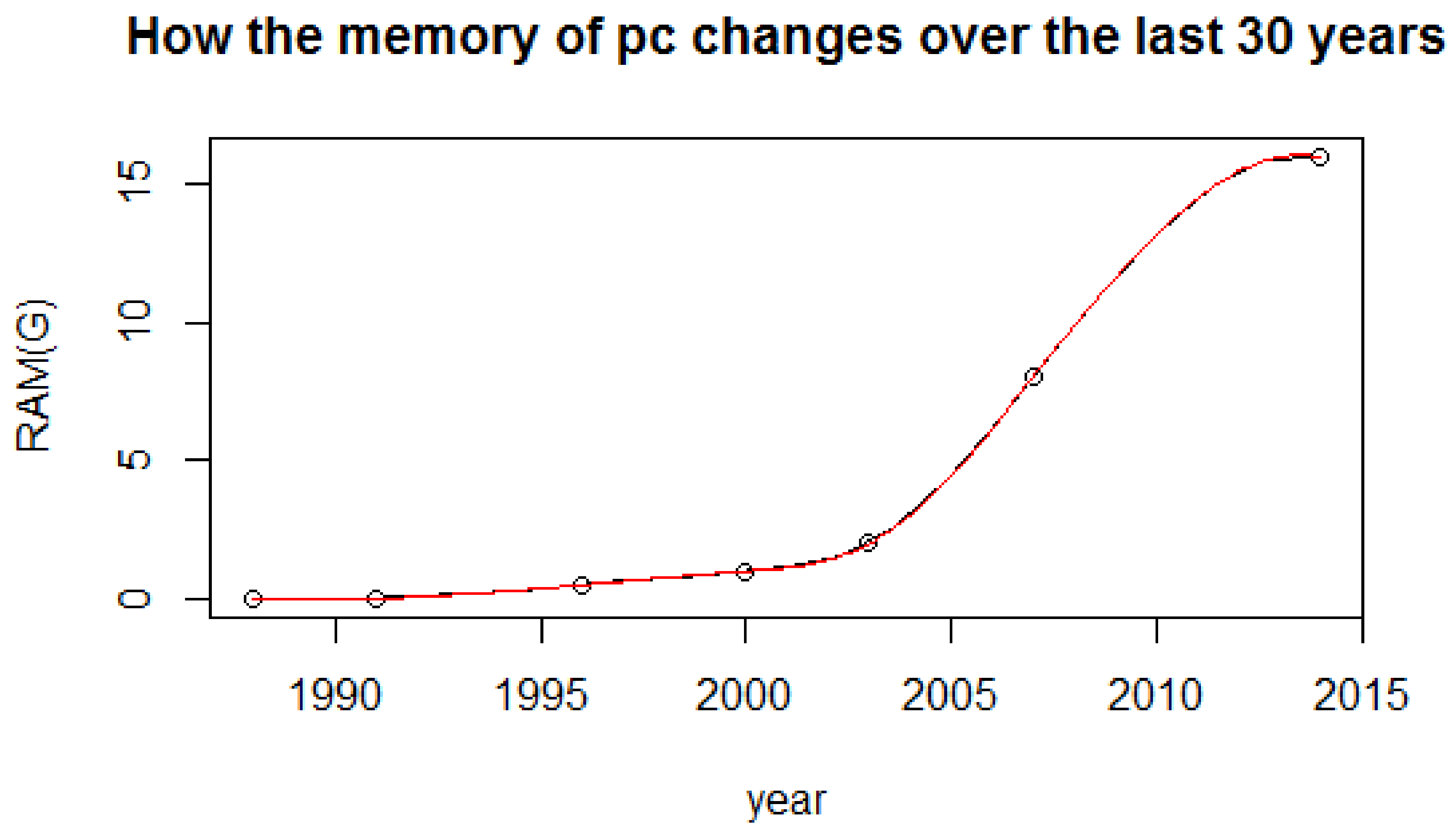
Homework 1

Lei Zang

The development path of memory of PC



Logistic Regression

Logistic Regression is a regression model where the dependent variable (DV) is categorical. This article covers the case of a binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed in multinomial logistic regression, or, if the multiple categories are ordered, in ordinal logistic regression.[2] In the terminology of economics, logistic regression is an example of a qualitative response/discrete choice model.

Homework 2

Lei Zang

Question 1

Code:

```
x = 6
n = 1000
lambda = 2
p = lambda / n
dbinom (x,2*n,p) # binomial probability mass function
dpois (x, 2*lambda ) # Poisson probability mass function
dpois (0, 5 )
```

Question 2

Code:

```
plot(year,number,type = "b",
    col="black",main = "The history of computer memory",
    sub = "This is the change in the number of types of computer memory",
    xlab = "year",ylab = "The number of memory")
barplot(number,
      xlab = "year",ylab ="The number of memory ")
```

Question 3

Code:

```
lambda=2
x=seq(0:6)
P<-data.frame(dpois(x,lambda))
sum<-(P[1,]+P[7,]+P[2,]+P[6,]+P[3,]+P[5,]+P[4,]+P[4,])
sum
lambda=5
x=0
dpois(x,lambda)
```

Homework 3

Lei Zang

Homework 3

```
library(digest)
digest("I learn a lot from this class when I am proper listening to  the professor","sha256")
```

```
## [1] "3f3461bb29b98d680b3b335fb9f801e6831c9e3842ca2710d716076593954567"
```

```
library(digest)
digest('I do not learn a lot from this class when I am absent and playing on my Iphone',"sha256")
```

```
## [1] "2533d529768409d1c09d50451d9125fdbaa6e5fd4efdeb45c04e3c68bcb3a63e"
```

Digital Signature Algorithm

Lei Zang

What is a digital signature

A digital signature is a mathematical technique used to validate the authenticity and integrity of a message, software or digital document.

Common digital signature algorithm

1. RSA-based signature schemes, such as RSA-PSS.
2. DSA and its elliptic curve variant ECDSA.
3. Edwards-curve Digital Signature Algorithm and its Ed25519 variant.
4. ElGamal signature scheme as the predecessor to DSA, and variants Schnorr signature and Pointcheval–Stern signature algorithm.

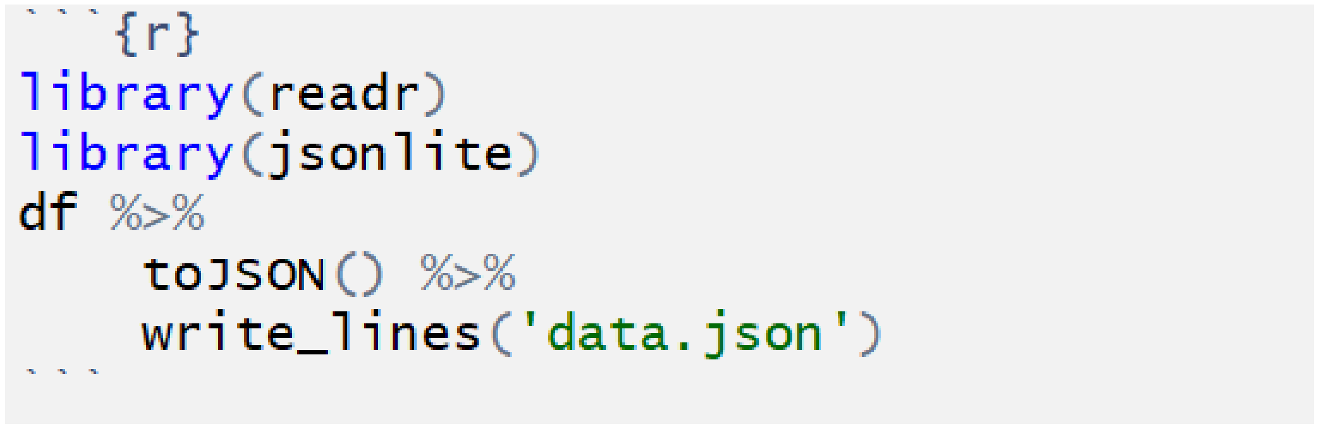
How does digital signature algorithm work

Digital signatures are based on public key encryption, also known as asymmetric encryption. Using a public key algorithm such as RSA, you can generate two keys that are mathematically linked: a private and a public key. To create a digital signature, the signature software (such as an e-mail program) creates a one-way hash of the electronic data to be signed. Then the private key is used to encrypt the hash. The encrypted hash and other information such as the hash algorithm are digital signatures. The reason for encrypting a hash rather than an entire message or document is that the hash function can convert any input to a fixed-length value, which is usually much shorter. This saves time because the hash is much faster than the signature.

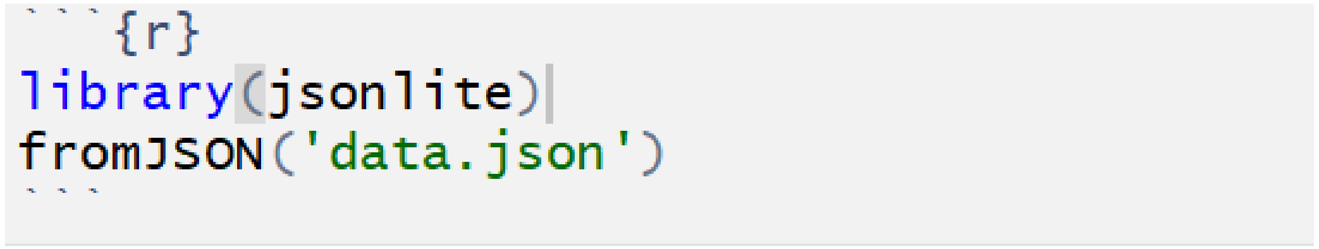
Save and read Jason format data in R

Lei Zang

This is how to save and read Jason format data in R



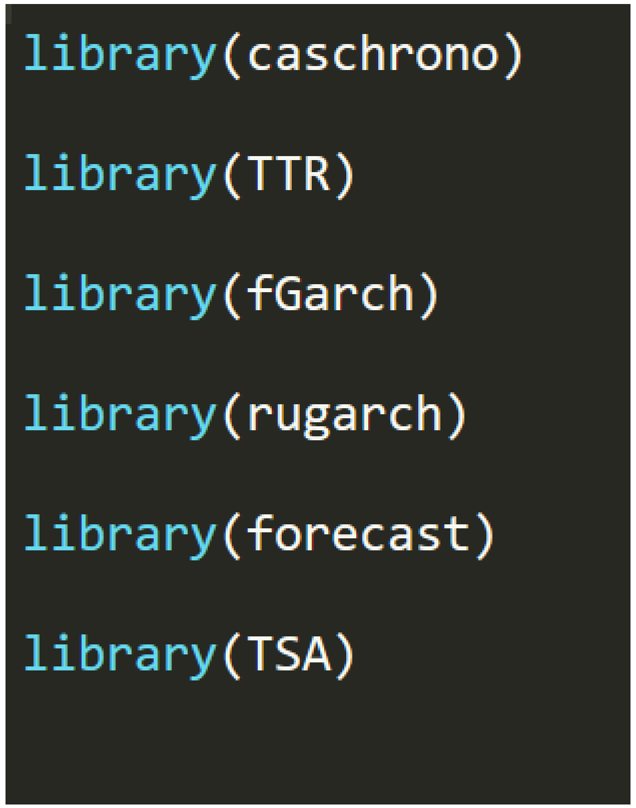
~~~
```{r}
library(readr)
library(jsonlite)
df %>%
    toJSON() %>%
    write_lines('data.json')
```
~~~



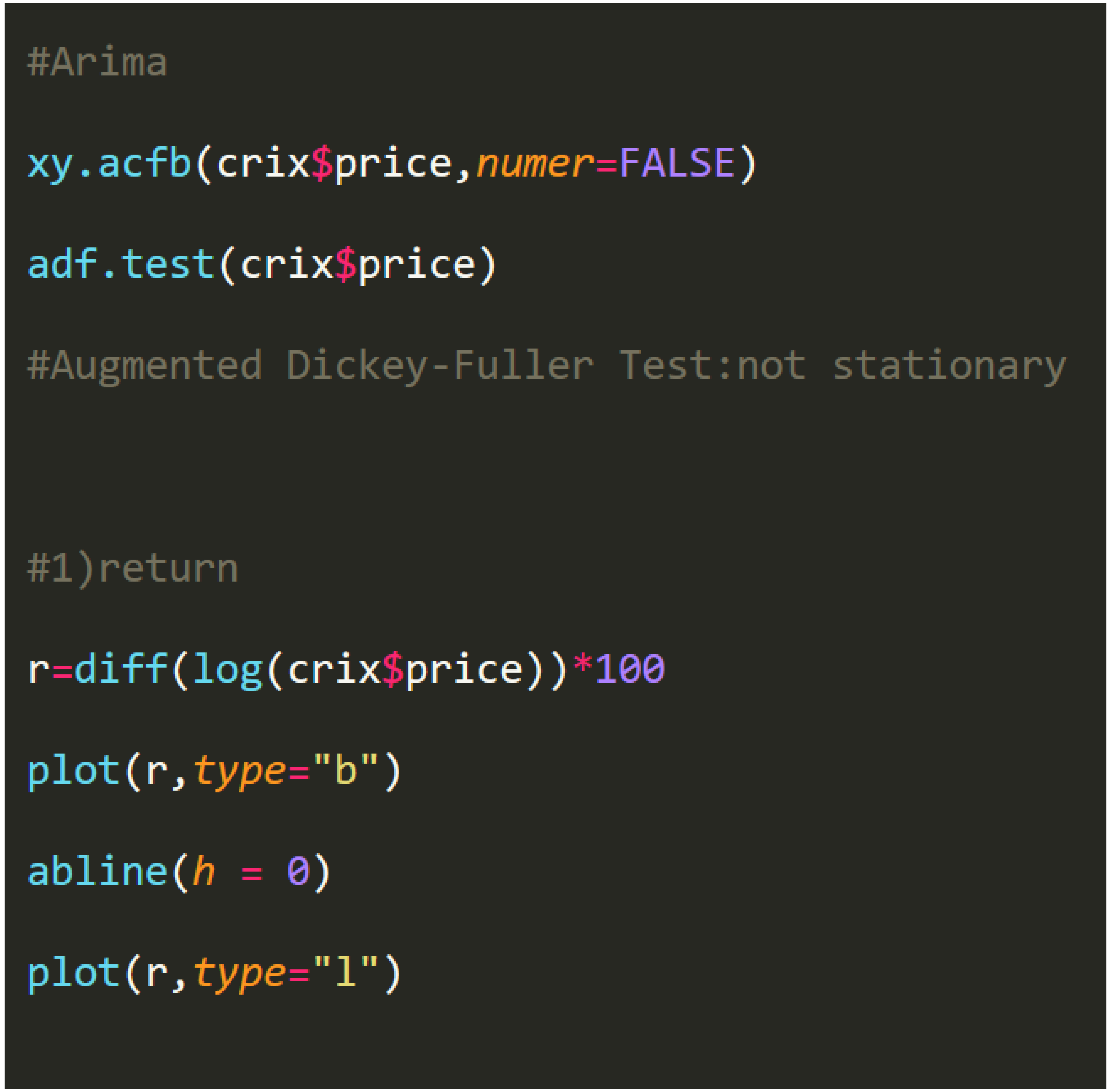
~~~
```{r}
library(jsonlite)
fromJSON('data.json')
```
~~~

Question 4

Lei Zang



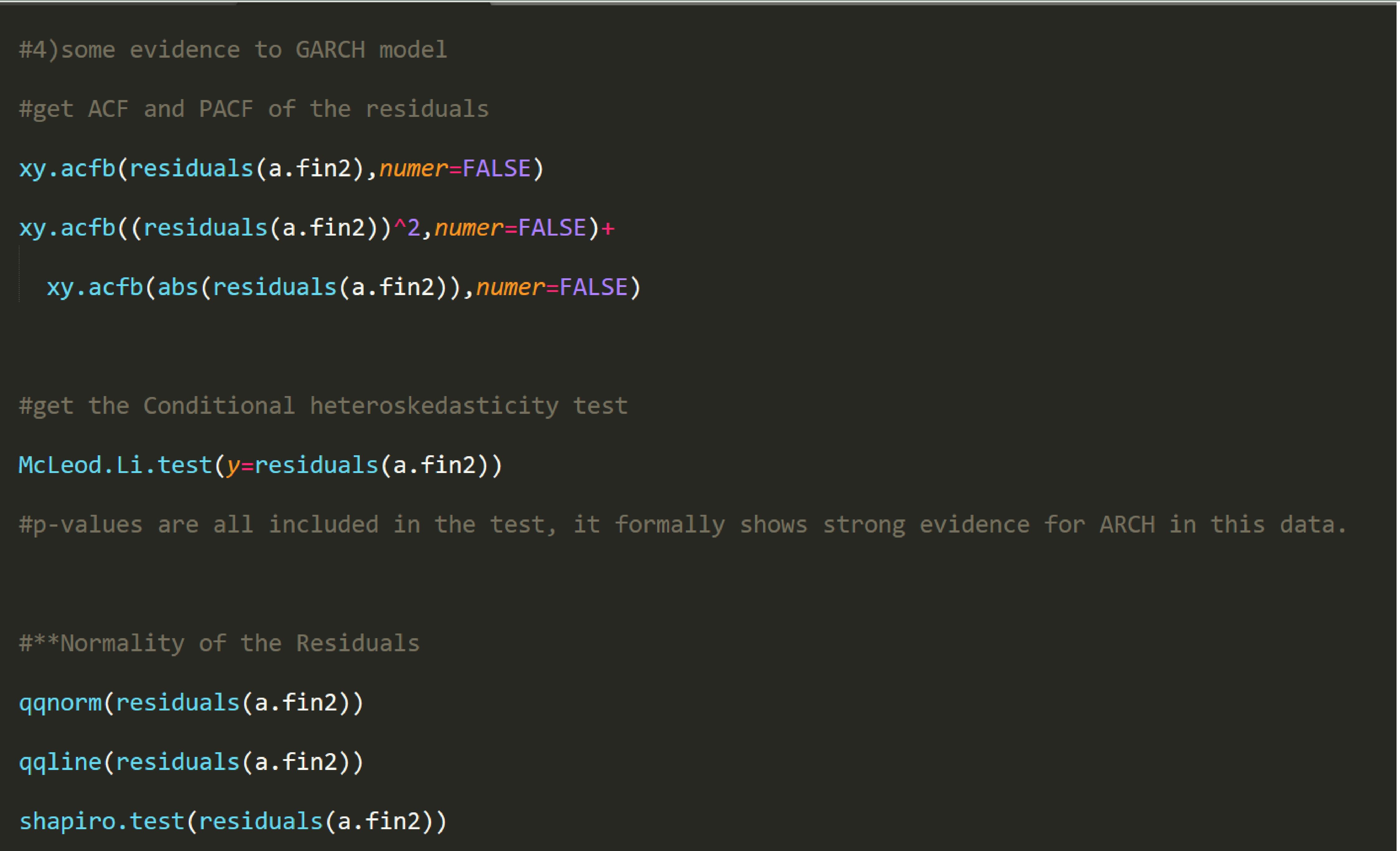
```
library(caschrono)

library(TTR)

library(fGarch)

library(rugarch)

library(forecast)

library(TSA)
```



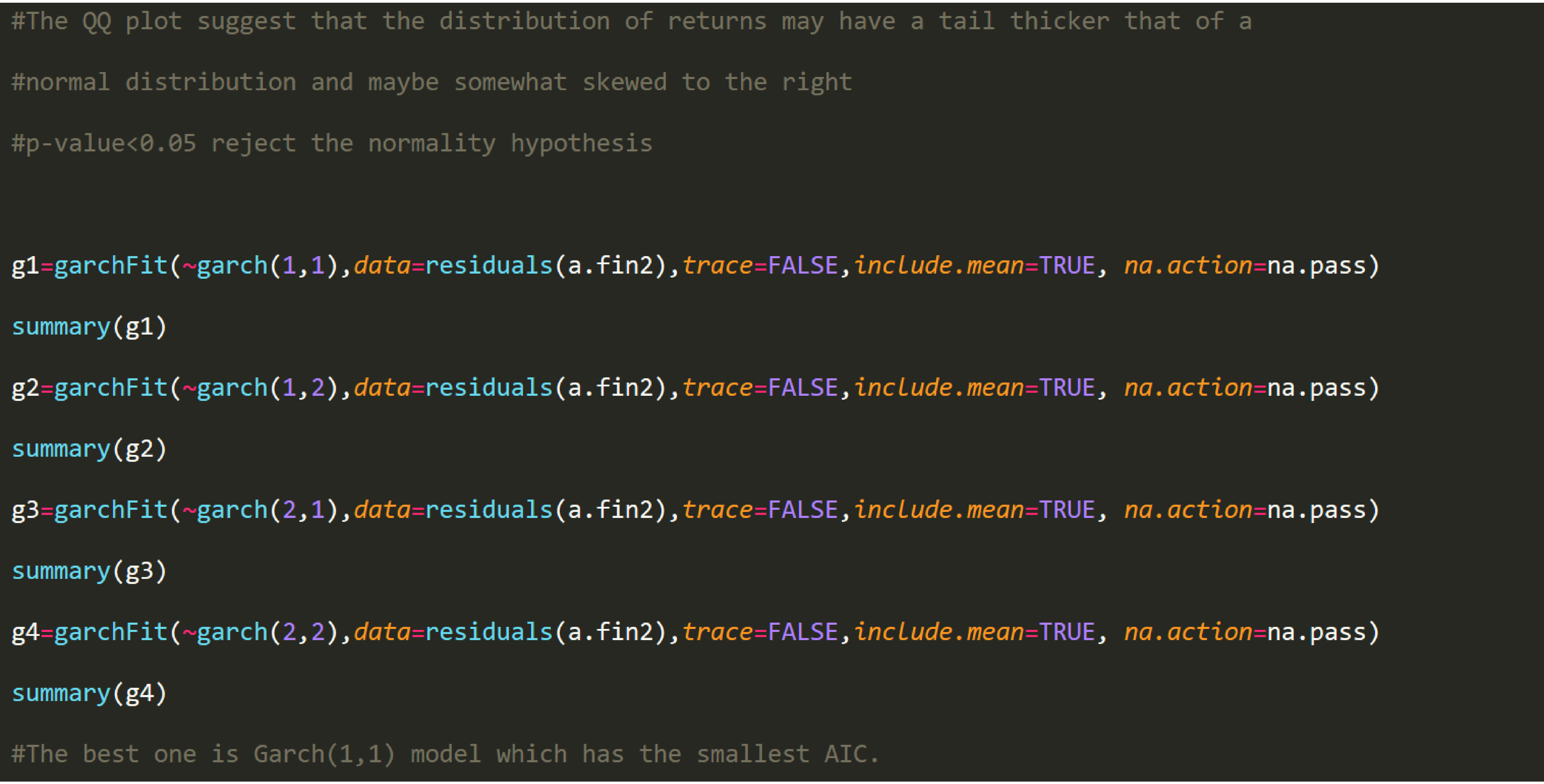
```r
#Arima

xy.acfb(crix$price,numer=FALSE)

adf.test(crix$price)

#Augmented Dickey-Fuller Test:not stationary



#1)return

r=diff(log(crix$price))*100

plot(r,type="b")

abline(h = 0)

plot(r,type="l")
```

```r
#2)Model Specification ARIMA(p,d,q)

#ADF test-H0:unit root H1:no unit root(test for stationarity)

adf.test(r)

#p-value=0.27,not stationary.

dr=diff(r)

plot(dr,type="b")

abline(h = 0)

adf.test(dr)

#p-value=0.01,stationary.(d=1)
```

```r
#3)Parameter Estimation

#estimation of p and q



a.fin1=auto.arima(dr)

summary(a.fin1)

#ARMA(0,0) therefore r fits ARIMA(0,1,0)

a.fin2=arima(r,order=c(0,1,0))

summary(a.fin2)

help("forecast.Arima")

f=forecast(a.fin2,h=3,level=c(99.5))

acf(f$residuals,lag.max = 20)

Box.test(f$residuals,lag=20,type='Ljung-Box')

#the residuals follow Gaussian distribution

plot.ts(f$residuals)
```



```r
#4)some evidence to GARCH model

#get ACF and PACF of the residuals

xy.acfb(residuals(a.fin2),numer=FALSE)

xy.acfb((residuals(a.fin2))^2,numer=FALSE)+

  xy.acfb(abs(residuals(a.fin2)),numer=FALSE)


#get the Conditional heteroskedasticity test

McLeod.Li.test(y=residuals(a.fin2))

#p-values are all included in the test, it formally shows strong evidence for ARCH in this data.


#**Normality of the Residuals

qqnorm(residuals(a.fin2))

qqline(residuals(a.fin2))

shapiro.test(residuals(a.fin2))
```



```
#The QQ plot suggest that the distribution of returns may have a tail thicker that of a

#normal distribution and maybe somewhat skewed to the right

#p-value<0.05 reject the normality hypothesis



g1=garchFit(~garch(1,1),data=residuals(a.fin2),trace=FALSE,include.mean=TRUE, na.action=na.pass)

summary(g1)

g2=garchFit(~garch(1,2),data=residuals(a.fin2),trace=FALSE,include.mean=TRUE, na.action=na.pass)

summary(g2)

g3=garchFit(~garch(2,1),data=residuals(a.fin2),trace=FALSE,include.mean=TRUE, na.action=na.pass)

summary(g3)

g4=garchFit(~garch(2,2),data=residuals(a.fin2),trace=FALSE,include.mean=TRUE, na.action=na.pass)

summary(g4)

#The best one is Garch(1,1) model which has the smallest AIC.
```

Homework 4

Lei Zang

Q1.Improve the R quantlets on GH (from CRIX directory on quantlet.de) and make excellent graphics that follow Fig 3,4,5,6 of the "Econometrics of CRIX" paper.

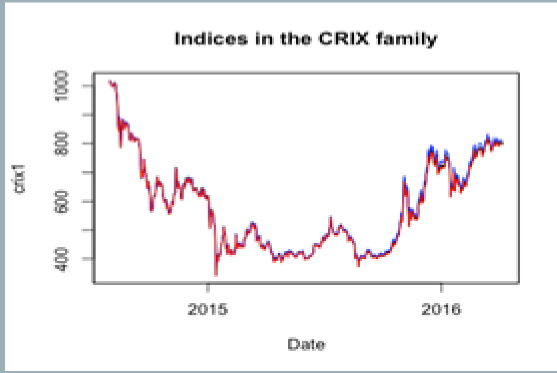


Figure 3: The daily value of indices in the CRIX family

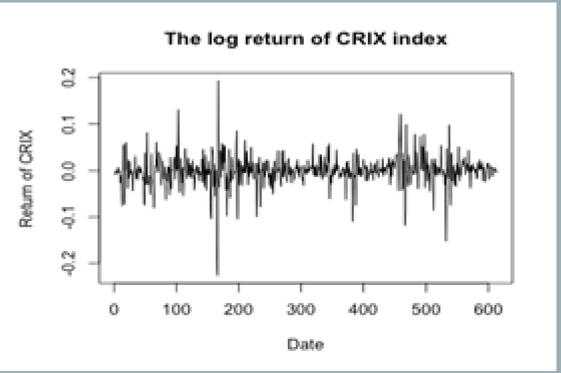


Figure 4: The log returns of CRIX index

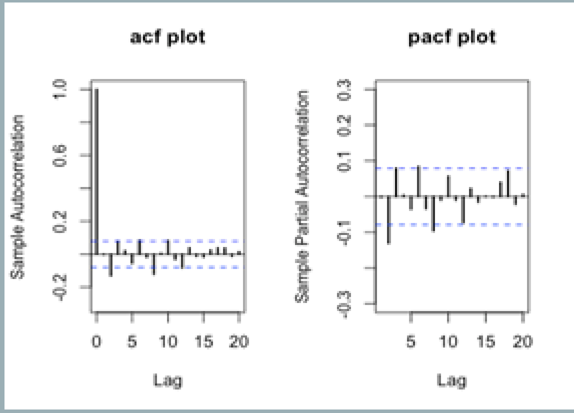


Figure 5: Histogram and QQ plot of CRIX returns

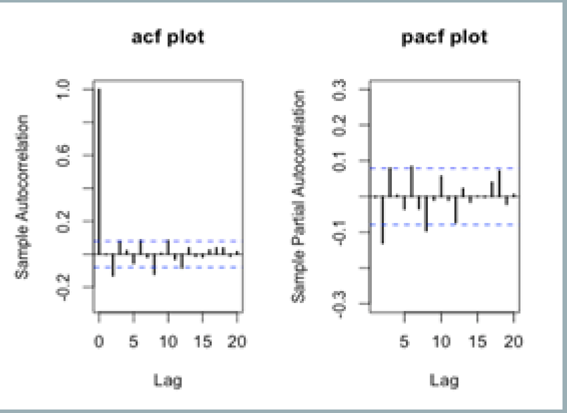


Figure 6: The sample ACF and PACF of CRIX returns

```
rm(list = ls(all = TRUE))
graphics.off()
# install and load packages
libraries = c("zoo", "tseries", "xts", "ccgarch")
lapply(libraries, function(x) if (!(x %in% installed.packages())) { install.packages(x)}
lapply(libraries, library, quietly = TRUE, character.only = TRUE)

# load dataset
load(file.choose())
load(file.choose())
load(file.choose())

# three indices return
ecrix1 = zoo(ecrix, order.by = index(crix1))
efcrix1 = zoo(efcrix, order.by = index(crix1))

# plot with different x-axis scales with zoo
my.panel <- function(x, ...) {
  lines(x, ...)
  lines(ecrix1, col = "blue")
  lines(efcrix1, col = "red")
}
plot.zoo(crix1, plot.type = "multiple", type = "l", lwd = 1.5, panel = my.panel,
         main = "Indices in the CRIX family", xlab = "Date")
```

```
# plot of crix
# plot(as.xts(crix), type="l", auto.grid=FALSE, main = NA)
plot(crix1, ylab = "Price of CRIX", xlab = "Date")

# plot of crix return
ret   = diff(log(crix1))
# plot(as.xts(ret), type="l", auto.grid=FALSE, main = NA)
plot(ret, ylab = "Return of CRIX", xlab = "Date")

# stationary test
adf.test(ret, alternative = "stationary")
kpss.test(ret, null = "Trend")

par(mfrow = c(1, 2))
# histogram of returns
hist(ret, col = "grey", breaks = 20, freq = FALSE, ylim = c(0, 25), xlab = "Return of CRIX")
lines(density(ret), lwd = 2)
mu = mean(ret)
sigma = sd(ret)
x = seq(-4, 4, length = 100)
curve(dnorm(x, mean = mean(ret), sd = sd(ret)), add = TRUE, col = "red",
      lwd = 2)
```

```
#qq-plot
qqnorm(ret)
qqline(ret, col = "blue", lwd = 3)

#acf plot
autocorr = acf(ret, lag.max = 20, ylab = "Sample Autocorrelation", main = "acf plot",
              lwd = 2, ylim = c(-0.3, 1))

#pacf plot
autopcorr = pacf(ret, lag.max = 20, ylab = "Sample Partial Autocorrelation",
                main = "pacf plot", ylim = c(-0.3, 0.3), lwd = 2)
```

Q2. Make your R code perfect as in the R examples on quantlet.de i.e. make sure that the code is "time independent" by using actual dimensions of the data that you are collecting from crix.hu-berlin.de Recreate Fig 7 from "Econometrics of CRIX".

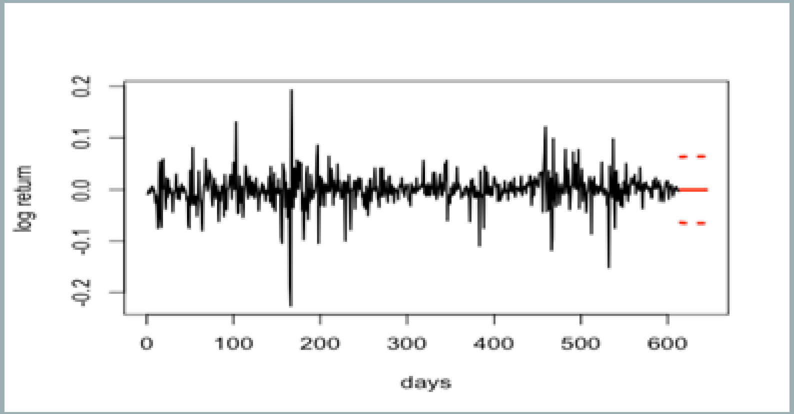


Figure 7: CRIX returns and predicted values.

Codes:

```
#arima model
par(mfrow = c(1, 1))
fit1 = arima(ret, order = c(1, 0, 1))
tsdiag(fit1)
Box.test(fit1$residuals, lag = 1)

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

```
#bic
bic = matrix(NA, 6, 6)
for (p in 0:4) {
  for (q in 0:3) {
    b.p.q = arima(ret, order = c(p, 0, q))
    bic.p.q = AIC(b.p.q, k =
log(length(ret)))
    bic[p + 1, q + 1] = bic.p.q
  }
}

#select p and q order of ARIMA
model
fit4 = arima(ret, order = c(2, 0, 3))
tsdiag(fit4)
Box.test(fit4$residuals, lag = 1)

fitr4 = arima(ret, order = c(2, 1, 3))
tsdiag(fitr4)
Box.test(fitr4$residuals, lag = 1)
```

```
#to conclude, 202 is better than 213
fit202 = arima(ret, order = c(2, 0, 2))

AIC(fit202, k = log(length(ret)))
AIC(fit4, k = log(length(ret)))
AIC(fitr4, k = log(length(ret)))
fit202$aic
fit4$aic
fitr4$aic

#arima202 predict
predict_num = 30
fit202 = arima(ret, order = c(2, 0, 2))
crpre = predict(fit202, n.ahead = predict_num)

dates = seq(as.Date("02/08/2014", format = "%d/%m/%Y"), by = "days", length = length(ret))
plot(ret, type = "l", xlim = c(0, length(ret)+predict_num), ylab = "log return", xlab = "days",
    lwd = 1.5, col = "black")
lines(crpre$pred, col = "red", lwd = 3)
lines(crpre$pred + 2 * crpre$se, col = "red", lty = 3, lwd = 3)
lines(crpre$pred - 2 * crpre$se, col = "red", lty = 3, lwd = 3)
```

Q3. Redo as many figures as you can.

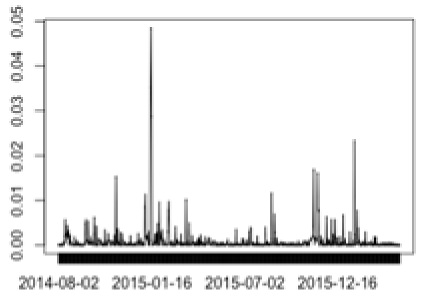


Figure 8: The squared ARIMA(2,0,2) residuals of CRIX returns.

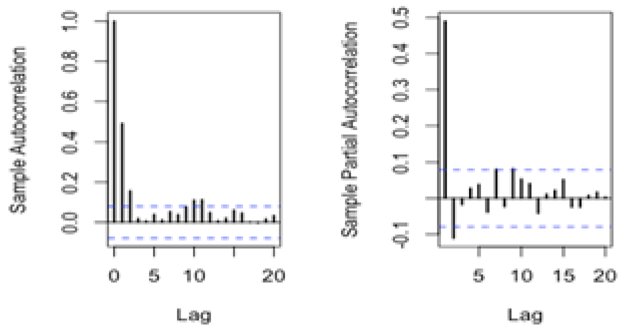


Figure 9: The ACF and PACF of squared ARIMA(2,0,2) residuals

Codes:

```
rm(list = ls(all = TRUE))
graphics.off()

# install and load packages
libraries = c("tseries")
lapply(libraries, function(x) if (!(x %in% installed.packages())) {
  install.packages(x)
})
lapply(libraries, library, quietly = TRUE, character.only = TRUE)

# please change your working directory
setwd()
load(file.choose())
Pr = as.numeric(crix)
Da = factor(date1)
crx = data.frame(Da, Pr)
# plot of crix return
ret = diff(log(crx$Pr))
Dare = factor(date1[-1])
retts = data.frame(Dare, ret)
# arima202 predict
fit202 = arima(ret, order = c(2, 0, 2))
```

```
# vola cluster
par(mfrow = c(1, 1))
res = fit202$residuals
res2 = fit202$residuals^2
tsres202 = data.frame(Dare, res2)
plot(tsres202$Dare,
tsres202$res2, type = "o", ylab =
NA)
lines(tsres202$res2)

# plot(res2, ylab='Squared
residuals', main=NA)
par(mfrow = c(1, 2))
acfres2 = acf(res2, main = NA,
lag.max = 20, ylab = "Sample
Autocorrelation", lwd = 2)
pacfres2 = pacf(res2, lag.max = 20,
ylab = "Sample Partial
Autocorrelation", lwd = 2, main =
NA)
```

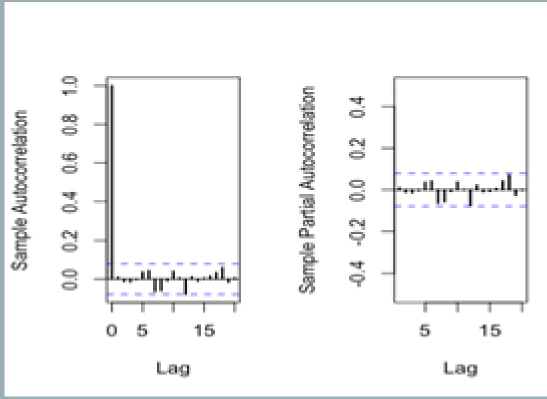


Figure 10: The ACF and PACF of squared ARIMA(2,0,2) residuals

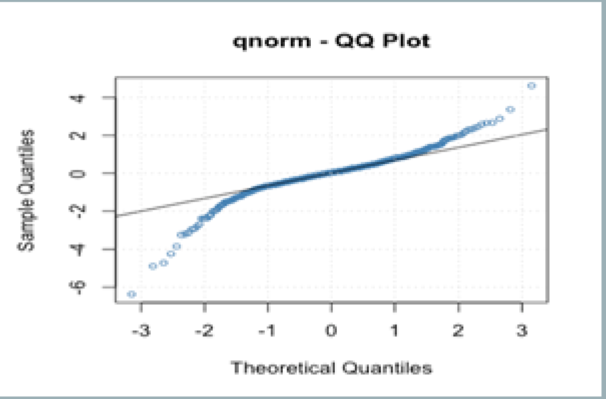


Figure 11: The QQ plots of model residuals of ARIMA-GARCH process.

Codes:

```
rm(list = ls(all = TRUE))
graphics.off()

# install and load packages
libraries = c("tseries")
lapply(libraries, function(x) if (!(x %in% installed.packages())) {
  install.packages(x)
})
lapply(libraries, library, quietly = TRUE, character.only = TRUE)

# please change your working directory
setwd()
load(file.choose())
Pr = as.numeric(crix)
Da = factor(date1)
crx = data.frame(Da, Pr)
# plot of crix return
ret = diff(log(crx$Pr))
Dare = factor(date1[-1])
retts = data.frame(Dare, ret)
# arima202 predict
fit202 = arima(ret, order = c(2, 0, 2))
```

```
# vola cluster
par(mfrow = c(1, 1))
res = fit202$residuals
res2 = fit202$residuals^2
tsres202 = data.frame(Dare, res2)
plot(tsres202$Dare,
tsres202$res2, type = "o", ylab =
NA)
lines(tsres202$res2)

# plot(res2, ylab='Squared
residuals', main=NA)
par(mfrow = c(1, 2))
acfres2 = acf(res2, main = NA,
lag.max = 20, ylab = "Sample
Autocorrelation", lwd = 2)
pacfres2 = pacf(res2, lag.max = 20,
ylab = "Sample Partial
Autocorrelation", lwd = 2, main =
NA)
```

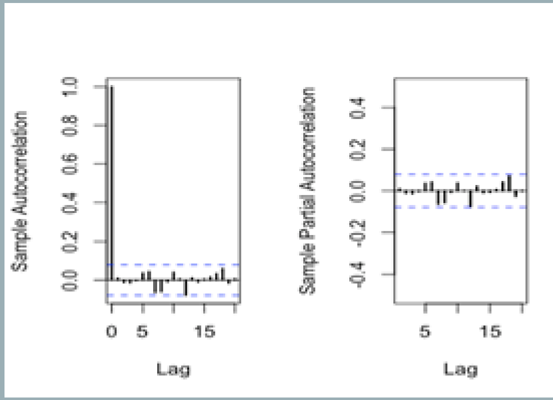


Figure 10: The ACF and PACF of squared ARIMA(2,0,2) residuals

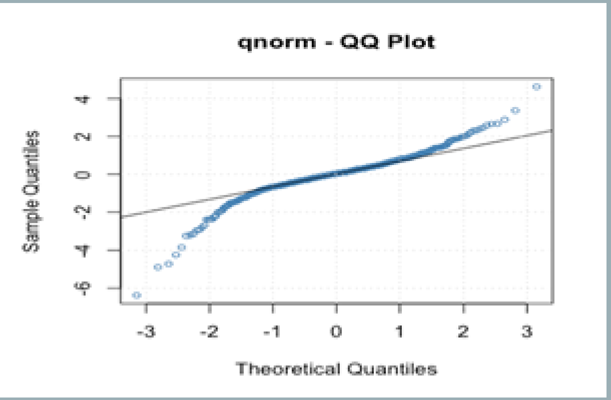


Figure 11: The QQ plots of model residuals of ARIMA-GARCH process.