfall(ES) at $\alpha = 0.99$ level. The data of Tesla's stock price and gold price come from Yahoo Finance[1]. We extract daily stock prices from 2013-01-02 to 2022-11-07.

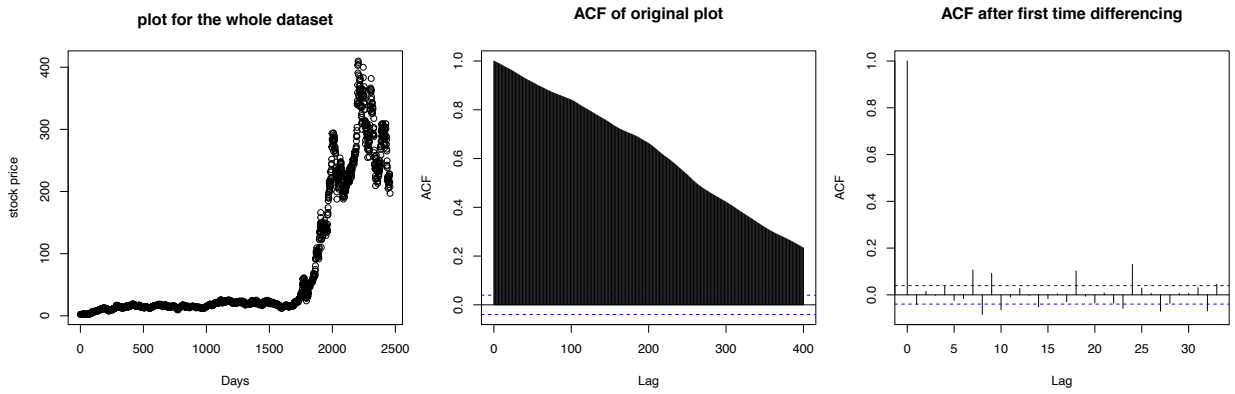
Thus, this project is a statistical analysis containing two main sections. The first section demonstrates three methods of Tesla's stock price prediction under different situations. These three prediction methods are: polynomial regression, Holtwinters prediction and Box-jenkins method. In this part, knowledge of time series will be involved and detailed explanation will also be demonstrated in each graph and plot below. The second section of this report is the real life application of two risk measures, value at risk(VaR) and expected shortfall(ES) on a give portfolio consisting of 10 shares of Tesla and 1 share of Gold. In this section, four methods are used to estimate the value at risk(VaR) and expected shortfall(ES) of this portfolio at $\alpha = 0.99$ level. These four methods include variance-covariance method, historical simulation, Monte Carlo (normal and t) and peaks over threshold(POT). We will analyze the performance of each method in this section and obtain the maximum losses.

[1]The data comes from Yahoo Finance(https://finance.yahoo.com/quote/TSLA/ & https://finance.yahoo.com/quote/GC=F/)

2 Section I: Prediction of Future Stock Price of Tesla

The concepts and R code of this section comes from the book Time Series Analysis and Its Applications with R Example.

We will first use Tesla stock price data from Yahoo Finance to do the prediction. This dataset includes stock price of 2458 working-days. To begin with, the following graphs provide some basic analysis of this stock.

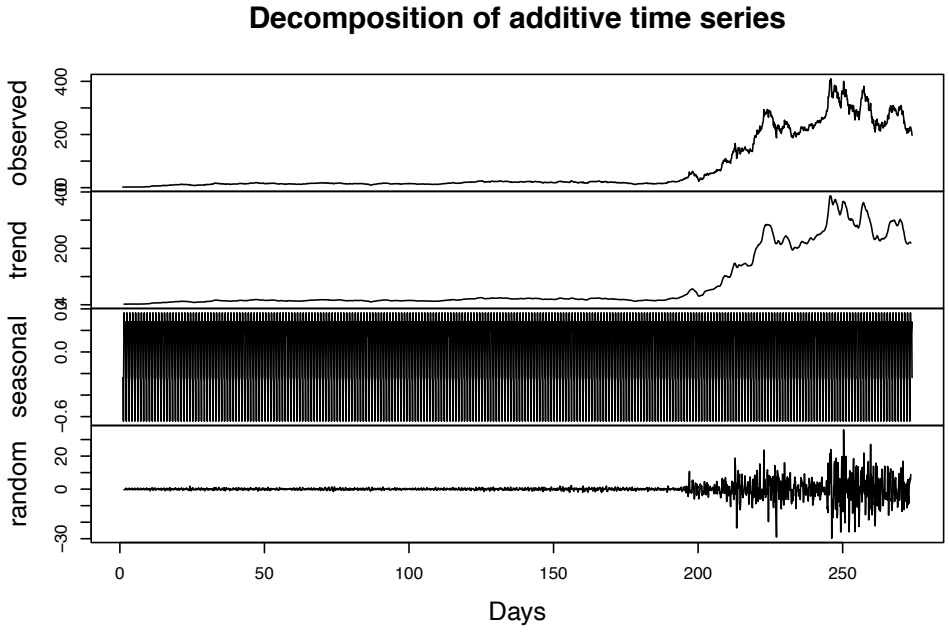


The first left graph is a scatter plot of days vs. Tesla's stock prices.

From the ACF graph(the middle), we can clearly notice that there is a slow decay, which implies there is a trend.

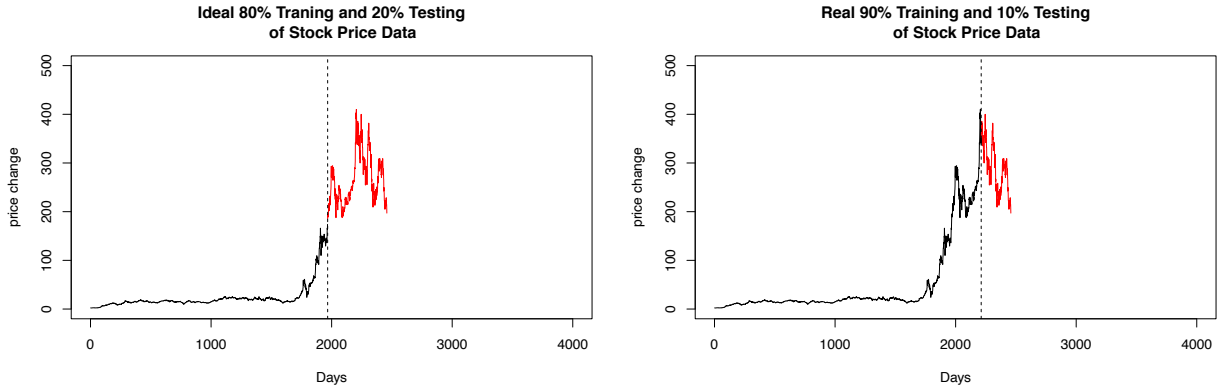
By observation from the right graph, we decided to set our seasonality to be 9 since lag 9,18,27 are in general significant, which indicates that these data points have correlation.

The graph below decomposes the time series based on its observation, trend, seasonality and randomness.



Based on this decomposition graph, we are able to observe that the trend is significant as we mentioned above. However from the scale's perspective, seasonal component is less significant because its range is from -0.6 to 0.2. Therefore, we can ignore seasonality when we conduct the prediction in the future.

For the purpose of prediction's accuracy and effectiveness, ideally, we would use 80% of data to be the training set shown as black part, and the red part is our testing set in the left plot below. However, we find that the volatility is large, therefore, we decide to use 90% of data to be our new training set and the rest of 10% to be our testing set as shown in the right plot below.



2.1 Method 1: Polynomial Regression

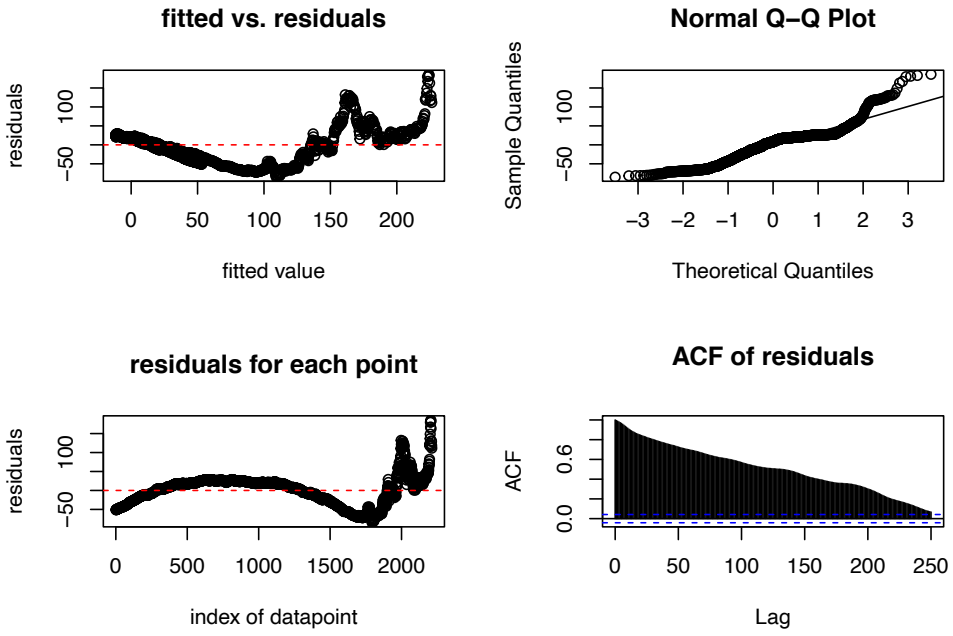
We then start to do the prediction by using our first method which is polynomial regression. To determine the polynomial's degree, we need to calculate the mean square error(MSE) of each poloynomial with different degrees.

Table 1: Summary of MSE

	MSE	Regression with seasonality's MSE
Polynomial with degree 1	23556.346	23556.642
Polynomial with degree 2	4945.286	4944.907
Polynomial with degree 3	24646.043	24646.643
Polynomial with degree 4	57681.985	57685.380
Polynomial with degree 5	50060.571	50064.580

From above MSE table, polynomial regression with degree 2 has the smallest MSE. We can then choose polynomial regression with degree 2 and combine with seasonality to do the prediction. However, we previously made an assumption that seasonality component is not significant. Also, by observation from the table, it can be seen that there is no large difference between a polynomial regression model with degree of 2 with seasonality and that model without seasonality since the difference between MSE without seasonality and MSE with seasonality is not large. Therefore, we choose the degree 2 polynomial without seasonality to be our final polynomial regression model.

After finding the appropriate degree, we can do the linear regression.



For the first plot, we can see that the line rarely fluctuated around 0. The model assumption is that fitted value and residuals are uncorrelated and we would expect that the graph has no clear trend. However in this plot, we can clearly observe that there exists a trend. Also, mean and variance are not constant. Thus, it seems that our assumption does not apply for this model.

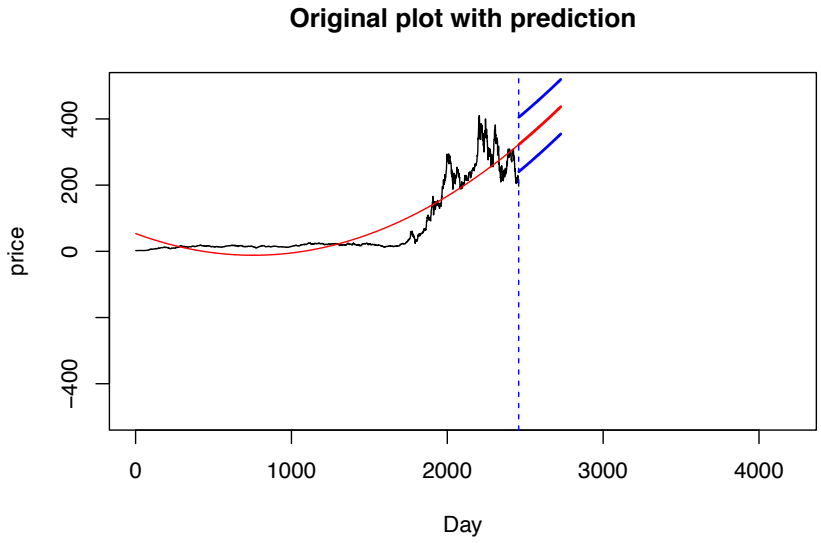
For the second plot, it is almost a straight line except the top right part. Thus, it seems that the normality is not satisfied.

For the third plot, it shows that the variance is not constant, which means that this model does not fit the data very well.

For the fourth plot, there is a decay, which means the trend exists.

Hence, we conclude that this linear regression may not be the best model.

Based on the regression, we predict the 270 working days of future stock price.



In the above plot, the red line is our prediction and the blue line represents the prediction interval. As the graph shown above, we have 95% of confidence to say that the real price in the future will be located within this interval. Based on above analysis, though polynomial regression with degree 2 is superior in MSE to polynomials with other degrees, it performs badly in the goodness of fit. Thus, it can be concluded that polynomial regression may not be the best model to fit the data.

2.2 Method 2: Holtwinter and Smoothing Methods

We want to transform the non-stationary data into stationary data. In reality, the raw dataset is usually not stationary, which means the mean and variance are not constant and each data point may have some correlation. Therefore, in general, we need to compute

X_t - X_{t-1} to eliminate the trend. Next, we will do this subtraction with periodic lag to eliminate the seasonality and then we can model the rest of the part. However, when we conduct elimination, we might change the dataset, which may result in the loss of information from the raw data. Therefore, Holtwinter method can be used here to avoid this situation. Hence, we can directly apply Holtwinter method to do the prediction.

We will be using simple exponential smoothing, double exponential smoothing, additive smoothing, and multiplicative smoothing. These models do not have much difference but only differ in some parameters.

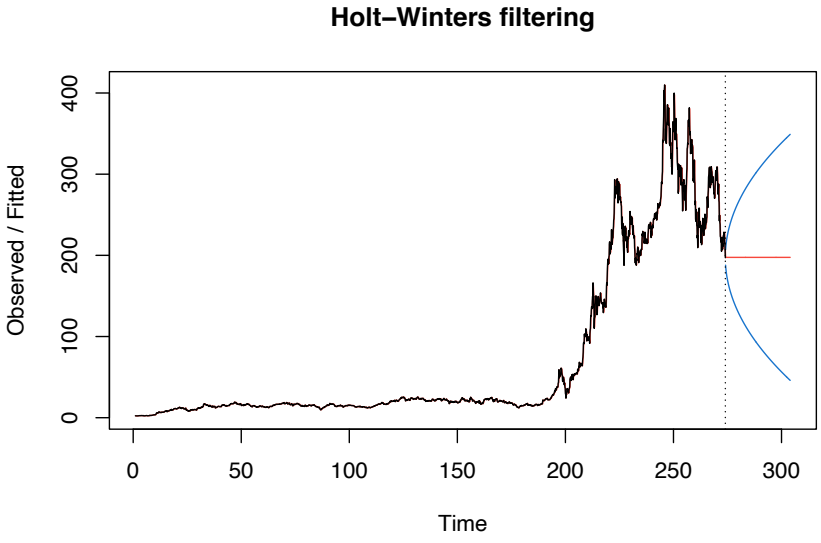
Similarly, we will compute MSE for each model to select the best one.

Table 2: MSE of Holtwinter and Smoothing Methods

Model	MSE
Simple Exponential Smoothing	4684.215
Double Exponential Smoothing	20895.700
Additive-HoltWinters	20880.380
Multiplicative-HoltWinters	21186.490

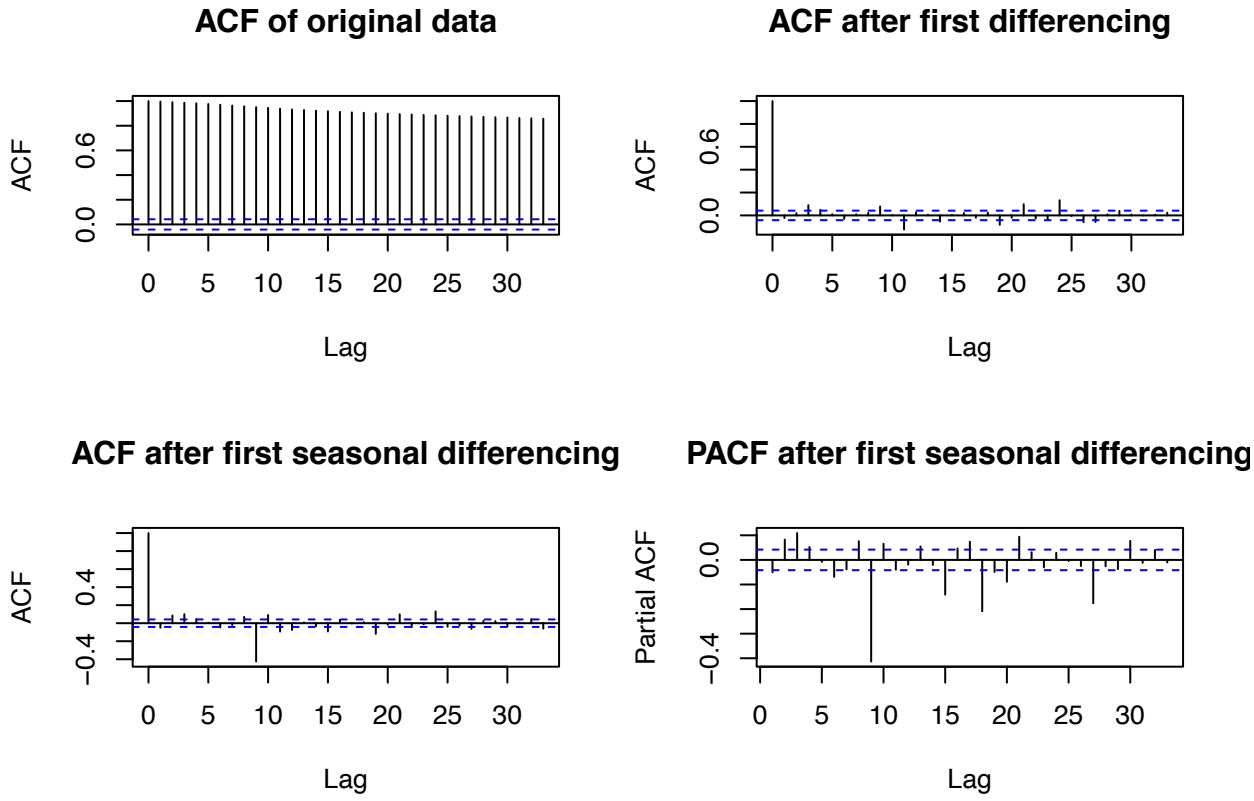
From the table above, simple exponential smoothing has the smallest MSE of 4684.215, so we choose simple exponential smoothing for prediction.

Then we fit a new model based on the new data, and predict the 270 working days of future stock price.



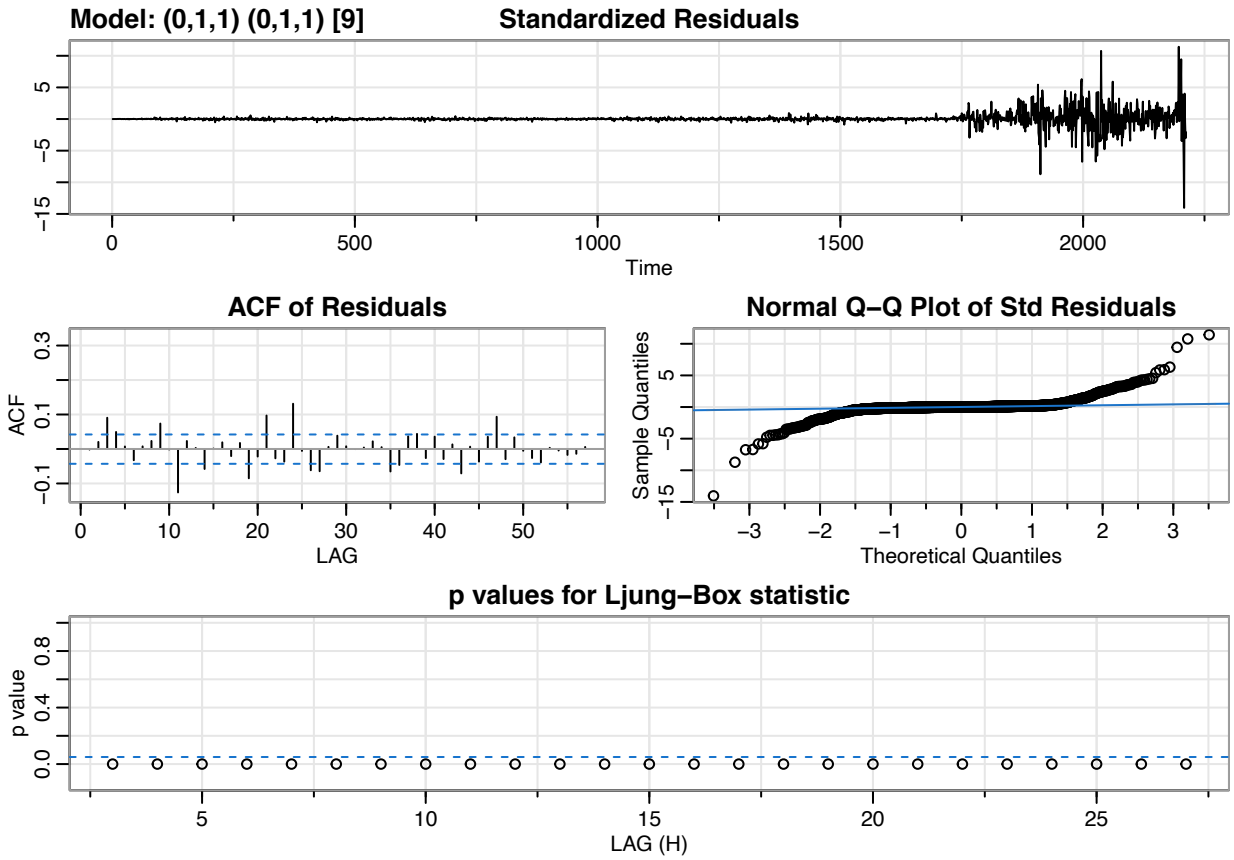
2.3 Method 3: Box-jenkins

The third method is called Box-jenkins. This method is primarily a combination of MA and AR model. MA stands for "Moving average" and AR stands for "Autoregression Method".



By observation and assumption, we set our seasonality to be 9.
From above, we remove the trend once and remove the seasonality once, so d = 1, D = 1.
Ignoring the non-seasonal lag, the ACF is cut off after the first seasonal lag, and the partial ACF has an exponential decay, hence P = 0, Q = 1.
Ignoring the seasonal lags, the ACF seems to be cut off at lag 1 and partial ACF has damped sine wave, hence p=0, q =1.

From above, We propose a model(p=0,d=1,q=1,P=0,D=1,Q=1,S=9).



Observation: For the first plot(standardized residuals), standardized residuals generally have zero variance but it changes afterwards, which means the variance is not constant.

For the second plot(ACF of Residuals), there is no significant lags, so the residuals are uncorrelated.

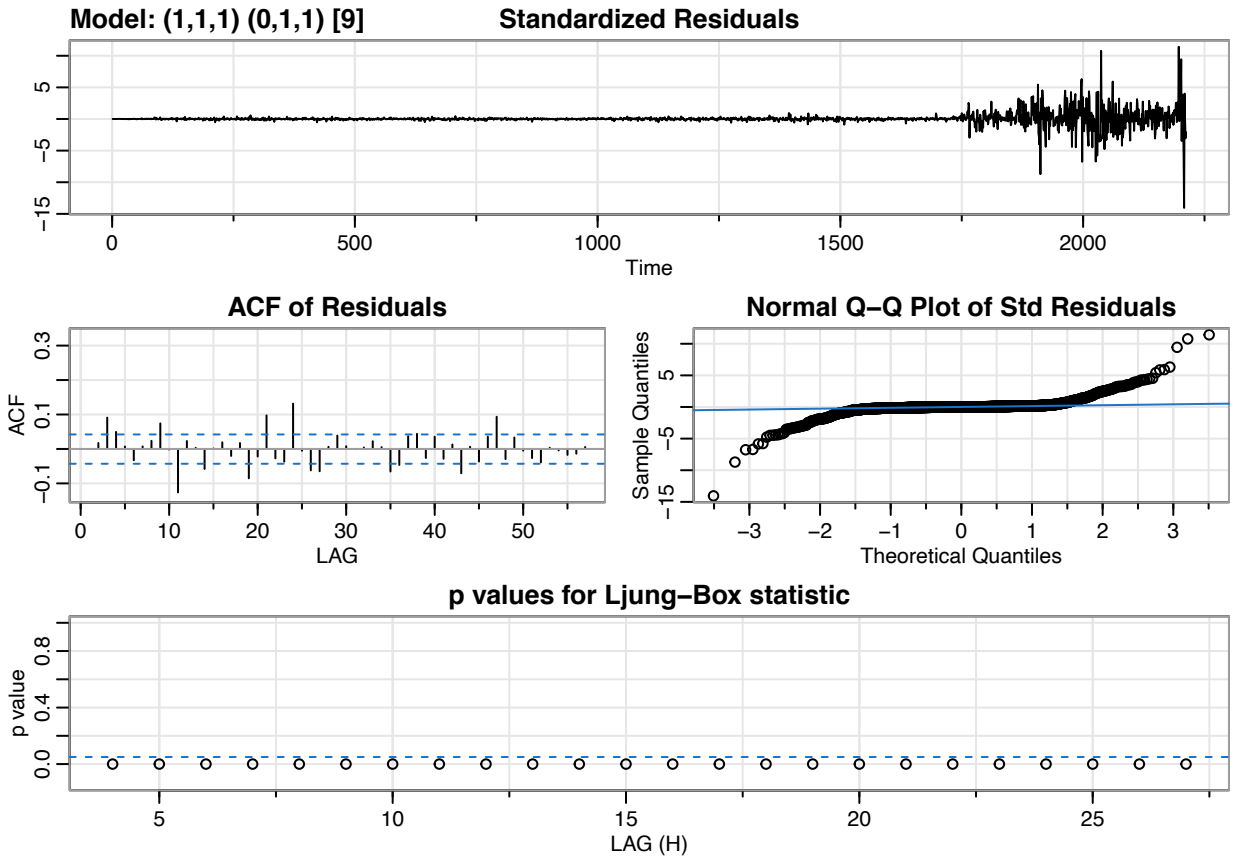
For the third plot(QQ plot), the points are not on a straight line, so it does not follow a normal distribution.

For the fourth plot(p value for Ljung-Box statistic), most of the points are under the blue lines, which means the model we choose is not good enough.

Now we have our second model:

Ignoring the seasonal lags, both the ACF and the partial ACF show the damp since it is a wave pattern. Hence p = 1, and q = 1.

From above, we propose another model(p=1,d=1,q=1,P=0,D=1,Q=1,S=9).

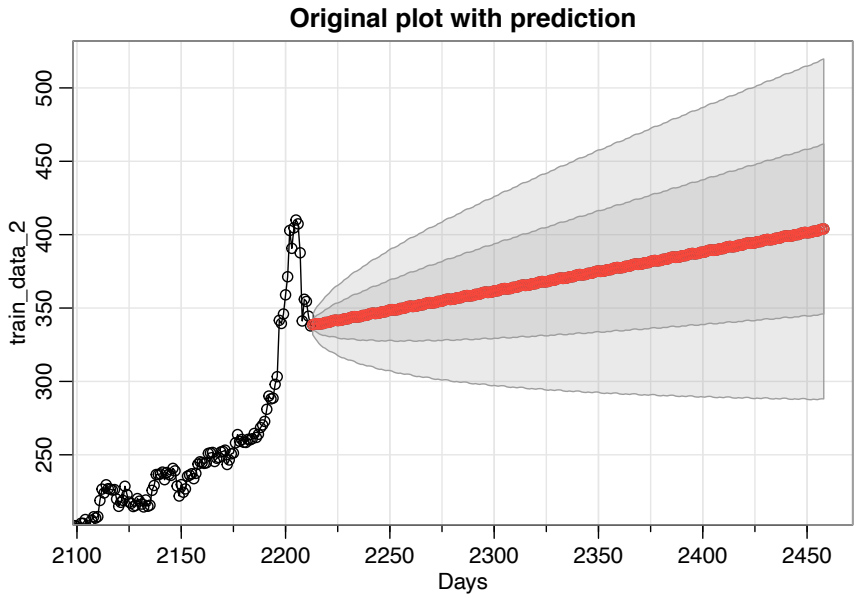


This model with p=1,d=1,q=1,P=0,D=1,Q=1,S=9 shows similar results to the first model above with p=0,d=1,q=1,P=0,D=1,Q=1,S=9. Thus, it is hard to select the best model based on these ACF plots and residuals. We then compare the Akaike Information Criterion(AIC), AICc ,and Bayesian Information Criterion(BIC) to decide which model is better.

Table 3: Summary of AIC, AICc, and BIC

	AIC	AICc	BIC
Model 1(p=0,d=1,q=1,P=0,D=1,Q=1,S=9)	5.293237	5.293240	5.300999
Model 2(p=1,d=1,q=1,P=0,D=1,Q=1,S=9)	5.294074	5.294079	5.304423

From above, model 1(p=0,d=1,q=1,P=0,D=1,Q=1,S=9) has the lowest value and the MSE of model 1 is 10618.6



For each method, we pick the relatively good model for each category. Now, we compare each MSE and we find that simple exponential smoothing has the lowest MSE. Therefore, within these three categories, the simple exponential smoothing is the relatively best model to predict future Tesla's stock prices.

This is the end of the first part. Prediction plays an important role in QRM. Using these statistical models, we are able to predict the range of future stock prices and then we can apply risk management techniques to calculate maximum losses and so on. This leads to the next section of this report, which is estimating the value at risk(VaR) and expected shortfall(ES) for a given portfolio consisting of 10 shares of Tesla and 1 share of Gold.

3 Section II: Estimation of Value at Risk(VaR) and Expected Shortfall(ES)

According to chapter 1, one type of risks in QRM is market risk which refers to the risk of loss in a financial position due to changes in the underlying component like stock price [2]. This section of the report focuses on measuring the market risks from stock price given a specific portfolio. According to Chapter 2, to measure the risks, the following key statistical tasks of Quantitative Risk Management(QRM) are needed to be tackled [3]:

1) Find a statistical model for risk factor changes X_{t+1}

2) Compute the cdf $F_{L_{t+1}}$ of L_{t+1}

3) Compute the risk measure from $F_{L_{t+1}}$

[2]Definition of risk types can be found in Chapter 1 of the lecture notes.
[3]Key statistical tasks of QRM can be found in Chapter 2 of the lecture notes.

Nevertheless, sometimes it is hard to directly compute the cdf of L_{t+1}, thus, sometimes we will need to use linearized loss L^{Δ}_{t+1} with cdf $F_{L^{\Delta}_{t+1}}$ as approximation to L_{t+1} with cdf $F_{L_{t+1}}$. There are three general approaches to solve this issue: analytical method, historical simulation and Monte Carlo. In this report, we will use variance-covariance method for analytical method, historical simulation, Monte Carlo and the Peaks-over-threshold (POT) in Chapter 5 to estimate the value at risk(VaR) and expected shortfall(ES).

We start by assuming that we have a portfolio consisting of 10 shares of Tesla and 1 share of Gold. The goal is to estimate value at risk and expected shortfall at confidence level α = 99% ($VaR_{0.99}(L)$ and $ES_{0.99}(L)$ where L is the loss of our portfolio) using the following methods:

1) Variance-covariance method (one type of analytical method)

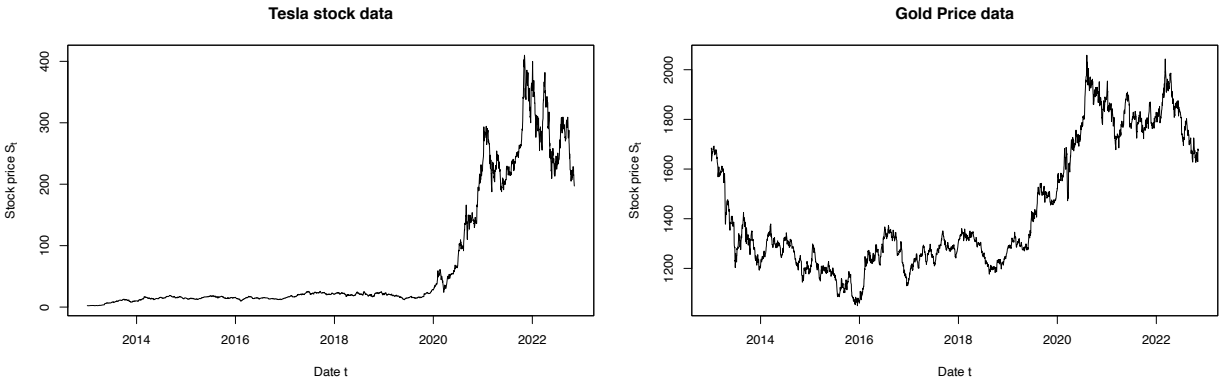
2) Historical Simulation

3) Monte Carlo (normal and t)

4) Peaks over threshold

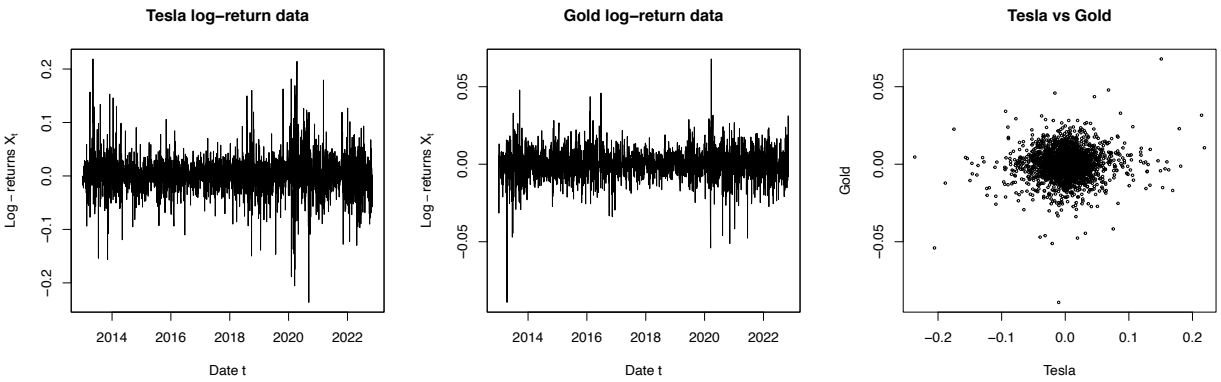
We are given portfolio of d = 2 stocks $S_{t,1},\ldots, S_{t,d}$ where d = 2, $S_{t,j}$ is the value of jth stock at time t, and λ_j is the number of shares of stock j (i.e. $\lambda_1 = 10$ representing 10 shares of Tesla, and $\lambda_2 = 1$ representing 1 share of Gold)

The two graphs below show the plots of each time series for Tesla ($S_{t,1}$) and Gold ($S_{t,2}$) stock price data



We then consider logarithmic prices as risk factors, i.e. $Z_{t,j} = logS_{t,j}$ for j = 1, 2. Then, the risk-factor changes can be computed as $X_{t+1,j} = Z_{t+1,j} - Z_{t,j}$ for j = 1, 2.

The plots below reveal the log return of each stock. Note that both plots show extreme events around 2020 for Tesla before 2014 or around 2020 for Gold and and volatility clustering.



We will then estimate value at risk(VaR) and expected shortfall(ES) using the four methods mentioned above. This is implemented in R using code from R Script of R_Standard_methods_for_market_risk by Professor Erik Hintz.

3.1 Method 1: Variance-Covariance Method

The main idea is to choose $F_{X_{t+1}}$ and f such that $F_{L_{t+1}}$ can be determined explicitly.

For the variance-covariance method, we make the following assumptions [4]:

(1) the risk factor changes $X_{t+1} \sim N_d(\mu, \Sigma)$

(2) the linearized loss's cdf $F_{L^{\Delta}_{t+1}}$ is a good approximation to $F_{L_{t+1}}$

Then, we have $L^{\Delta}_{t+1} \sim N(-c_t - b_t^T \mu, b_t^T \Sigma b_t)$.

Using R, we obtained the required parameters:

$$\hat{\mu} = \begin{bmatrix} \hat{\mu_1} \\ \hat{\mu_2} \end{bmatrix} = \begin{bmatrix} 1.800956 \times 10^{-3} \\ -1.510553 \times 10^{-6} \end{bmatrix}$$

where $\hat{\mu_1}$ is for Tesla, and $\hat{\mu_2}$ is for Gold

$$\hat{\Sigma} = \begin{bmatrix} 0.001282499 & 2.230600 \times 10^{-5} \\ 0.000022306 & 8.901822 \times 10^{-5} \end{bmatrix}, E(L^{\hat{\Delta}}_{t+1}) = -3.546793, \ SD(L^{\hat{\Delta}}_{t+1}) = 73.33769$$

Thus, using R, the value at risk at 99% confidence level can be calculated as $VaR_{0.99}(L) = \mu + \sigma\Phi^{-1}(\alpha) = 167.0754$, and the expected shortfall at 99% confidence level can be calculated as $ES_{0.99}(L) = \mu + \sigma \frac{\phi(\Phi^{-1}(\alpha))}{1-\alpha} = 191.9310$ by R.

[4] Definition of variance-covariance method comes from lecture 2 page 18.

3.2 Method 2: Historical Simulation

The main idea is to estimate $F_{L_{t+1}}$ by its empirical distribution function (edf) [5].

Given n past risk factor changes $X_{t-n+1}, ..., X_t$, if we let

$L_k = L(X_k) = -(f(t+1, Z_t + X_k) - f(t, Z_t)), k \in \{t-n+1, ..., t\}$,

then $F_{L_{t+1}}$ can be estimated via

$\hat{F}_{L_{t+1,n}}(x) = \frac{1}{n} \sum_{i=1}^{n} 1_{\{L_{t-i+1} \leq x\}}, x \in \mathbb{R}$

Thus, we can estimate the VaR and ES nonparametrically through R, and it gives us $VaR_{0.99}(L) = 183.5529$, and $ES_{0.99}(L) = 263.9727$

3.3 Method 3: Monte Carlo (normal and t)

The main idea of Monte Carlo method is to take any model for X_{t+1} to simulate it. Then, we compute corresponding losses L_k and estimate $F_{L_{t+1}}$ typically via $\hat{F}_{L_{t+1,\ n}}(x)$ as shown in the historical simulation method. We will use two methods here: Monte Carlo based on a fittted multivariate normal distribution and Monte Carlo based on a fitted multivariate t distribution [6].

For Monte Carlo using a fitted multivariate normal distribution, we get

$$\hat{\mu} = \begin{bmatrix} \hat{\mu_1} \\ \hat{\mu_2} \end{bmatrix} = \begin{bmatrix} 1.808649 \times 10^{-3} \\ -2.236954 \times 10^{-6} \end{bmatrix}$$

where $\hat{\mu_1}$ is for Tesla, and $\hat{\mu_2}$ is for Gold

$$\hat{\Sigma} = \begin{bmatrix} 1.282876 \times 10^{-3} & 2.232882 \times 10^{-5} \\ 2.232882 \times 10^{-5} & 8.905318 \times 10^{-5} \end{bmatrix}$$

Similarly, we get $VaR_{0.99}(L) = 160.3882$ and $ES_{0.99}(L) = 180.7825$

For Monte Carlo using a fitted multivariate t distribution, we first use R to fit a multivariate t distribution and compute corresponding losses. Using R, we get

$$\hat{\mu} = \begin{bmatrix} \hat{\mu_1} \\ \hat{\mu_2} \end{bmatrix} = \begin{bmatrix} 0.0017113309 \\ 0.0001169642 \end{bmatrix}$$

where $\hat{\mu_1}$ is for Tesla, and $\hat{\mu_2}$ is for Gold.

The fitted covariance matrix Sigma is:

$$\hat{\Sigma} = \begin{bmatrix} 1.261467 \times 10^{-3} & 6.181291 \times 10^{-6} \\ 6.181291 \times 10^{-6} & 9.920416 \times 10^{-5} \end{bmatrix}$$

[5] Definition and Method of Historical Simulation comes from Lecture Note 2 page 20

[6] Definition and concept of Monte Carlo method comes from Lecture Note 2 page 22.

The degree of freedom is 3.789158.

Then, using R to compue the corresponding VaR and ES, we get:

$VaR_{0.99}(L) = 184.835$ and $ES_{0.99}(L) = 264.7508$

3.4 Method 4: Peaks Over Threshold(POT)

The POT method is used for estimating the distribution of excess losses above a threshold, and it is a more efficient use of the data on extreme events than the block maximum method. The key idea is to select a reasonable threshold u. We need to select u as the smallest point where $e_n(v), v \geq u$ becomes linear. When selecting the threshold u, sample mean excess plot is an approach. However, it is not good for particularly large threshold u, due to lack of data and there is a bias-variance trade off. Thus, it is important to choose a sufficiently large threshold u. In this scenario, we set u = 72.29727. We then find out excess losses: $Y_k = \tilde{X_k} - u,\ k \in \{1, ..., N_u\}$ where $\tilde{X_1}, ..., \tilde{X_{N_u}}$ are the exceedances and N_u is the number of exceedances over the threshold u. We assume each Y_k is independent and use Maximum Likelihood Estimation(MLE) to estimate ξ and β for $G_{\xi,\beta}(x)$. Then, by Pickands-Balkma-de Haan theorem, the excess losses over the chosen threshold u can be approximated by a GPD, where the parameters can be estimated by maximum likelihood as mentioned above. After fitting the GPD model, we assume that we have $F_u(x) = G_{\xi,\beta}(x),\ x \in [0, x_F - u)$ and $\xi \neq 0$. Then, we can calculate our risk measures $VaR_\alpha(X)$ and $ES_\alpha(X)$ [7].

By definition of $VaR_\alpha(x)$, we can let $\alpha = F(x) \Rightarrow 1 - \alpha = \bar{F}(x)$

Thus, $1 - \alpha = \bar{F}(u)(1 + \xi \cdot \frac{x-u}{\beta})^{-\frac{1}{\xi}}$

Solving for x, we can get our risk measure VaR: $VaR_\alpha(X) = u + \frac{\beta}{\xi}((\frac{1-\alpha}{\bar{F}(u)})^{-\xi} - 1),\ \alpha \geq F(u)$

Also, the risk measure, expected shortfall can be computed by its definition once we had the value at risk.

$ES_\alpha(X) = \frac{1}{1-\alpha}\int_\alpha^1 VaR_s(X)\ ds = \frac{1}{1-\alpha}\int_\alpha^1 u + \frac{\beta}{\xi}((\frac{1-s}{\bar{F}(u)})^{-\xi} - 1)\ ds$

Solving for it, we get the expected shortfall: $ES_\alpha(X) = \frac{VaR_\alpha(X)}{1-\xi} + \frac{\beta - \xi u}{1-\xi},\ \xi < 1$

Using R to compute the VaR and ES at $\alpha = 0.99$ level, we get $\xi = 0.1194852,\ \beta = 44.96117$ and $VaR_{0.99}(X) = 191.563,\ ES_{0.99}(X) = 258.8096$

3.5 Results of Estimating VaR and ES

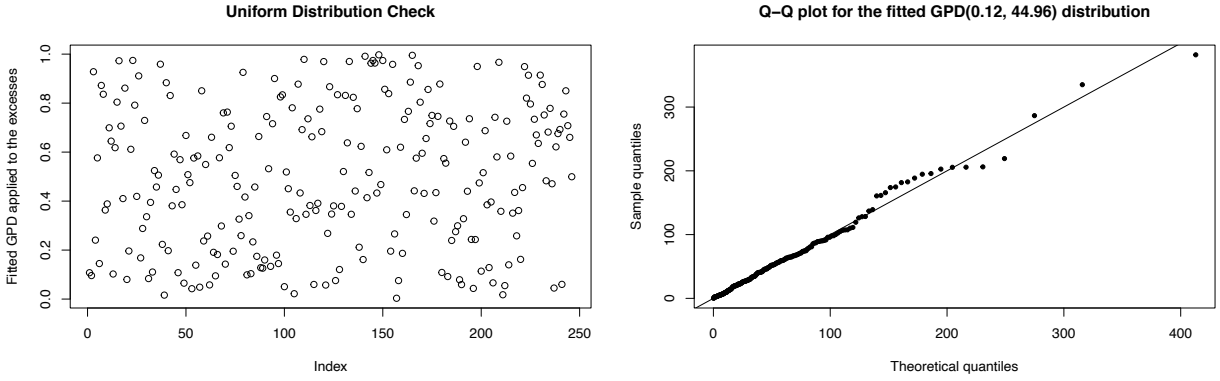
The table below provides a summary of the two risk measures, VaR and ES for all methods at $\alpha = 0.99$. From the table, expected shortfall is always greater than or equal to value at risk. POT method gives us the largest estimated value at risk of 191.5630, meaning that the risk of this portfolio having a loss larger than 191.5630 is at most 1% cases on average. Also, the Monte Carlo method based on multivariate normal distribution yields the smallest value at risk of 160.3882. From this table, it is hard to tell which method performs better.

[7] Concept of POT and formula come from lecture note 12 page 18

Table 4: VaR and ES for all Methods at alpha = 0.99

	VaR	ES
MC (normal)	160.3882	180.7825
Var.-cov.	167.0754	191.9310
Hist. sim.	183.5529	263.9727
MC (Student t)	184.8350	264.7508
POT	191.5630	258.8096

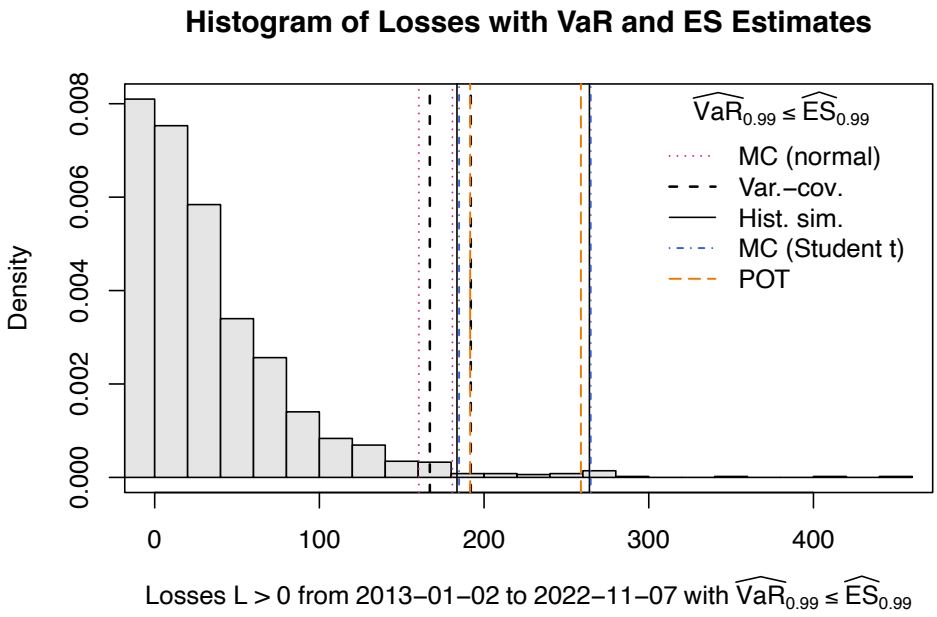
Since POT gives the largest value at risk and expected shortfall, we will first check the goodness-of-fit for the fitted GPD distribution function. From the two plots below, we can see that the resulting sample roughly follows a standard uniform distribution since all the points are within 0 and 1. From the Q-Q plot for the fitted GPD distribution, we could see that almost all the points lie on the line except some outliers. Thus, the fitted GPD distribution has good graphical fitness.



After examing the fitness of the fitted GPD distribution, we will further analyze the performance of the four methods on estimating VaR and ES.

Table 5: Summary Statistics of Historical Losses

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-524.7913	-39.21357	-3.635066	-4.892362	30.75479	454.2253



From the histogram plot, we make the following observations:

- The Monte Carlo method based on a fitted multivariate normal distribution and the variance-covariance method lead to similar results since they both assume multivariate normal distributed risk-factor changes but differ in the computation of the loss distribution (the variance-covariance method is analytical, while the Monte Carlo based on a fitted multivariate normal distribution is a type of empirical method).
- The Monte Carlo method(with multivariate normal distribution) and the variance-covariance method both underestimate VaR and ES in comparison to the historical simulation method.
- The Monte Carlo method based on multivariate student t distribution shows very close results on estimation of VaR and ES with the historical simulation method
- An interesting point is that POT method slightly overestimate VaR but slightly underestimate ES compared with the historical method.
- The historical simulation method implies that the loss distribution is more heavy-tailed. This is captured quite well by POT method and the Monte Carlo method for multivariate t distributed risk-factor changes (degrees of freedom are MC.t\$df = 3.789158).
- POT method and Monte Carlo for student t distribution both have slight departure for ES. It's overall reassuring that several methods including historical simulation, POT method and Monte Carlo for student t distribution lead to similar results.
- Thus, in the range between around 180 and 260, one can then determine an adequate amount of risk capital

4 Conclusion

In the first section of this report, we demonstrate three methods of prediction: polynomial regression, Holtwinter, and Box-jenkins. For each category, we adjust function's parameters to test models' prediction power by MSE. Eventually, we make the conclusion that simple smoothing model in Holtwinter category will have the best performance in predicting Tesla's stock prices within the three category. In the second section, we create a portfolio with 10 shares of Tesla and 1 share of gold and we use four methods: variance-covariance method, historical simulation, Monte Carlo(multivariate normal and t distribution), and peaks over threshold(POT) to estimate value at risk(VaR) and expected shortfall(ES) at $\alpha = 0.99$ level. It can be concluded that Monte Carlo based on multivariate normal distribution has the smallest VaR and ES(160.3882, 180.7825 respectively), followed by variance-covariance method. They both have similar results since they both assume a multivariate normal distribution. Also, they show an underestimate of VaR and ES compared with historical simulation. While, POT has the largest VaR and ES (191.5630, 258.8096). It can be concluded that historical simulation, Monte Carlo based on multivariate t distribution and maybe POT show similar results.

Reference

Gold. (2013, January). Yahoo Finance. Retrieved December 10, 2022, from https://finance.yahoo.com/quote/GC=F/

McNeil, A. J., Frey, R., & Embrechts, P. (2015). *Quantitative Risk Management: Concepts, Techniques and Tools - Revised Edition.* Amsterdam University Press.

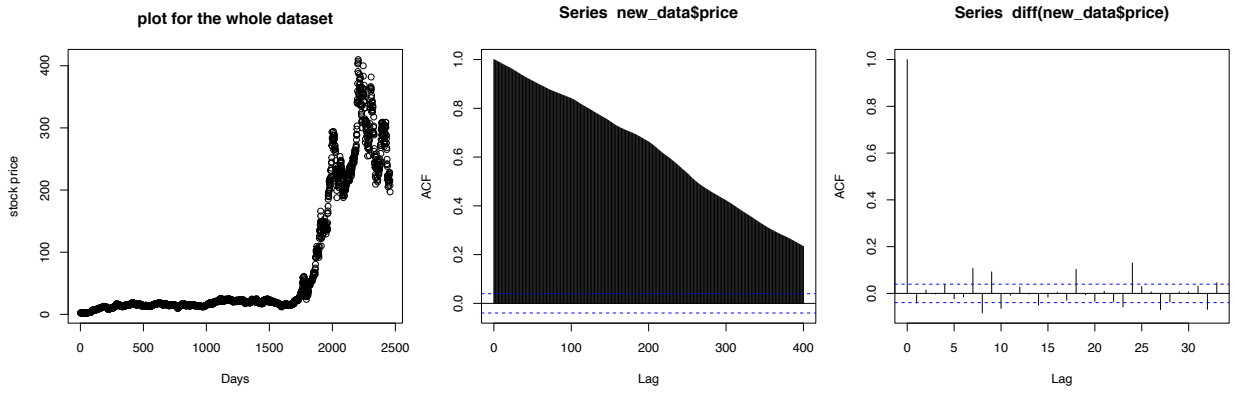
Shumway, R. H., & Stoffer, D. S. (2017). *Time Series Analysis and Its Applications: With R Examples (Springer Texts in Statistics)* (4th ed. 2017). Springer. https://link.springer.com/book/10.1007/978-3-319-52452-8

Tesla, Inc. (TSLA). (2022, January). Yahoo Finance. Retrieved December 10, 2022, from https://finance.yahoo.com/quote/TSLA/

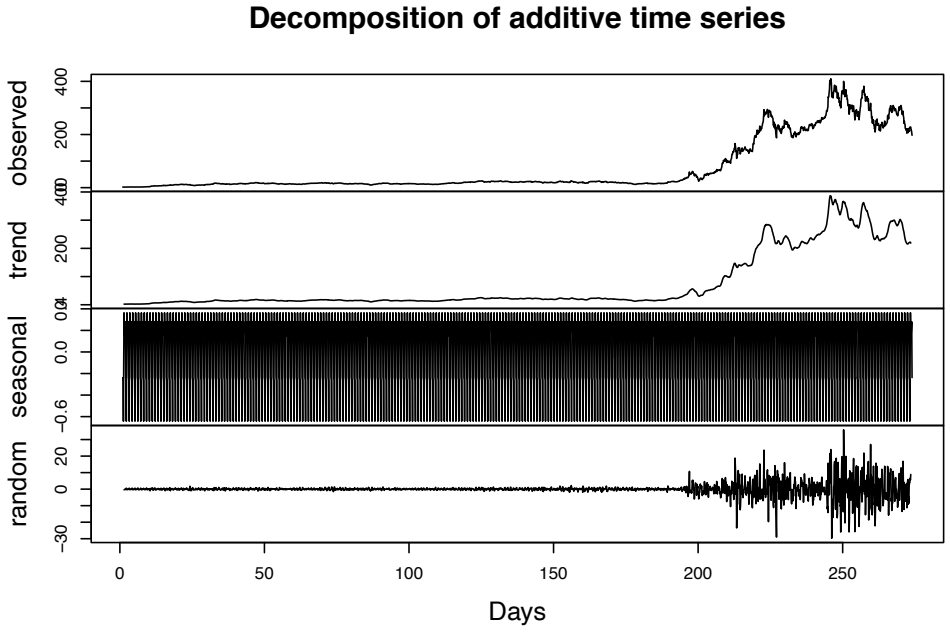
Appendix

```
#load the dataset including Tesla and gold stock prices
portfolio_data = read.csv("data.csv")
library(mvtnorm) # for sampling from a multivariate t distribution
library(QRM) # for fit.mst()
library(xts) # for na.fill()
library(qrmdata) # for the data
# for the data analysis; get the latest version
# via install.packages("qrmtools", repos = "http://R-Forge.R-project.org"):
library(qrmtools)
```

```
data = read.csv("data.csv") ## data combined with price of Tesla and Gold
price = data$Tesla.Price
New_price = as.numeric(price)
new_data = data.frame(X = c(1:length(New_price)),price = New_price)
par(mfrow = c(1,3), cex = 0.95, mar=c(4,4,4,1), oma=c(0.5,0.5,0.5,0))
plot(new_data$X,new_data$price, xlab = "Days", ylab = "stock price",
     main = "plot for the whole dataset")
acf(new_data$price,lag.max = 400)
acf(diff(new_data$price))
```

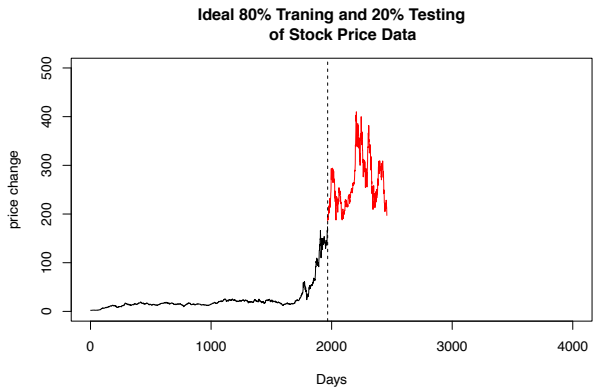


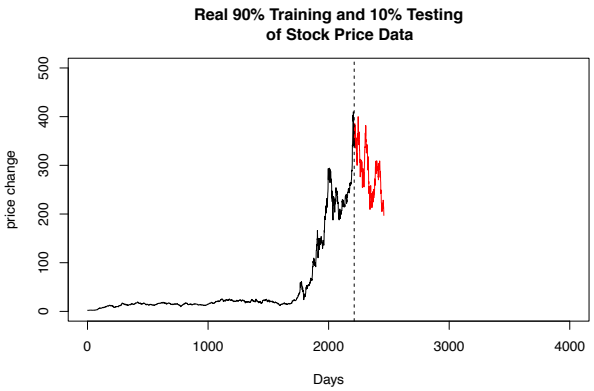

```
plot(decompose(ts(new_data$price, start = 1, frequency = 9)),xlab = "Days")
```




```
## We found out that lag 9,18,27 excess the 95% range
```

```
par(mfrow = c(1,2))
train_data = new_data$price[1:1966]
test_data = new_data$price[1967:2458]

plot(train_data, type = "l", xlab = "Days", ylab = "price change",
     xlim = c(0, 4000), ylim = c(0,500), main = "Ideal 80% Traning and 20% Testing
     of Stock Price Data")
lines((1967:2458), test_data,col = "red")
abline(v = 1967, lty = 2)

train_data_2 = new_data$price[1:2212]
test_data_2 = new_data$price[2213:2458]

plot(train_data_2, type = "l", xlab = "Days", ylab = "price change",
     xlim = c(0, 4000), ylim = c(0,500), main = "Real 90% Training and 10% Testing
     of Stock Price Data")
lines((2213:2458), test_data_2,col = "red")
abline(v = 2212, lty = 2)
```





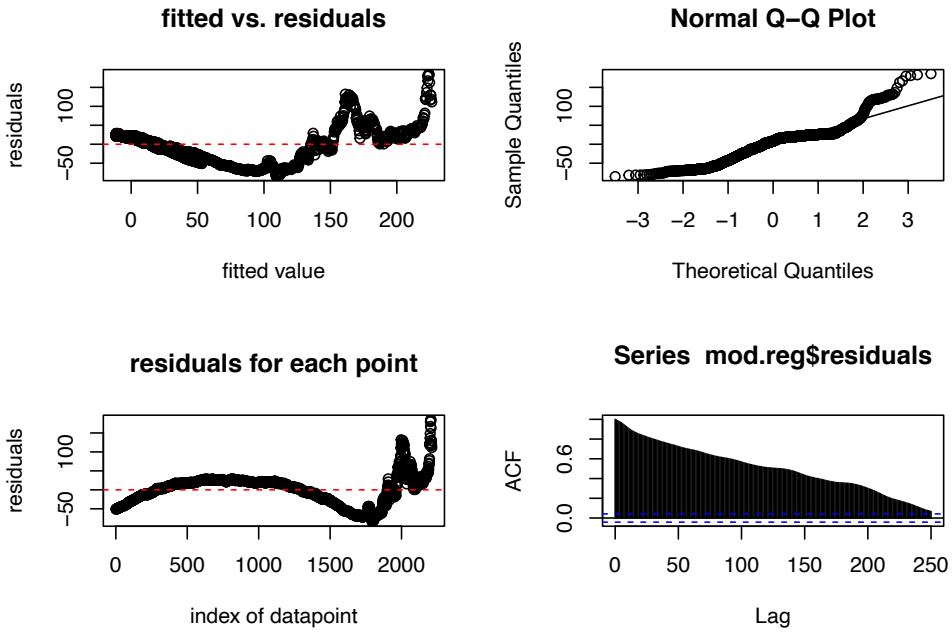

```
day_train = 1:2212
day_test = 2213:2458

period_train = as.factor(c(rep(1:9,245),1:7))
period_test = as.factor(c(8,rep(1:9,27),1,2))
mse.reg = rep(0,5)
mse.reg.withseasonal = rep(0,5)

for(i in 1:5){
  mod.reg1 = lm(train_data_2 ~ poly(day_train, i))
  pred.reg1 = predict(mod.reg1, data.frame(day_train = day_test))
  mse.reg[i] = mean((pred.reg1 - test_data_2)**2)
  mod.reg2 = lm(train_data_2 ~ poly(day_train, i) +period_train)
  pred.reg2 = predict(mod.reg2, data.frame(day_train = day_test,
  period_train = period_test))
  mse.reg.withseasonal[i] = mean((pred.reg2 - test_data_2)**2)
}

MSE = cbind(mse.reg, mse.reg.withseasonal)
rownames(MSE) = c("Polynomial with degree 1", "Polynomial with degree 2",
                  "Polynomial with degree 3", "Polynomial with degree 4",
                  "Polynomial with degree 5")
colnames(MSE) = c("MSE", "Regression with seasonality's MSE")

# Make a table
kable(MSE, caption = "Summary of MSE",
      valign = 't')%>%
  kable_styling(full_width = F, position = 'center')%>%
  kable_styling(latex_options = "HOLD_position")
```

Table 6: Summary of MSE

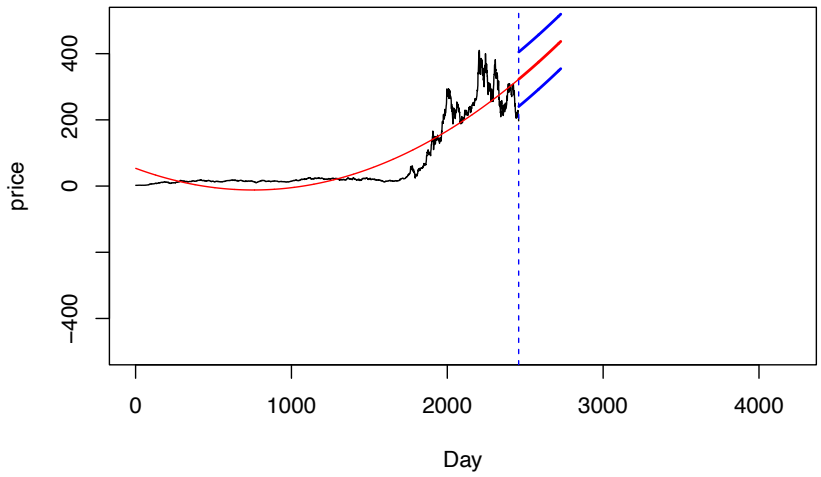
	MSE	Regression with seasonality's MSE
Polynomial with degree 1	23556.346	23556.642
Polynomial with degree 2	4945.286	4944.907
Polynomial with degree 3	24646.043	24646.643
Polynomial with degree 4	57681.985	57685.380
Polynomial with degree 5	50060.571	50064.580

```
p.best = which.min(mse.reg)
##degree 2, include seasonal component
```

```
mod.reg = lm(train_data_2~ poly(day_train,p.best))
par(mfrow=c(2,2))
plot(mod.reg$fitted.values,mod.reg$residuals,xlab = "fitted value",ylab = "residuals",
     main = "fitted vs. residuals")
abline(h=0,lty=2,col="red")
qqnorm(mod.reg$residuals)
qqline(mod.reg$residuals)
plot(mod.reg$residuals,xlab = "index of datapoint",ylab = "residuals",
     main = "residuals for each point")
abline(h=0,lty=2,col="red")
acf(mod.reg$residuals,lag.max = 250)
```



```
day =1:2458
period = as.factor(c(rep(1:9,273),1))
final_reg = lm(new_data$price ~poly(day,p.best))
day_new = 2458 +(1:270)
period_new = as.factor(c(2:9,rep(1:9,29),1))
prediction_reg = predict(final_reg,
                         data.frame(day = day_new,period = period_new),
                         interval = "prediction")
plot(new_data$price, xlim = c(0, 4200), type = "l",
    ylim = c(-500, 500), xlab = "Day", ylab = "price")
lines(as.numeric(day),fitted(final_reg),col="red")
lines(day_new, prediction_reg[, 1], col = "red", lwd = 2)
lines(day_new, prediction_reg[, 2], col = "blue", lwd = 2)
lines(day_new, prediction_reg[, 3], col = "blue", lwd = 2)
abline(v = length(new_data$price), col = "blue", lty = 2)
```




```
train_data_ts = ts(train_data_2, start = 1, frequency = 9)
# simple exponential smoothing
exponential <- HoltWinters(train_data_ts, gamma = FALSE, beta = FALSE)
exponential_predict = predict(exponential, n.ahead = 246)
mse_exponential = mean((test_data_2- exponential_predict)^2)
### This is the smallest MSE we have.
```

```
train_data_ts = ts(train_data_2, start = 1, frequency = 9)
# double exponential smoothing
dou_exponential <- HoltWinters(train_data_ts, gamma = FALSE)
```

```
dou_exponential_predict = predict(dou_exponential, n.ahead = 246)
mse_dou_exponential = mean((test_data_2- dou_exponential_predict)^2)
```

```
# additive
additive <- HoltWinters(train_data_ts, seasonal = "additive")
additive_predict = predict(additive, n.ahead = 246)
mse_additive_predict = mean((test_data_2 - additive_predict)^2)
```

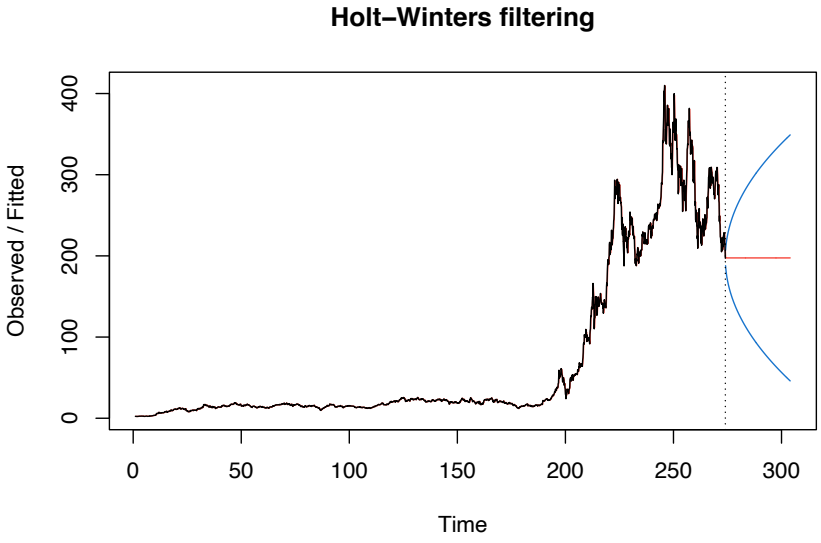
```
# multiplicative
multiplicative <- HoltWinters(train_data_ts, seasonal = "multiplicative")
multiplicative_predict = predict(multiplicative, n.ahead = 246)
mse_multiplicative_predict = mean((test_data_2 - multiplicative_predict)^2)
```

```
# Make a table to summarize the MSEs
tab = data.frame(Model = c("Simple Exponential Smoothing",
                      "Double Exponential Smoothing",
                      "Additive-HoltWinters",
                      "Multiplicative-HoltWinters "),
           MSE = c(4684.215, 20895.7, 20880.38, 21186.49))

kable(tab, caption = "MSE of Holtwinter and Smoothing Methods",valign = 't')%>%
  kable_styling(full_width = F)%>%
  kable_styling(latex_options = "HOLD_position")
```

Table 7: MSE of Holtwinter and Smoothing Methods

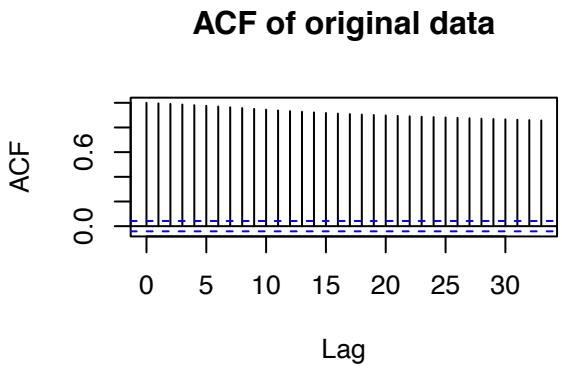
Model	MSE
Simple Exponential Smoothing	4684.215
Double Exponential Smoothing	20895.700
Additive-HoltWinters	20880.380
Multiplicative-HoltWinters	21186.490

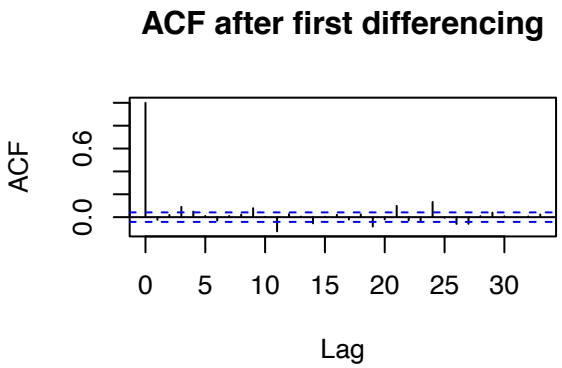
```
final_smoothing <- HoltWinters(ts(new_data$price,start = 1, frequency = 9),
                               gamma = FALSE, beta = FALSE)
plot(final_smoothing, predict(final_smoothing, n.ahead = 270,
                              prediction.interval = TRUE))
```

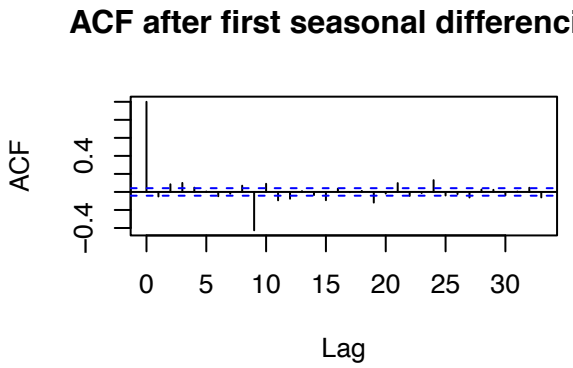


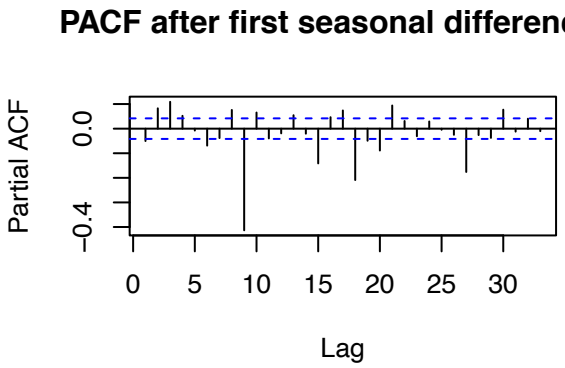

```
## due to amount of dataset, Holt wnter prediction overlap with original data.
##If we zoom in the plot, we can still observe red line.
```

```
par(mfrow = c(2, 2))
acf(train_data_2,main = "ACF of original data")
acf(diff(train_data_2),main = "ACF after first differencing")
acf(diff(diff(train_data_2),lag =9),main = "ACF after first seasonal differencing")
pacf(diff(diff(train_data_2),lag =9),main = "PACF after first seasonal differencing")
```

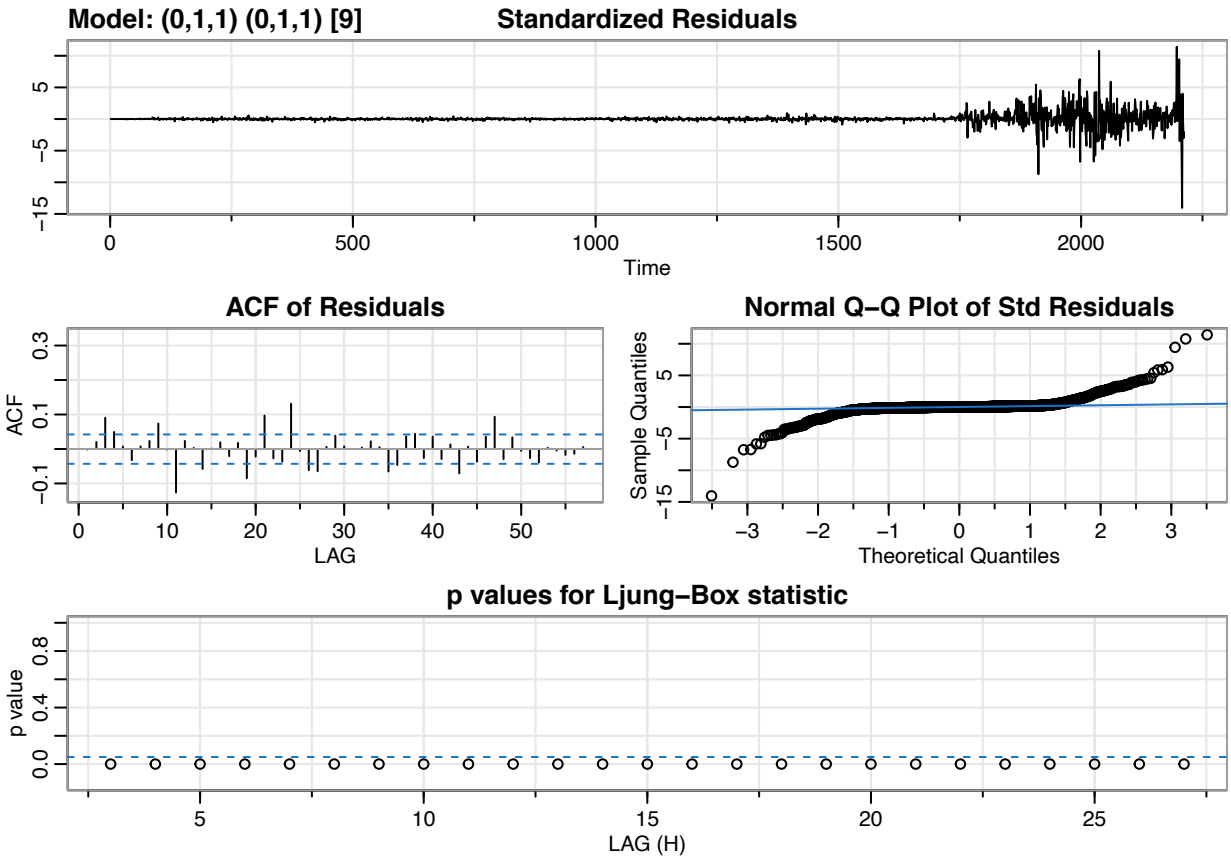




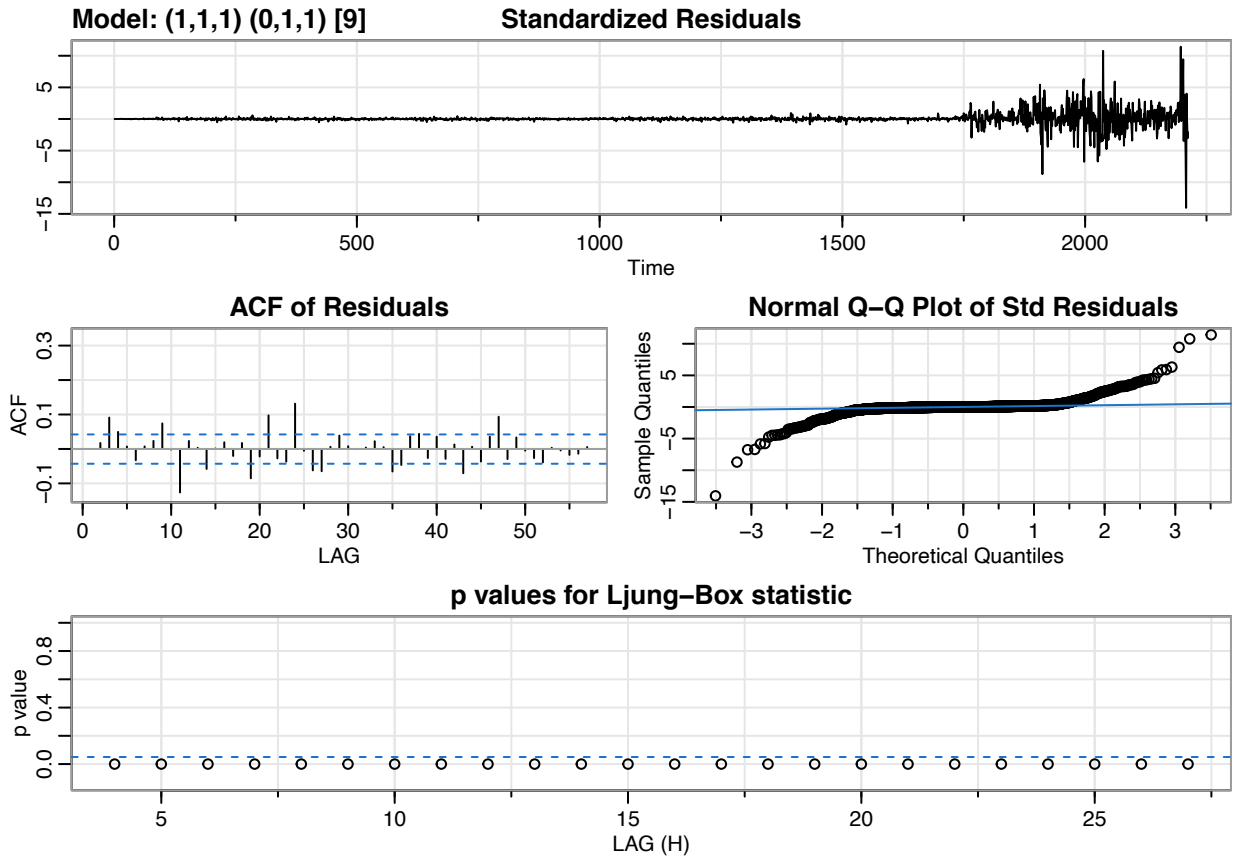





```
model1 = sarima(train_data_2,p=0,d=1,q=1,P=0,D=1,Q=1,S=9)
```




```
model2 = sarima(train_data_2,p=1,d=1,q=1,P=0,D=1,Q=1,S=9)
```



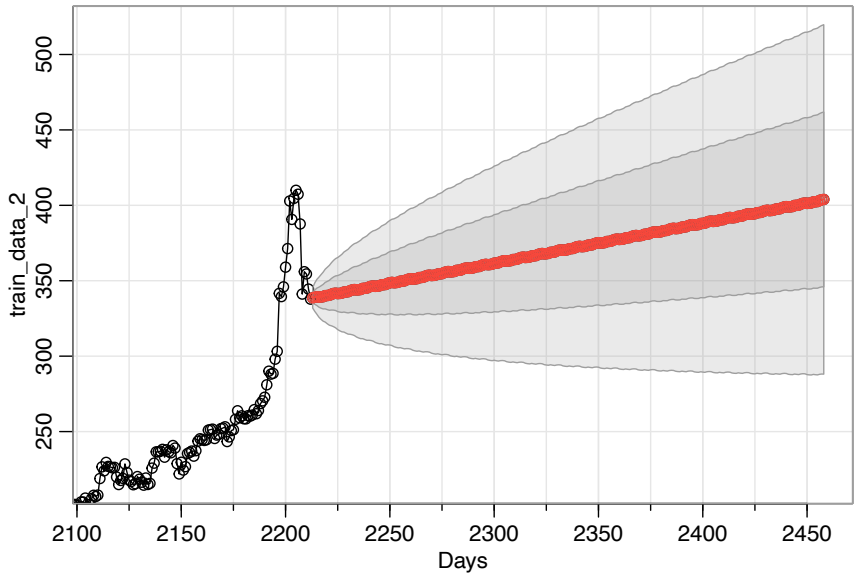

```
# Make a table for AIC, AICc, BIC
tab2 = data.frame(AIC = c(5.293237, 5.294074), AICc = c(5.293240, 5.294079),
                  BIC = c(5.300999, 5.304423))
rownames(tab2) = c("Model 1(p=0,d=1,q=1,P=0,D=1,Q=1,S=9)",
                   "Model 2(p=1,d=1,q=1,P=0,D=1,Q=1,S=9)")

kable(tab2, caption = "Summary of AIC, AICc, and BIC",valign = 't')%>%
  kable_styling(full_width = F)%>%
  kable_styling(latex_options = "HOLD_position")
```

Table 8: Summary of AIC, AICc, and BIC

	AIC	AICc	BIC
Model 1(p=0,d=1,q=1,P=0,D=1,Q=1,S=9)	5.293237	5.293240	5.300999
Model 2(p=1,d=1,q=1,P=0,D=1,Q=1,S=9)	5.294074	5.294079	5.304423

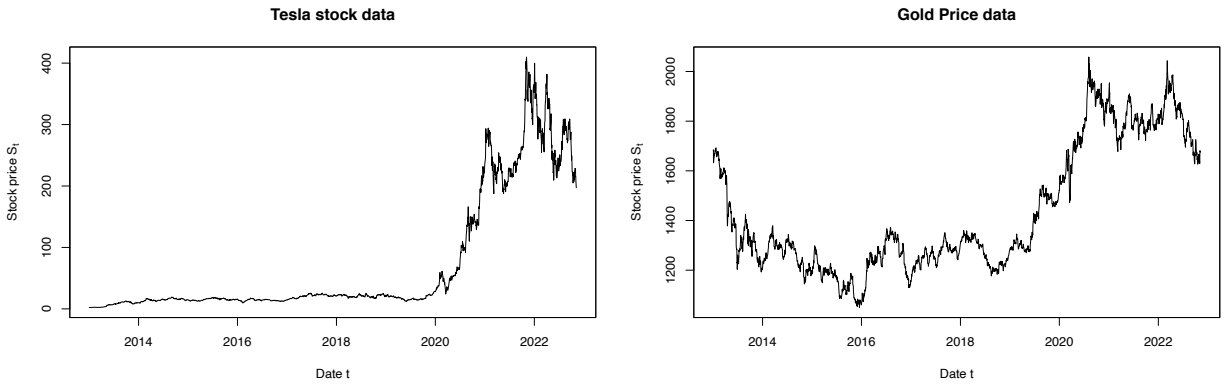
From above, model 1(p=0,d=1,q=1,P=0,D=1,Q=1,S=9) has the lowest value and the MSE of model 1 is 10618.6

```
prediction_Box = sarima.for(train_data_2,xlab = "Days",n.ahead=246,
                            p=0,d=1,q=1,P=0,D=1,Q=1,S=9)
```

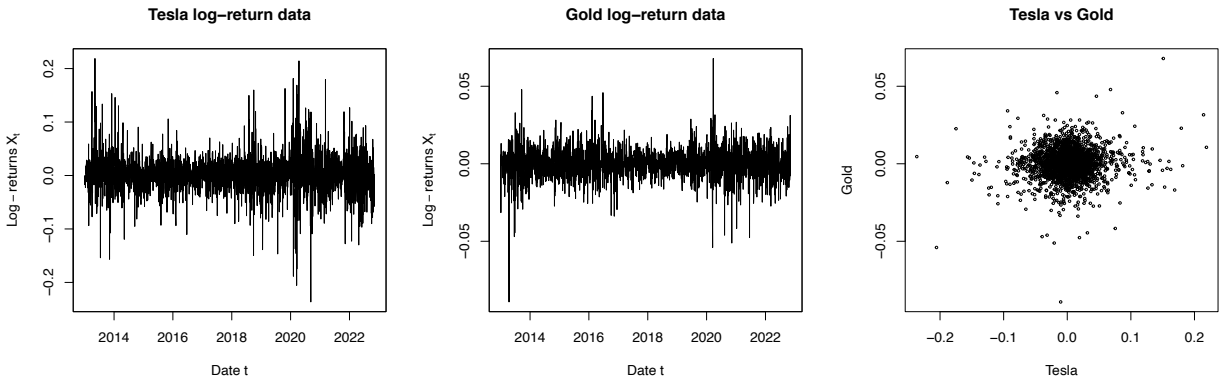



```
MSE1 = mean((prediction_Box$pred-test_data_2)^2)
```

```
S = read.zoo(portfolio_data) # convert the dataframe into zoo type

par(mfrow = c(1,2), cex = 0.95)
## Use scatter plots of each time series to check if anything is 'suspicious'
plot.zoo(S[,"Tesla.Price"], main = "Tesla stock data",
         xlab = "Date t", ylab = expression(Stock~price~S[t]))
plot.zoo(S[,"Gold.Price"], main = "Gold Price data",
         xlab = "Date t", ylab = expression(Stock~price~S[t]))
```



```r
## Compute the risk-factor changes and plot them (here: against each other)
X = returns(S) # compute log-returns:
colnames(X) <- c("Tesla", "Gold")

par(mfrow= c(1,3), cex = 1)
plot.zoo(X[,"Tesla"], main = "Tesla log-return data",
        xlab = "Date t", ylab = expression(Log-returns~X[t]))
plot.zoo(X[,"Gold"], main = "Gold log-return data",
         xlab = "Date t", ylab = expression(Log-returns~X[t]))
X <- as.matrix(X)
plot(X, cex = 0.4, main = "Tesla vs Gold")
```




```r
### ## The R code is used from "R_Standard_methods_for_market_risk" on learn
##by Professor Erik Hintz
### 2 Implement (and document) some auxiliary functions ########################

##' @title Compute the Loss Operator
##' @param x Matrix of risk-factor changes
##' @param weights Weights tilde{w}_{t,j} = lambda_j * S_{t,j}
##' @return Losses -sum_{j=1}^d(tilde{w}_{t,j}(exp(X_{t+1,j})-1))

loss_operator <- function(x, weights)
    -rowSums(expm1(x) * matrix(weights, nrow = nrow(x), ncol = length(weights),
                               byrow = TRUE))

##' @title Estimate VaR and ES
##' @param S Stock data, an (n, d)-matrix
##' @param lambda Number of shares of each stock
##' @param alpha Confidence level for VaR and ES
##' @param method A character string specifying the estimator
##' @param ... Additional arguments passed to the various methods
##' @return A list containing the estimated risk measures VaR and ES, and
```

```
##'         possibly other results (depending on the estimator)

risk_measure <- function(S, lambda, alpha,
                         method = c("Var.Cov", "hist.sim", "MC.N", "MC.t", "POT"),
                         ...)
{
    ## Input checks and conversions
    if(!is.matrix(S)) S <- rbind(S, deparse.level = 0L)
    stopifnot(0 < alpha, alpha < 1,
              length(lambda) == ncol(S), lambda > 0)
    method <- match.arg(method)

    ## Ingredients required for *all* methods
    X <- returns(S)[-1, ] # compute risk-factor changes
    if(!length(X)) stop("'S' should have more than just one line") # check
    S. <- as.numeric(tail(S, n = 1)) # pick out last available stock prices ("today")
    w. <- lambda * S. # weights tilde{w} today

    ## Method switch (now consider the various methods)
    switch(method,
           "Var.Cov" = { # variance-covariance method
               ## Estimate a multivariate normal distribution
               mu.hat <- colMeans(X) # estimate the mean vector mu
               Sigma.hat  <- var(X) # estimate the covariance matrix Sigma
               L.delta.mean <- -sum(w. * mu.hat) # mean of the approx. normal df
               #of the linearized loss L^{Delta}
               L.delta.sd <- sqrt(t(w.) %*% Sigma.hat %*% w.) # corresponding
               #standard deviation
               ## Compute VaR and ES and return
               qa <- qnorm(alpha)
               list(VaR = L.delta.mean + L.delta.sd * qa,
                    ES  = L.delta.mean + L.delta.sd * dnorm(qa) / (1-alpha))
           },
           "hist.sim" = { # historical simulation method
               ## Using nonparametrically estimated risk measures
               L <- loss_operator(X, weights = w.) # compute historical losses
               ## Nonparametrically estimate VaR and ES and return
               list(VaR = VaR_np(L, alpha),
                    ES  =  ES_np(L, alpha))
           },
           "MC.N" = { # Monte Carlo based on a fitted multivariate normal
               stopifnot(hasArg(N))
               N <- list(...)$N
               mu.hat <- colMeans(X) # estimate the mean vector mu
```

```
        Sigma.hat  <- var(X) # estimate the covariance matrix Sigma
        # simulate risk-factor changes:
        X. <- rmvnorm(N, mean = mu.hat, sigma = Sigma.hat)
        # compute corresponding (simulated) losses:
        L <- loss_operator(X., weights = w.)
        ## Compute VaR and ES and return
        list(VaR = VaR_np(L, alpha), # nonparametrically estimate VaR
             ES  =  ES_np(L, alpha), # nonparametrically estimate ES
             ## Additional quantities returned here
             mu    = mu.hat, # fitted mean vector
             Sigma = Sigma.hat) # fitted covariance matrix
    },
    "MC.t" = { # Monte Carlo based on a fitted multivariate t
        stopifnot(hasArg(N))
        N <- list(...)$N
        fit <- fit.mst(X, method = "BFGS") # fit a multivariate t distribution
        # simulate risk-factor changes:
        X. <- rmvt(N, sigma = as.matrix(fit$Sigma), df = fit$df, delta = fit$mu)
        # compute corresponding (simulated) losses
        L <- loss_operator(X., weights = w.)
        ## Compute VaR and ES and return
        list(VaR = VaR_np(L, alpha), # nonparametrically estimate VaR
             ES  =  ES_np(L, alpha), # nonparametrically estimate ES
             ## Additional quantities returned here
             mu    = fit$mu, # fitted location vector
             sigma = fit$Sigma, # fitted dispersion matrix
             Sigma = fit$covariance, # fitted covariance matrix
             df    = fit$df) # fitted degrees of freedom
    },
    "POT" = {
        stopifnot(hasArg(q))
        L. <- loss_operator(X, weights = w.) # historical losses
        # determine the threshold as the q-quantile of the historical losses
        u <- quantile(L., probs = list(...)$q, names = FALSE)
        excess <- L.[L. > u] - u
        fit <- fit_GPD_MLE(excess) # fit a GPD to the excesses
        xi <- fit$par[["shape"]] # fitted xi
        beta <- fit$par[["scale"]] # fitted beta
        if(xi <= 0) stop("Risk measures only implemented for xi > 0.")
        Fbu <- length(excess) / length(L.)
        VaR <- u + (beta/xi)*(((1-alpha)/Fbu)^(-xi)-1)
        ES <- (VaR + beta-xi*u) / (1-xi)
        if(xi >= 1) ES <- Inf
        list(VaR = VaR, # parametrically estimate VaR
```

```
                    ES  = ES, # parametrically estimate ES
                    ## Additional quantities returned here
                    xi     = xi, # fitted xi
                    beta   = beta, # fitted beta
                    converged = fit$converged,
                    u      = u, # threshold
                    excess = excess) # excesses over u
            },
            stop("Wrong 'method'"))
}
```

```
### Determine lambda and alpha values
# number of shares of the two stocks: 10 share of Tesla, 1 share of Gold
lambda <- c(10, 1)
alpha <- 0.99 # confidence levels for computing the risk measures
N <- 1e4 # Monte Carlo sample size

S. <- as.numeric(tail(S, n = 1)) # pick out last available stock prices ("today")
w. <- lambda * S. # weights tilde{w}
L <- loss_operator(X, weights = w.) # historical losses
```

```
## Variance-Covariance Method:
## Estimate a multivariate normal distribution
mu.hat <- colMeans(X) # estimate the mean vector mu
Sigma.hat  <- var(X) # estimate the covariance matrix Sigma
Sigma.hat = as.matrix(Sigma.hat) # convert into matrix form
# mean of the approx. normal df of the linearized loss L^{Delta}:
L.delta.mean <- -sum(w. * mu.hat)
L.delta.sd <- sqrt(t(w.) %*% Sigma.hat %*% w.) # corresponding standard deviation

## Compute VaR and ES and return
qa <- qnorm(alpha)
l = list(VaR = L.delta.mean + L.delta.sd * qa,
     ES  = L.delta.mean + L.delta.sd * dnorm(qa) / (1-alpha))
```

```
## The R code is used from "R_Standard_methods_for_market_risk"
## by Professor Erik Hintz
# historical simulation method
## Using nonparametrically estimated risk measures
L <- loss_operator(X, weights = w.) # compute historical losses
## Nonparametrically estimate VaR and ES and return a list
list(VaR = VaR_np(L, alpha), ES  =  ES_np(L, alpha))
```

```r
set.seed(445)
MC.N <- risk_measure(S, lambda = lambda, alpha = alpha, method = "MC.N", N = N)
MC.t <- risk_measure(S, lambda = lambda, alpha = alpha, method = "MC.t", N = N)
```

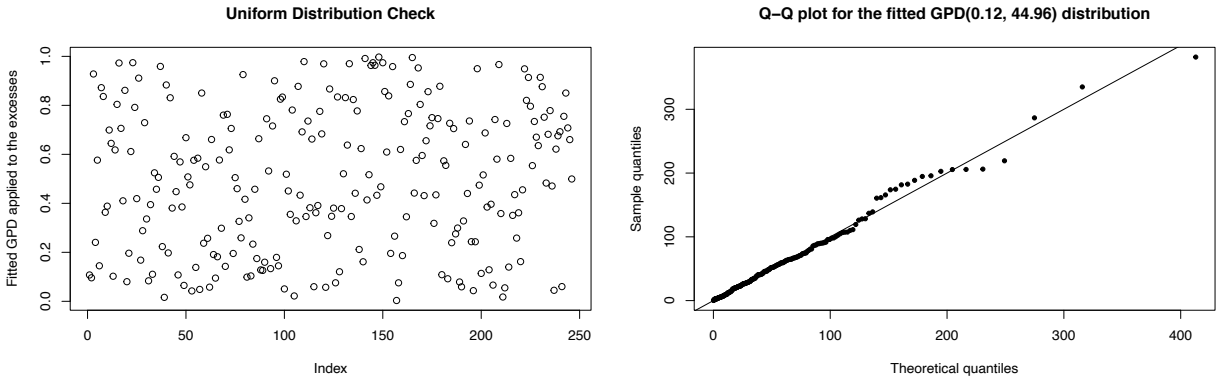
```r
POT <- risk_measure(S, lambda = lambda, alpha = alpha, method = "POT",
                    N = N, q = 0.9)
```

```r
## The R code is used from "R_Standard_methods_for_market_risk"
## by Professor Erik Hintz
### 3 Compute VaR and ES for the standard methods ###############################
lambda <- c(10, 1) # number of shares of the two stocks: 10 share of Tesla,
##1 share of Gold
alpha <- 0.99 # confidence levels for computing the risk measures
N <- 1e4 # Monte Carlo sample size

## Estimate VaR and ES with the various methods
set.seed(445)
var.cov <- risk_measure(S, lambda = lambda, alpha = alpha, method = "Var.Cov")
hist.sim <- risk_measure(S, lambda = lambda, alpha = alpha, method = "hist.sim")
MC.N <- risk_measure(S, lambda = lambda, alpha = alpha, method = "MC.N", N = N)
MC.t <- risk_measure(S, lambda = lambda, alpha = alpha, method = "MC.t", N = N)
POT <- risk_measure(S, lambda = lambda, alpha = alpha, method = "POT",  N = N,
                    q = 0.9)

## Pick out VaR and ES for all methods
rm <- rbind("MC (normal)"    = unlist(MC.N[c("VaR", "ES")]),
            "Var.-cov."      = unlist(var.cov),
            "Hist. sim."     = unlist(hist.sim),
            "MC (Student t)" = unlist(MC.t[c("VaR", "ES")]),
            "POT"            = unlist(POT [c("VaR", "ES")]))

# Make a table of VaR and ES
kable(rm,
      caption = "VaR and ES for all Methods at alpha = 0.99",
      valign = 't')%>%kable_styling(font_size = 12)%>%
  kable_styling(latex_options = "HOLD_position", full_width = F,
                bootstrap_options = c("striped", "hover", "condensed",
                                      "responsive"))
```

Table 9: VaR and ES for all Methods at alpha = 0.99

	VaR	ES
MC (normal)	160.3882	180.7825
Var.-cov.	167.0754	191.9310
Hist. sim.	183.5529	263.9727
MC (Student t)	184.8350	264.7508
POT	191.5630	258.8096

```
## The R code is used from "R_Standard_methods_for_market_risk"
## by Professor Erik Hintz
### Graphical goodness-of-fit check for the fitted GPD (=> Chapter 5)
par(mfrow = c(1,2))
excess <- POT$excess # excesses over the threshold
xi.hat <- POT$xi # estimated xi
beta.hat <- POT$beta # estimated beta
z <- pGPD(excess, shape = xi.hat, scale = beta.hat) # should be U[0,1]
plot(z, ylab = "Fitted GPD applied to the excesses",
     main = "Uniform Distribution Check")

qq_plot(excess, FUN = function(p) qGPD(p, shape = xi.hat, scale = beta.hat),
        main = paste0("Q-Q plot for the fitted GPD(", round(xi.hat, 2),", ",
                      round(beta.hat, 2),") distribution")) # looks fine
```



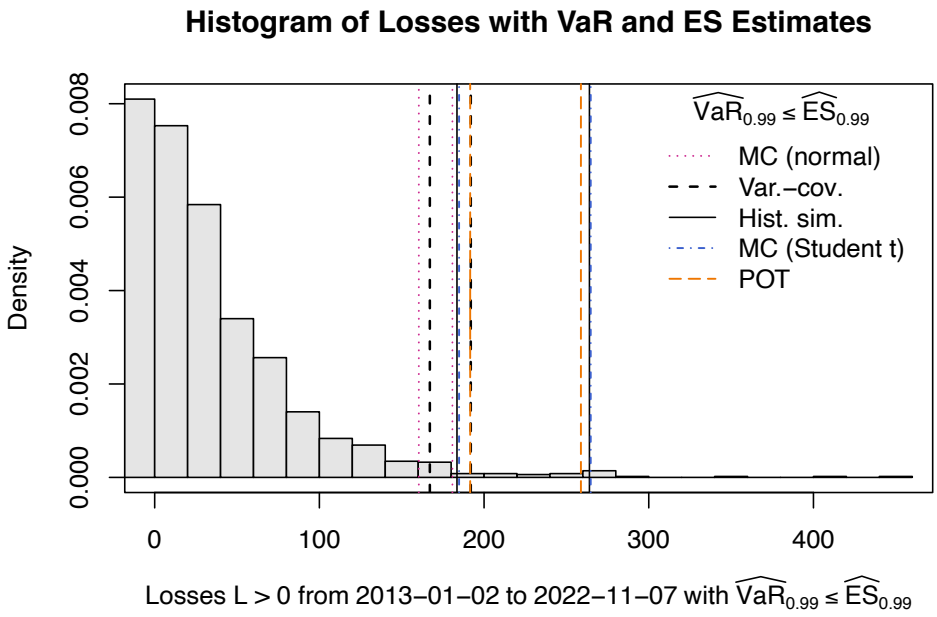

```
## The R code is used from "R_Standard_methods_for_market_risk"
## by Professor Erik Hintz
### 4 Graphical analysis ###################################################

## Compute historical losses (for creating a histogram); see risk.measure()
S. <- as.numeric(tail(S, n = 1)) # pick out last available stock prices ("today")
w. <- lambda * S. # weights tilde{w}
```

```r
L <- loss_operator(X, weights = w.) # historical losses
histo.loss = summary(L) # get important statistics about the losses
histo.loss = as.list.data.frame(histo.loss)

# Make a table of the summary statistics of historical losses
t(histo.loss) %>% kable(caption = "Summary Statistics of Historical
                        Losses", valign = 't') %>%
  kable_styling(latex_options = "HOLD_position", full_width = F)
```

Table 10: Summary Statistics of Historical Losses

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-524.7913	-39.21357	-3.635066	-4.892362	30.75479	454.2253

```r
time <- c("2013-01-02", "2022-11-07")
## Plot a histogram of the losses including the VaR and ES estimates
hist(L, breaks = "Scott", probability = TRUE, xlim = c(0, max(L, rm)),
     main = "Histogram of Losses with VaR and ES Estimates",
     xlab = substitute("Losses L > 0 from"~sd~"to"~ed~"with"~
                                          widehat(VaR)[a] <= widehat(ES)[a],
                       list(sd = time[1], ed = time[2], a = alpha)), col = "gray90")

box() # box around histogram
lty <- c(3, 2, 1, 4, 5)
lwd <- c(1.2, 1.6, 1, 1.2, 1.2)
cols <- c("maroon3", "black", "black", "royalblue3", "darkorange2")
for(k in seq_len(nrow(rm))) abline(v = rm[k,], lty = lty[k], lwd = lwd[k],
                                   col = cols[k])
legend("topright", bty = "n", inset = 0.02, lty = lty, lwd = lwd,
       col = cols, legend = rownames(rm),
       title = as.expression(substitute(widehat(VaR)[a] <= widehat(ES)[a],
                                        list(a = alpha)))) # legend
```


Histogram of Losses with VaR and ES Estimates
Density
0.000
0.002
0.004
0.006
0.008
0
100
200
300
400
$\widehat{\mathrm{VaR}}_{0.99} \le \widehat{\mathrm{ES}}_{0.99}$
MC (normal)
Var.–cov.
Hist. sim.
MC (Student t)
POT
Losses L > 0 from 2013–01–02 to 2022–11–07 with $\widehat{\mathrm{VaR}}_{0.99} \le \widehat{\mathrm{ES}}_{0.99}$