# Homework

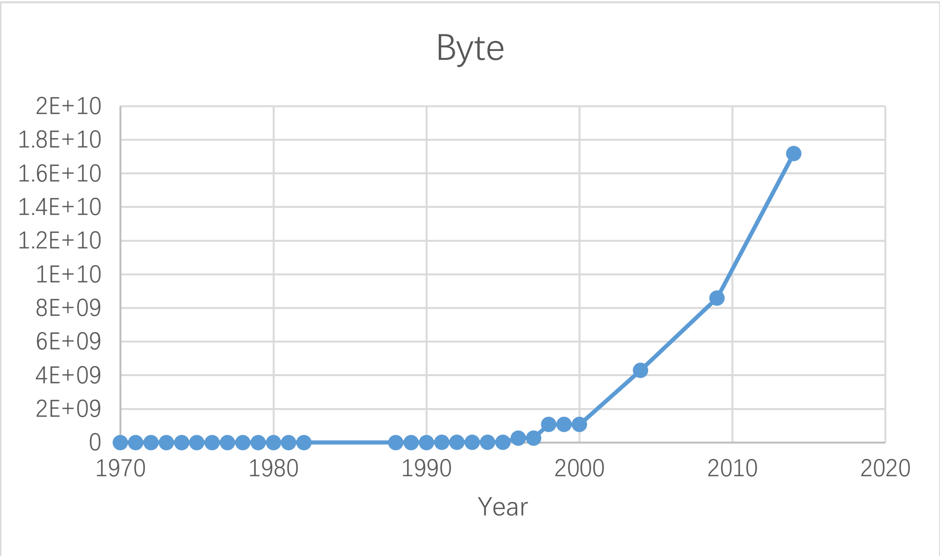
JIA ZHU（朱佳）, 27720151153580, WISE

# Homework 1

## Question 1

### The development of compute memory

| Year | Byte | Year | Byte |
|---|---|---|---|
| 1970 | 262144 | 1990 | 2097152 |
| 1971 | 262144 | 1991 | 16777216 |
| 1972 | 262144 | 1992 | 16777216 |
| 1973 | 262144 | 1993 | 16777216 |
| 1974 | 262144 | 1994 | 16777216 |
| 1975 | 262144 | 1995 | 16777216 |
| 1976 | 262144 | 1996 | 268435456 |
| 1977 | 262144 | 1997 | 268435456 |
| 1978 | 262144 | 1998 | 1073741824 |
| 1979 | 262144 | 1999 | 1073741824 |
| 1980 | 262144 | 2000 | 1073741824 |
| 1981 | 262144 | 2004 | 4294967296 |
| 1982 | 262144 | 2009 | 8589934592 |
| 1988 | 2097152 | 2014 | 17179869184 |
| 1989 | 2097152 | | |



## Question 2

**Brief explanation of Logistic regression**

The linear regression model assumes that the response Y is quantitative. But in many situations, the response variable is qualitative. For example, eye color is qualitative, taking on values blue, brown, or green. It is unreasonable to assume that the dependent variable Y is normal distribution. Therefore, the outcome variable may be categorical such as binary variables(for example, yes/no, passed/failed, lived/died) and polytomous variables(for example, poor/good/excellent).

Logistic regression is applied to situations in which the response variable Y is binary (0,1). We use a linear regression model to represent the logistic or logit function of p:

$$\log it(p) = \log \frac{p}{1-p} = \beta^T X \ , \quad (1)$$

where $p = P(Y=1|X)$. And we can know that $p = \mu_Y$ is the conditional mean of Y. In this case, $\log[p/(1-p)]$ is the link function.

Equation (1) can be rewritten as

$$p(X) = \frac{\exp(\beta^T X)}{1+\exp(\beta^T X)} \ .$$

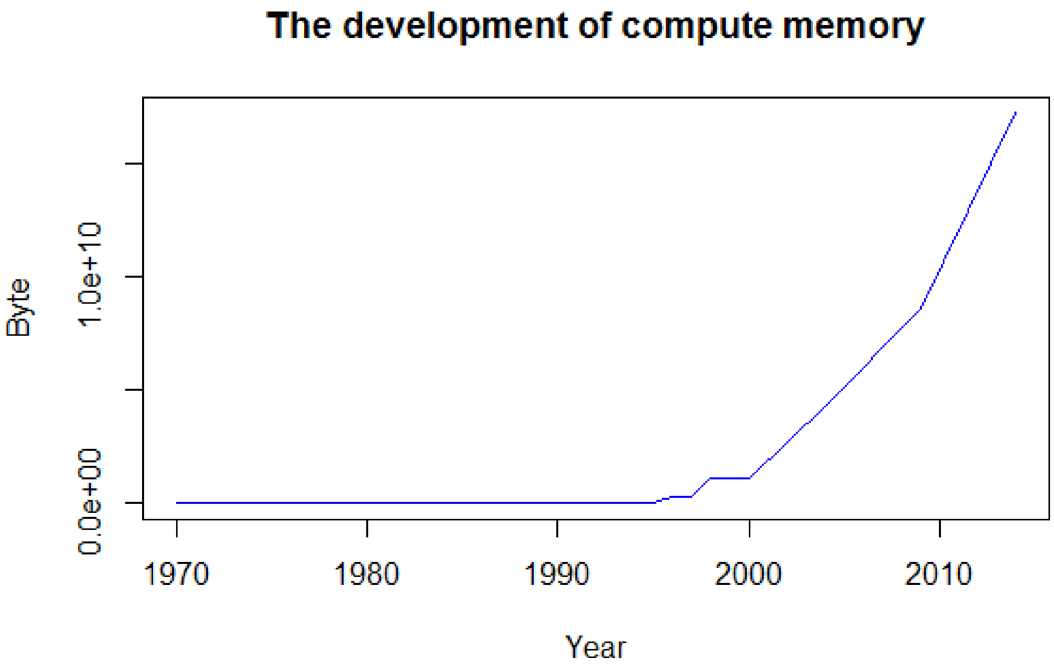
The logit function can take any real value, but the associated probability always lies in the required [0,1] interval. The parameters of the logistic regression model are estimated by maximum likelihood.

# Homework 2

## Question 1

**R Codes:**

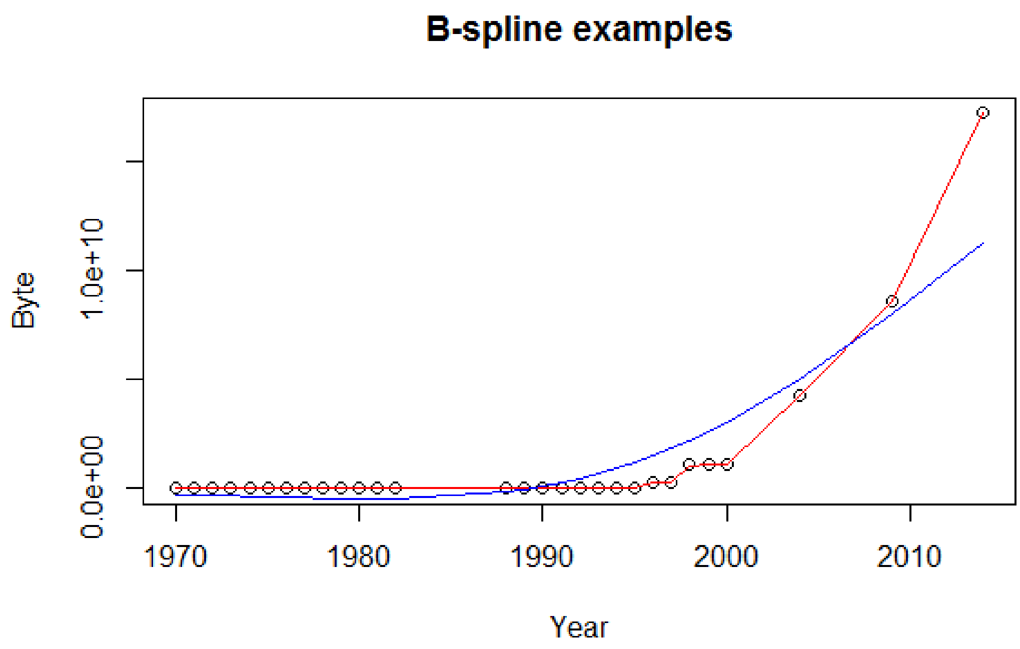
```
## question 1 ##
Byte <- read.csv("byte.csv",header = TRUE)
plot(Byte, type="l", col="blue", main="The development of compute memory")
```




## Question 2

**R Codes:**

```
## question 2 ##
plot(Byte, type="p", main="B-spline examples")
lines(smooth.spline(Byte, spar = 0.1),type = "l", col="red")
lines(smooth.spline(Byte, spar = 0.9),type = "l", col="blue")
```




## Question 3

Suppose we observe that in n=1000 mails (in 1 week) we have about 2 scams. In order to calculate that we have 6 scam emails in 2 weeks, we should calculate that we have 3 scam emails in 1 week. Therefore,

$$\lambda = 2, \quad P(X = 3) = \frac{2^3}{3!}\exp(-2) = 0.18 .$$

In Scammyland you have 5 scams on average, what is the probability to have no scam mail:

$$\lambda = 5, \quad P(X = 0) = \frac{5^0}{0!}\exp(-5) = 0.0067 .$$

# Homework 3

## Question 1

**R Codes:**

```
## question 1 ##
# install stuff for hash calculation
install.packages("digest")
# call the library doing the hashes
library("digest")
digest("I learn a lot from this class when I am proper listening to the professor")
# the result is "a8d3e4701672195e5dcd16ea9b062279"
digest("I do not learn a lot from this class when I am absent and playing on my phone")
# the result is "059ab10d478614d2eab3d70cfccd3fcc"

digest("I learn a lot from this class when I am proper listening to the professor","sha256")
# the result is "c16700de5a5c1961e279135f2be7dcf9c187cb6b21ac8032308c715e1ce9964c"
digest("I do not learn a lot from this class when I am absent and playing on my phone","sha256")
# the result is "f5e2cba48dac097355d0bb310fdbd5bd38a22a5c8e8215cd1ae67014cfc35b91"
```

## Question 2

The digital signature algorithm (DSA) is a federal information processing standard for digital signatures. In august 1991 the national institute of standards and technology (NIST) proposed DSA for use in their digital signature standard (DSS) and adopted it as FIPS 186 in 1993.

DSA consists of 2 parts: generation of a pair of public key and private key; generation and verification of digital signature.

Key generation has two phases: The first phase is a choice of algorithm parameters which may be shared between different users of the system, The second phase computes public and private keys for a single user.

Reference:

https://en.wikipedia.org/wiki/Digital_Signature_Algorithm

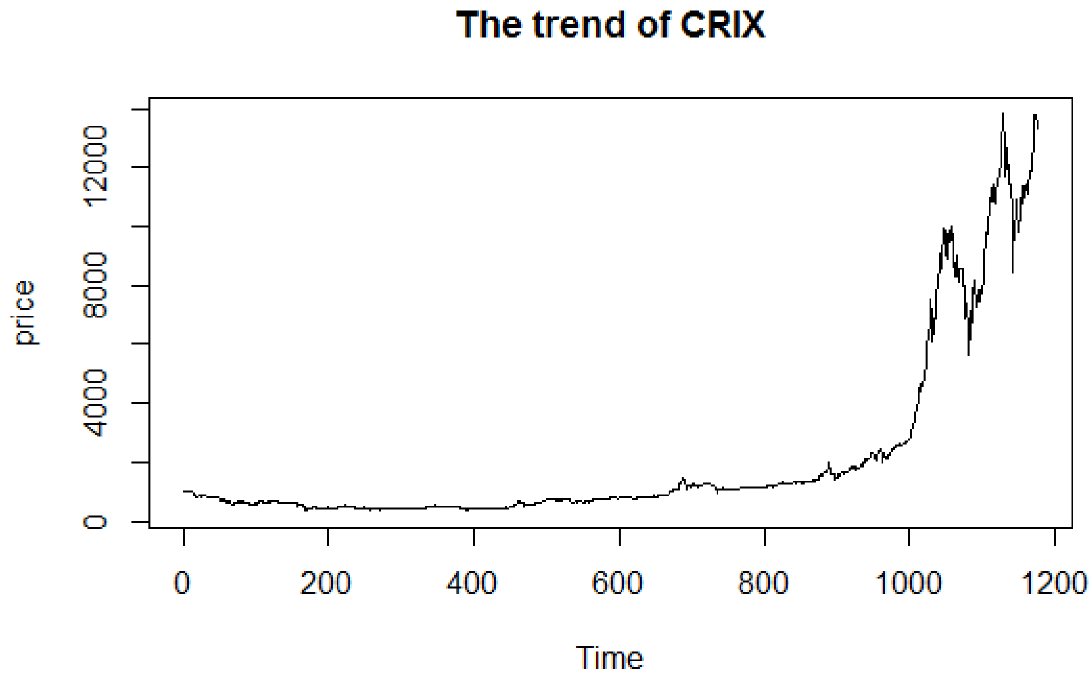
http://www.herongyang.com/Cryptography/DSA-Introduction-What-Is-DSA-Digital-Signature-Algorithm.html

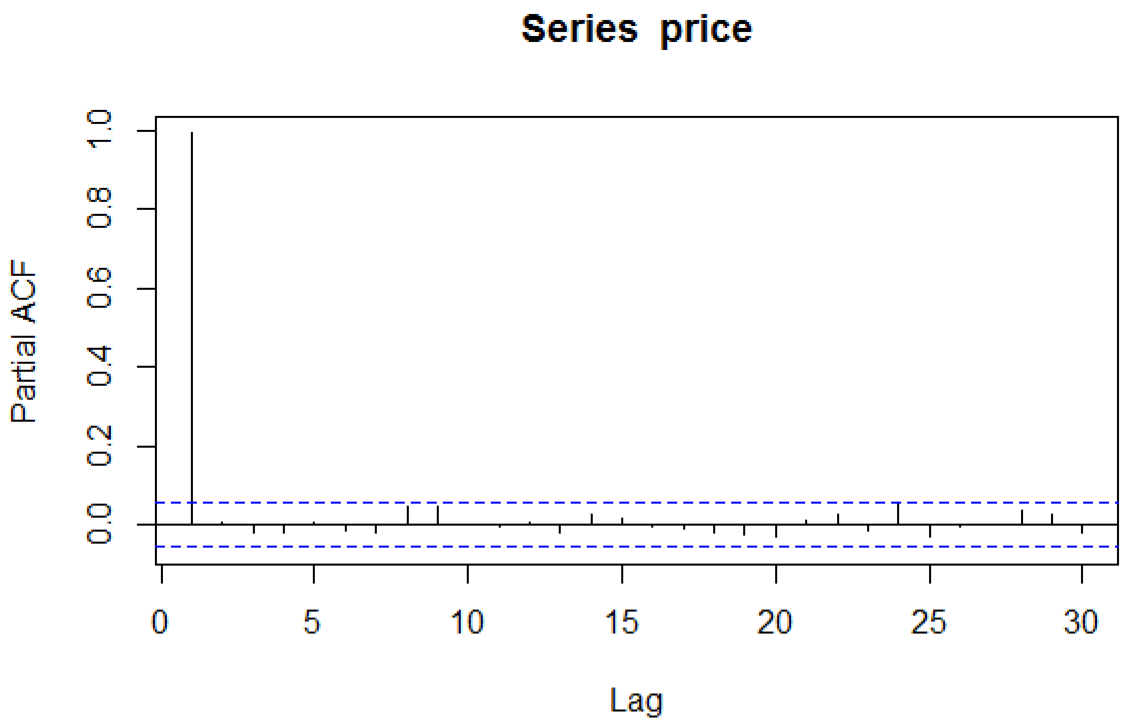
## Question 3

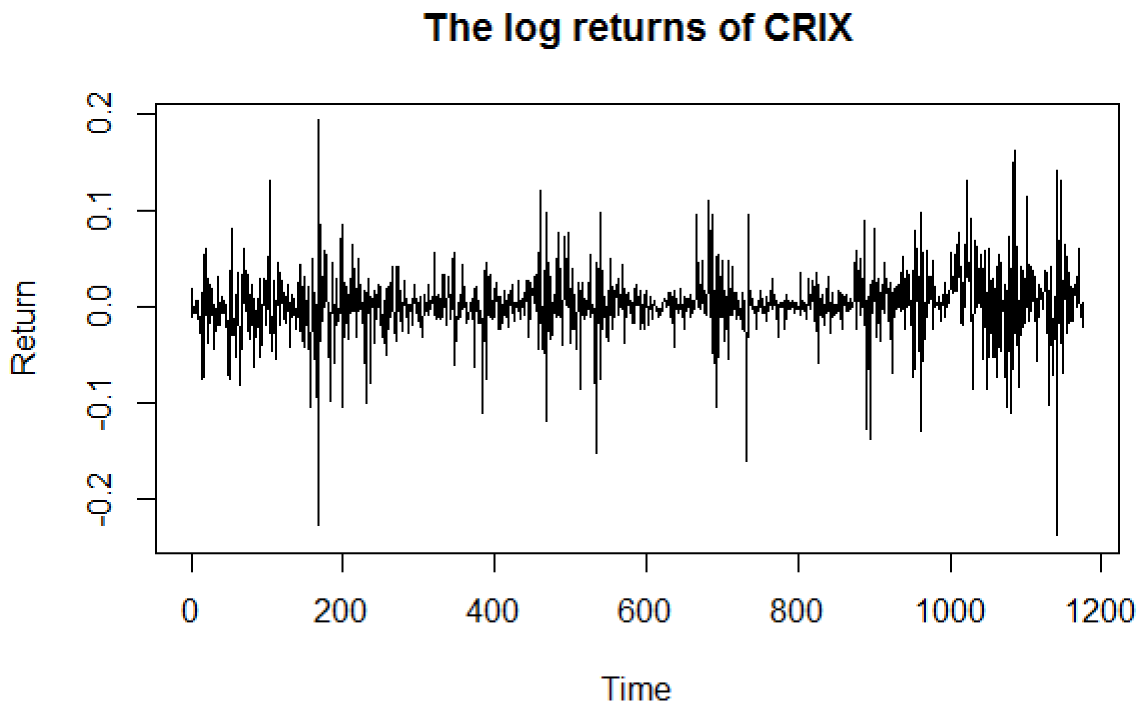
**R Codes:**

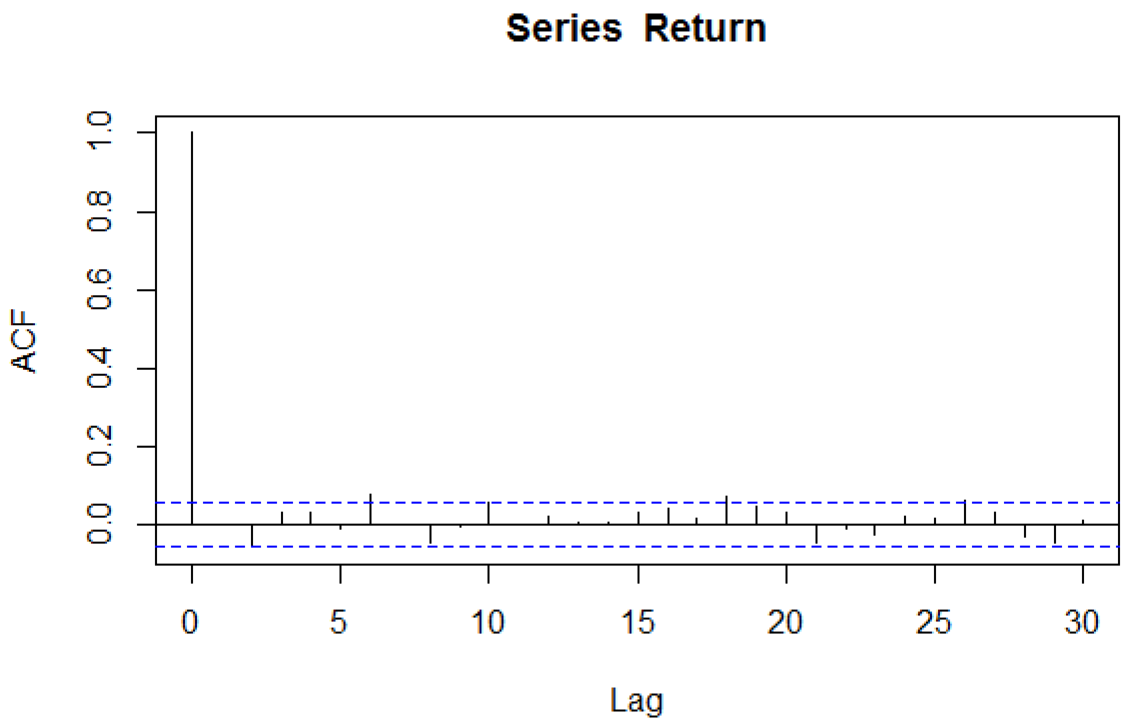
```
## question 3 ##
install.packages("rjson", repos="http://cran.us.r-project.org")
library("rjson")
json_file = "http://crix.hu-berlin.de/data/crix.json"
json_data = fromJSON(file=json_file)
crix_data_frame = as.data.frame(json_data)
a <- 1:(ncol(crix_data_frame)/2)
n <- 2*a
m <- n-1
time <- t(crix_data_frame[m])
price <- t(crix_data_frame[n])
mydata <- cbind(time,price)
ts.plot(price, main="The trend of CRIX")

install.packages("tseries")
library(tseries)
adf.test(price) #p-value = 0.99, accept H0: price is nonstationary
acf(price)
pacf(price)
install.packages("forecast")
library(forecast)
```

```
auto.arima(price)# ARIMA(5,2,0)
Return <- diff(log(price))
ts.plot(Return,main="The log returns of CRIX")
adf.test(Return)#p-value = 0.01, Return is stationary
acf(Return)
pacf(Return)
auto.arima(Return)# ARIMA(5,2,0)
```

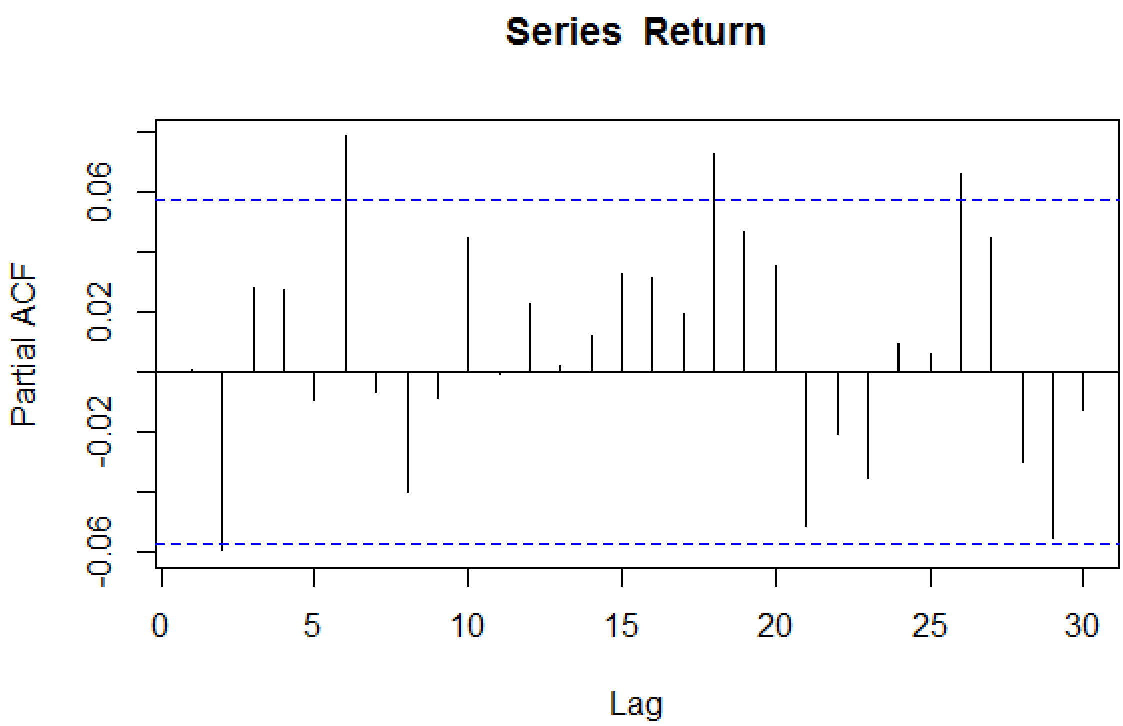




Series price
Partial ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30


The log returns of CRIX
Return
Time
-0.2
-0.1
0.0
0.1
0.2
0
200
400
600
800
1000
1200



Series Return
ACF
Lag

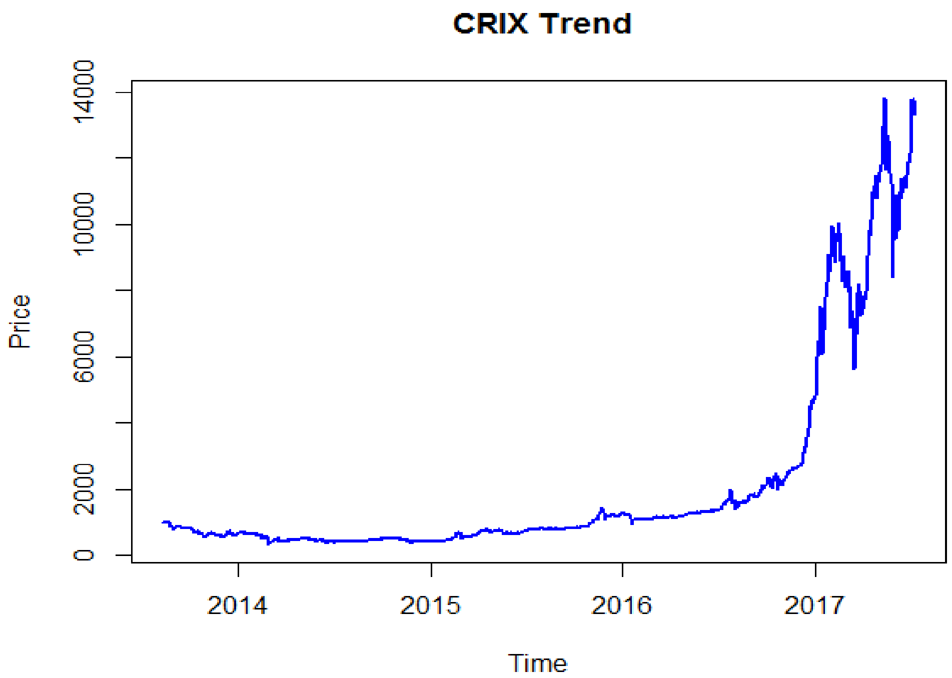
Series Return
Partial ACF
Lag

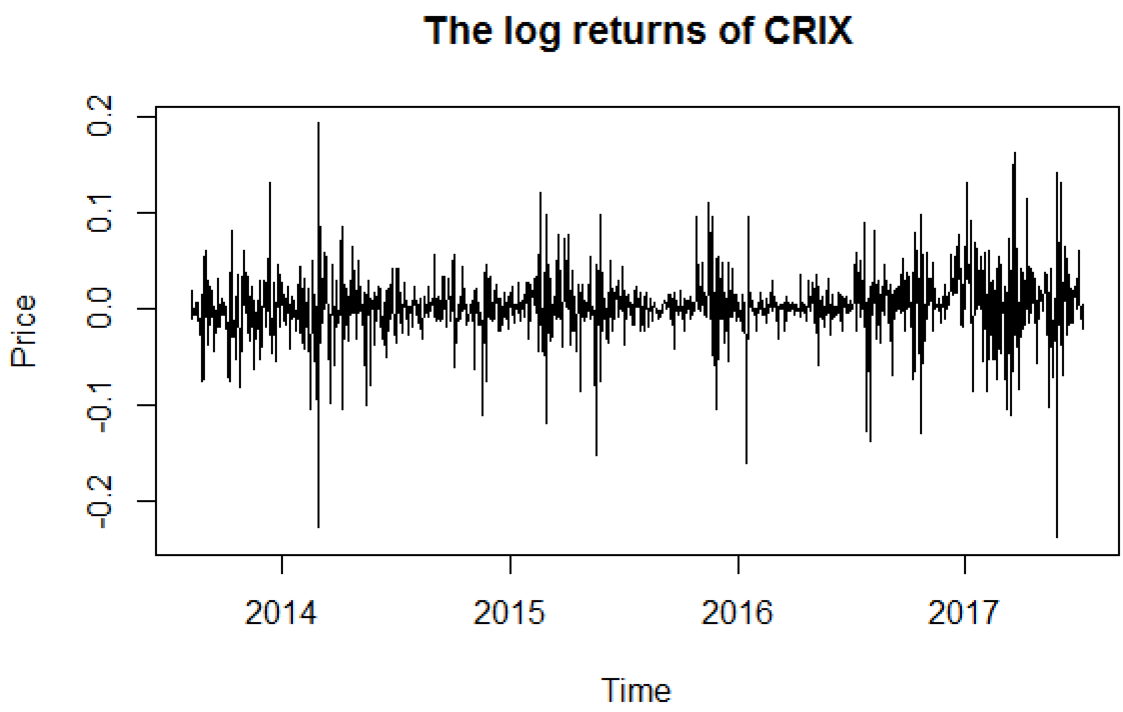
# Homework 4

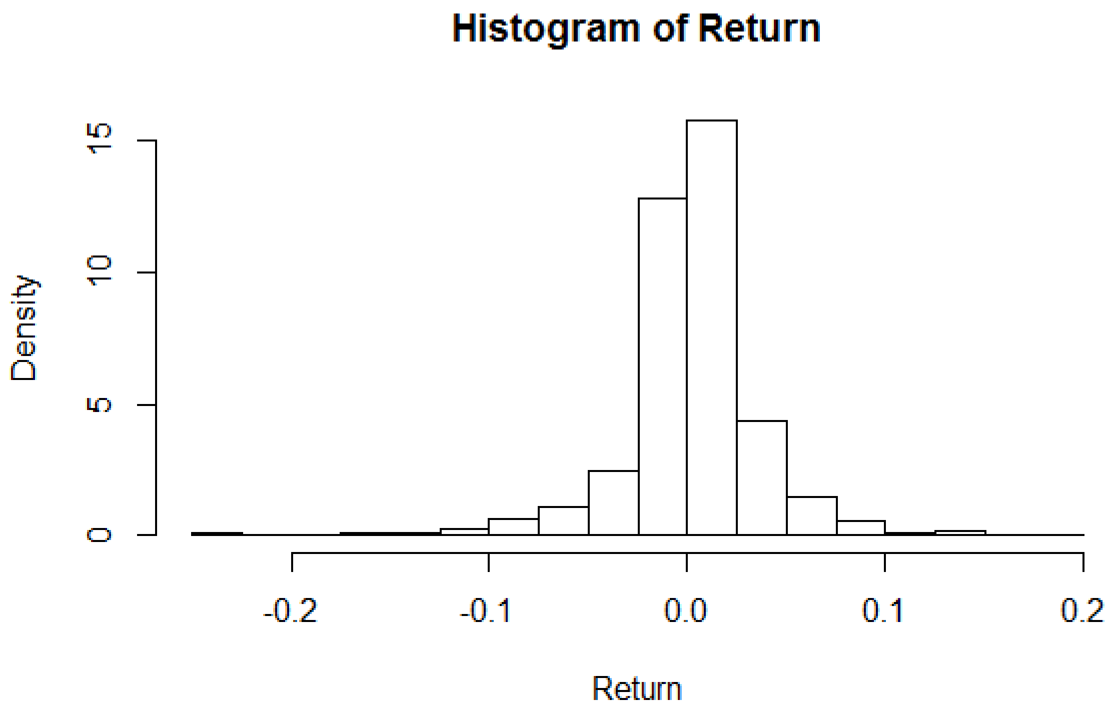
## Question 1

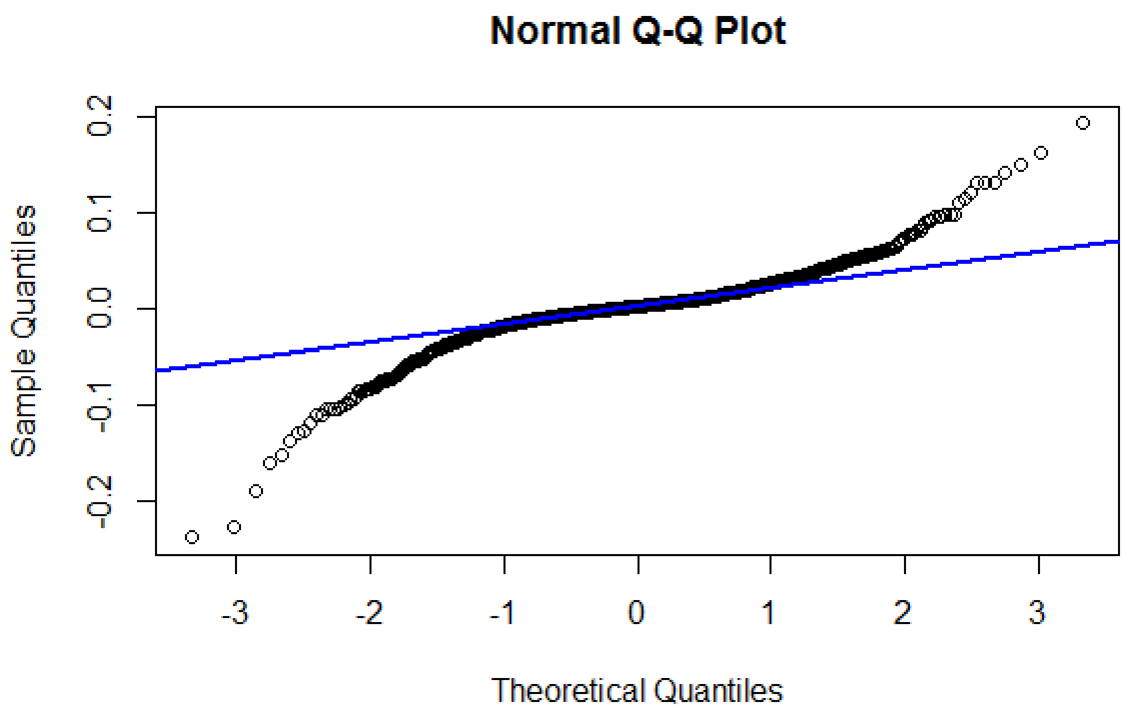
**R Codes:**

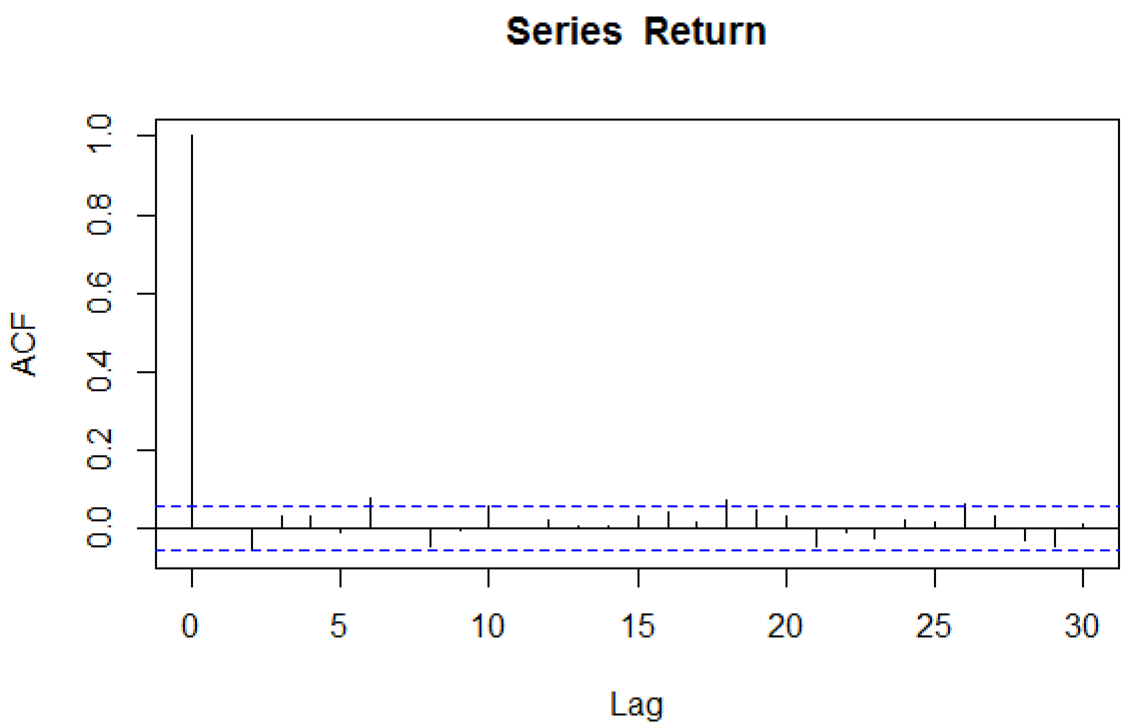
```
# figure 3 #
plot(price, type = "l", lwd = 2, col = "blue", xaxt='n',xlab="Time",ylab = "Price", main = "
CRIX Trend")
axis(1,c(120,420,720,1020),c("2014","2015","2016","2017"))
# figure 4 #
plot(Return, type = "l", xaxt='n',xlab="Time",ylab = "Price", main = "The log returns of CRIX")
axis(1,c(120,420,720,1020),c("2014","2015","2016","2017"))
# figure 5 #
hist(Return,freq=FALSE,breaks = seq(-0.25,0.2,by=0.025))
#lines(density(Return),col="blue",lwd=2)
qqnorm(Return)
qqline(Return,col="blue",lwd=2)# figure 6 #
# figure 6 #
acf(Return)
pacf(Return)
```

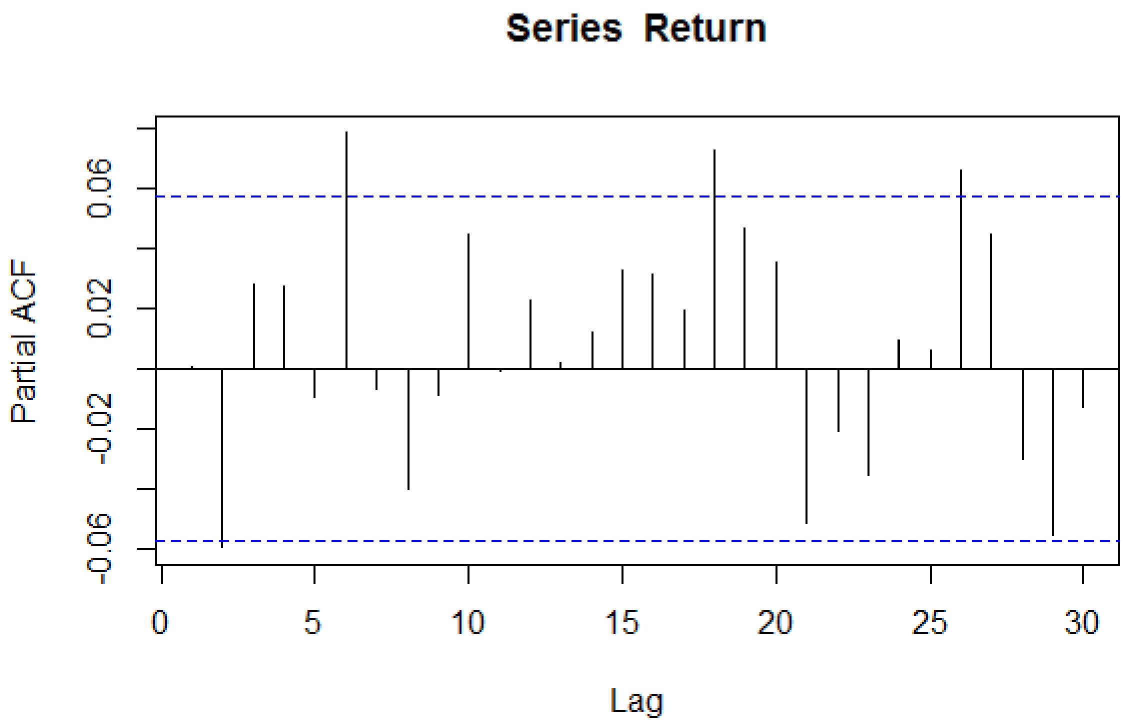



The log returns of CRIX
Price
0.2
0.1
0.0
-0.1
-0.2
2014
2015
2016
2017
Time


Histogram of Return
Density
15
10
5
0
-0.2
-0.1
0.0
0.1
0.2
Return


Normal Q-Q Plot
Sample Quantiles
Theoretical Quantiles
-0.2
-0.1
0.0
0.1
0.2
-3
-2
-1
0
1
2
3


Series Return
ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30

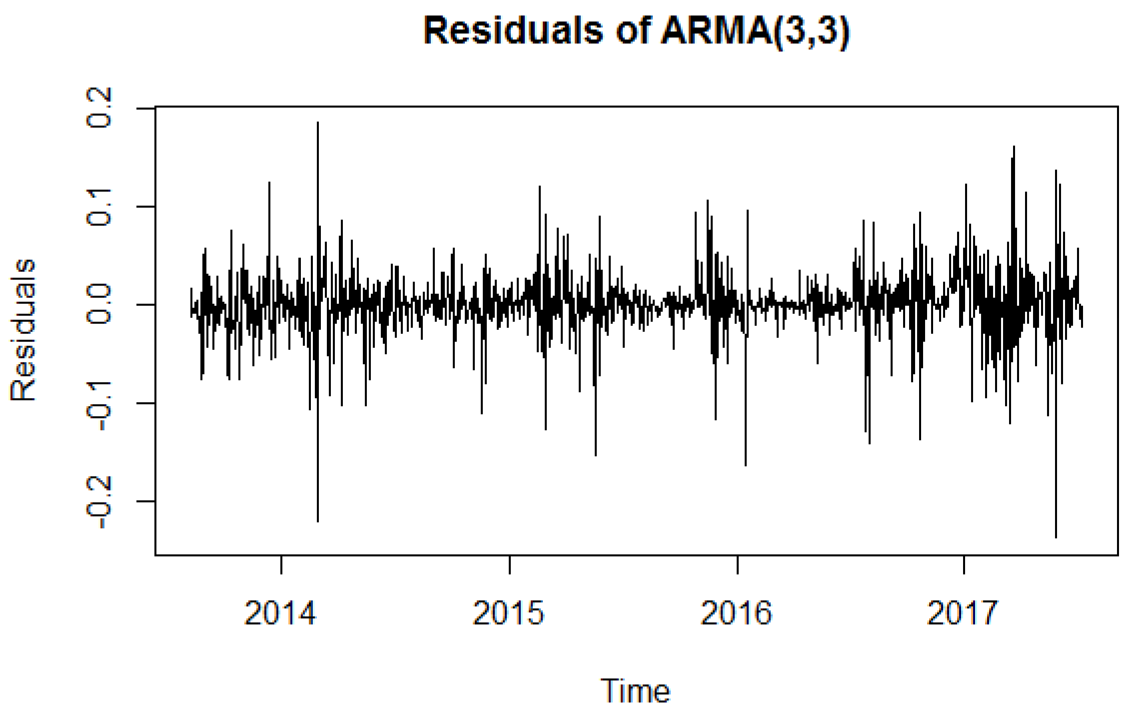


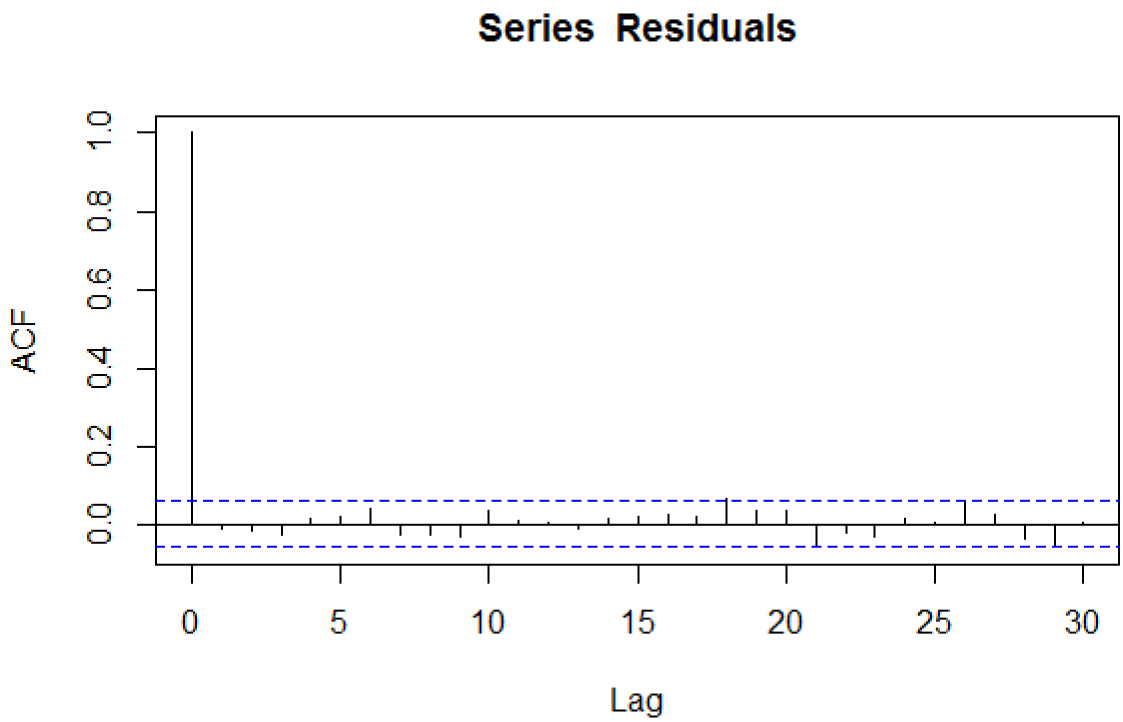

## Question 2

**R Codes:**

```
aic = matrix(NA, 6, 6)
for (p in 0:4) {
        for (q in 0:3) {
                a.p.q = arima(Return, order = c(p, 0, q))
                aic.p.q = a.p.q$aic
                aic[p + 1, q + 1] = aic.p.q
        }
}
aic

# bic
bic = matrix(NA, 6, 6)
for (p in 0:4) {
```

```
        for (q in 0:3) {
                b.p.q = arima(Return, order = c(p, 0, q))
                bic.p.q = AIC(b.p.q, k = log(length(ret)))
                bic[p + 1, q + 1] = bic.p.q
        }
}
bic

# select p and q order of ARIMA model
fit1 = arima(ret, order = c(3, 0, 3))
plot(fit1$residuals, xaxt='n',ylab="Residuals", main="Residuals of ARMA(3,3)")
axis(1,c(120,420,720,1020),c("2014","2015","2016","2017"))
Box.test(fit1$residuals, lag = 1)
Residuals <- fit1$residuals
acf(Residuals)
```




Series Residuals
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
Lag