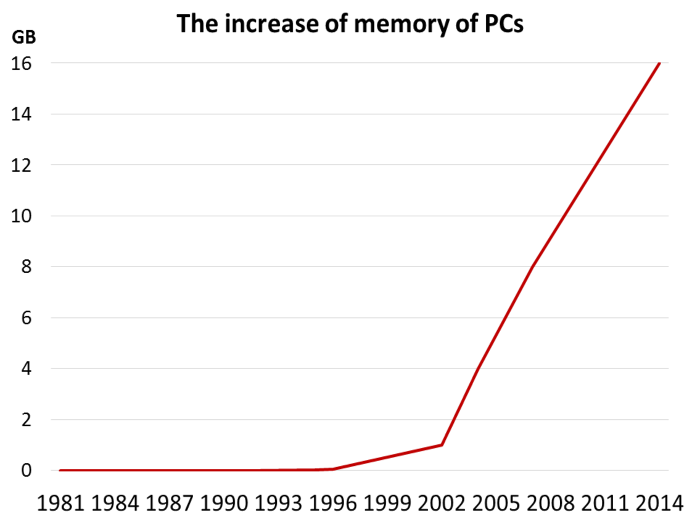
Biying Wan 15620161152279

HW1

1.1 Calculate the increase of memory of PCs over the last 30 years and check whether the FMRI analysis could have been done 20 years ago

Development of memory		Time	Capacity
No memory bank		before 286	64KB-256KB
FPM DRAM	30pin SIMM	1982	256KB
	72pin SIMM	1988-1990	512KB-2MB
EDO DRAM		1991-1995	4MB-16MB
SDR SDRAM	PC66/100/133/150/166	1996-1999	16MB-512MB
Rambus DRAM	PC600/PC700/PC800	1997-2002	64MB-512MB
DDR SDRAM	DDR	2002	128M-1GB
	DDR2	2004	256MB-4GB
	DDR3	2007	512MB-8GB
	DDR4	2014	4GB-16GB



1.2 prepare 2-5 slides explaining logistic regression

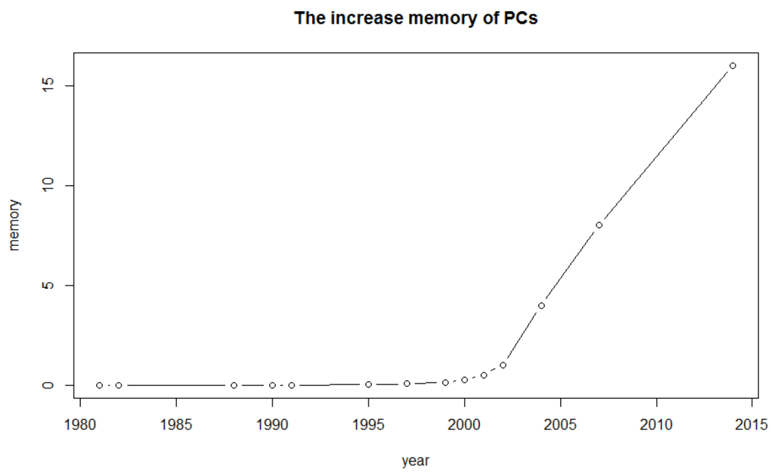
- Logistic Regression - Dichotomous Response variable and numeric and/or categorical explanatory variable(s)
  - ✓ Goal: Model the probability of a particular as a function of the predictor variable(s)
  - ✓ Problem: Probabilities are bounded between 0 and 1
- Distribution of Responses: Binomial
- Link Function: $g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$
- Response - Presence/Absence of characteristic
- Predictor - Numeric variable observed for each case

- Model - $p(x) \equiv$ Probability of presence at predictor level x
- $\pi(x) = \frac{e^{\alpha+\beta x}}{1+e^{\alpha+\beta x}}$
- $\beta = 0 \quad \Rightarrow$ P(Presence) is the same at each level of x
- $\beta > 0 \quad \Rightarrow$ P(Presence) increases as x increases
- $\beta < 0 \quad \Rightarrow$ P(Presence) decreases as x increases
- α, β are unknown parameters and must be estimated using statistical software such as SPSS, SAS, or STATA
- Primary interest in estimating and testing hypotheses regarding b
- Large-Sample test (Wald Test):
- H_0: $\beta = 0$ $\qquad$ H_A: $\beta \neq$
- $T.S.: X^2_{obs} = \left(\frac{\hat{\beta}}{\hat{\sigma}_{\hat{\beta}}}\right)^2$
  $R.R.: X^2_{obs} \geq \chi^2_{\alpha,1}$
  $P-val: P(\chi^2 \geq X^2_{obs})$

HW2

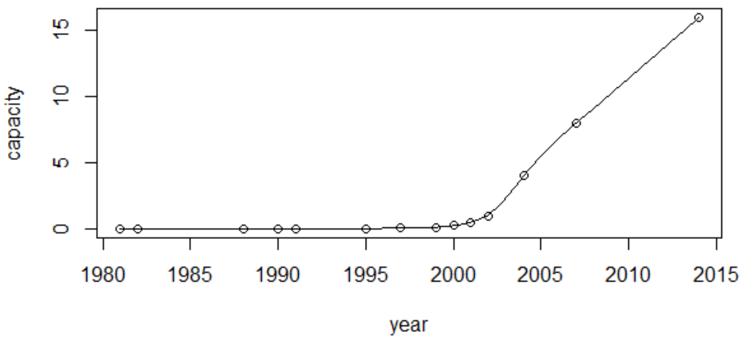
2.1 make an R quantlet to solve HW #1 from unit 1 with R and show it on Github (GH) hint: use the CMB Qs for this work

year<-c(1981,1982,1988,1990,1991,1995,1997,1999,2000,2001,2002,2004,2007,2014)
capacity<-c(6.10352E-05,0.000244141,0.000488281,0.001953125,0.00390625,0.015625,0.0625,0.125,0.25,0.5,1,4,8,16)
plot(year,capacity,type = "b",main = "The increase memory of PCs",xlab = "year",ylab = "memory")



2.2 use R with B-spline code to solve HW#1, any comments?

year<-c(1981,1982,1988,1990,1991,1995,1997,1999,2000,2001,2002,2004,2007,2014)
capacity<-c(6.10352E-05,0.000244141,0.000488281,0.001953125,0.00390625,0.015625,0.0625,0.125,0.25,0.5,1,4,8,16)
library(splines)
plot(year,capacity)
lines(spline(year,capacity))



2.3 Suppose you observe that in n=1000 mails (in 1 week) you have about 2 scams. Use the LvB /Poisson cdf to calculate that you have 6 scam emails in 2 weeks. In Scammyland you have 5 scams on average, what is the probability to have no scam mail.

（1）LvB PDF:

dbinom(x = 6,size = 2000,prob =2/1000)

[1] 0.1042477

With LvB PDF, the probability is 0.104.

（2）Possion PDF:

dpois(x = 0,lambda = 5)

[1] 0.006737947

With Possion PDF, the probability is 0.00674.

HW3

3.1 make an R quantlet on GH to produce hash code for the 2 sentences: “I learn a lot from this class when I am proper listening to the professor”, “I do not learn a lot from this class when I am absent and playing on my Iphone”. Compare the 2 hash sequences

- install.packages("digest")
- library("digest")
- digest("I learn a lot from this class when I am proper listening to the professor","sha256")
- [1]"c16700de5a5c1961e279135f2be7dcf9c187cb6b21ac8032308c715e1ce9964c"
- digest("I do not learn a lot from this class when I am absent and playing on my Iphone","sha256")
- [1]"2533d529768409d1c09d50451d9125fdbaa6e5fd4efdeb45c04e3c68bcb3a63e"

3.2 Make 3-5 slides (in PPTX) on the DSA (Digital Signature Algorithms)

- A **digital signature** is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file, etc.) is **authentic**. Digital signatures are used to verify that a message or document was authored by a certain person, and that it was not altered or modified by anyone else.
- One of the most common digital signature mechanisms is DSA. The **Digital Signature Algorithm (DSA)** is the basis of the **Digital Signature Standard (DSS)**, a U.S. Government document.
- DSA lets one person with a secret key “sign” a document, so that others with a matching public key can verify it must have been signed only by the holder of the secret key.
- Digital signatures depend on **hash functions**, which are one-way computations done on a

message. They are called “one-way” because there is no known way (without infeasible amounts of computation) to find a message with a given hash value. The result has a fixed length, which is 160 bits in the case of the Secure Hash Algorithm (SHA) used by DSA.

- In practice, digital signatures are used to sign the hash values of messages, not the messages themselves. Thus it is possible to sign a message’s hash value, without even knowing the content of the message. This makes it possible to have *digital notaries*, who can verify a document existed (and was signed), without the notary knowing anything about what was in the document.

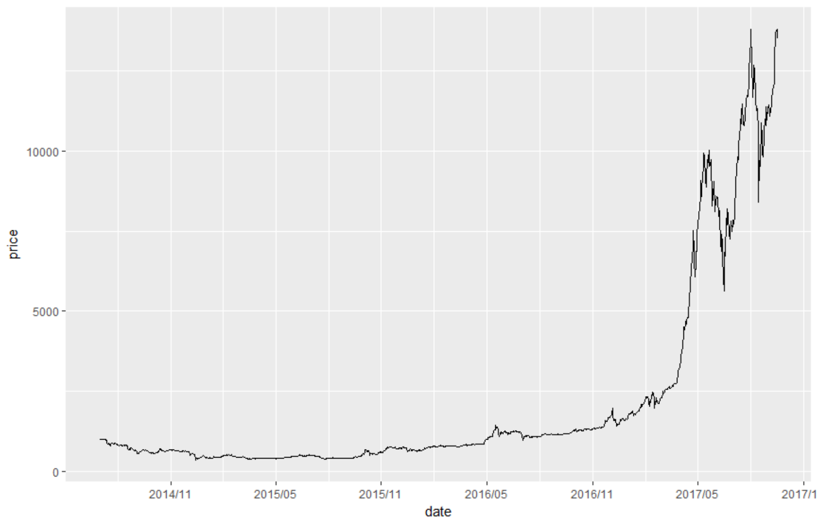
3.3 Make slides with R code where you create a JSON data set that you save and read again.

- install.packages("rjson")
- library("rjson")
- json_file3="D:/研二/研二上/大数据与互联网金融/HW3/test.json"
- json_data3<- fromJSON(paste(readLines(json_file3), collapse=""))
- json_data3<- as.data.frame(json_data3)
- print(json_data3)

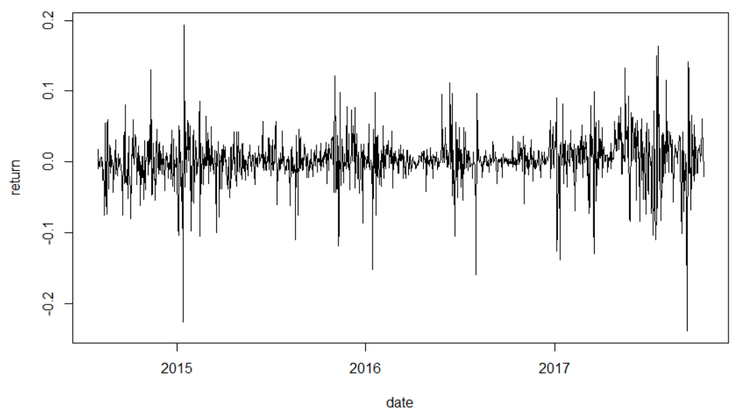
```
name email gender hobby name.1 email.1 gender.1 hobby.1
lucy   @01   male  surf    lim     @02     male    surf
lucy   @01   male  ball    lim     @02     male    ball
```

3.4 Download the CRIX data and make a plot of the time series, analyse its properties, i.e. fit ARMA, ARIMA etc. Is there a GARCH effect?

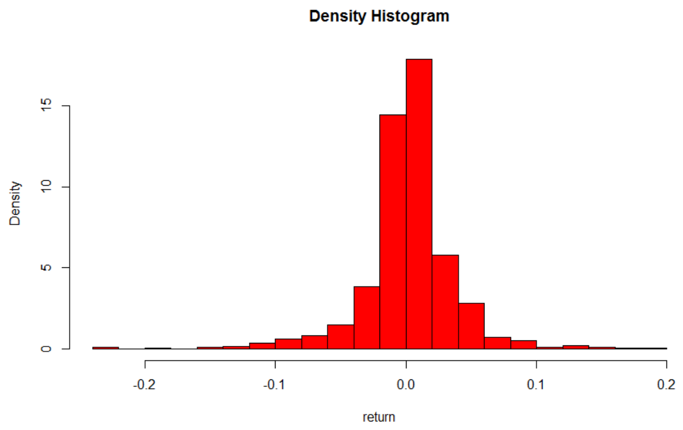
- install.packages("ggplot2")
- install.packages("scales")
- library("rjson")
- json_file="D:/研二/研二上/大数据与互联网金融/HW3/crix.json"
- json_data <- fromJSON(paste(readLines(json_file), collapse=""))
- json_df <- as.data.frame(c(json_data[[1]][1],json_data[[1]][2]))
- for (i in 2:length(json_data)){
-     json_df <- rbind(json_df,as.data.frame(c(json_data[[i]][1],json_data[[i]][2])))
-     }
- json_df$date <- as.POSIXct(json_df$date)
- library(ggplot2)
- library(scales)
- ggplot(json_df)+
-     geom_line(aes(x=date,y=price))+
-     scale_x_datetime(breaks=date_breaks("6 month"),labels=date_format("%Y/%m"))



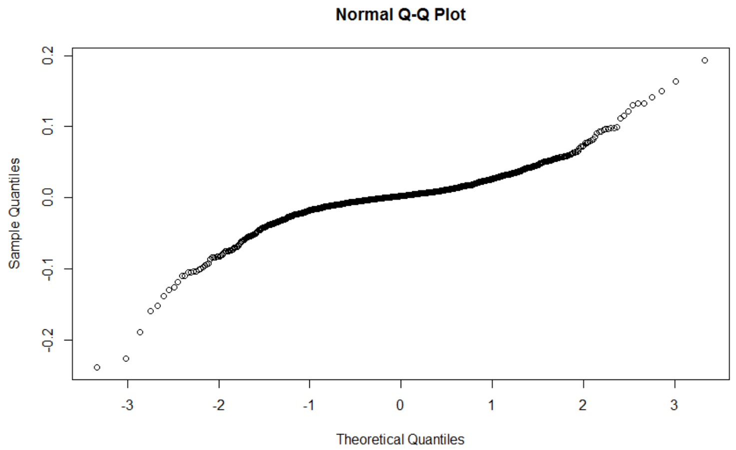
- x<-json_df[,2]
- return<-log(x[2:nrow(json_df)])-log(x[1:nrow(json_df)-1])
- return<-c(NA,return)
- json_return<-as.data.frame(cbind(json_df,return))
- json_return<-json_return[,-2]
- plot(json_return,type="l")



- hist(json_return$return,freq=FALSE,breaks=12,col="red",xlab="return",main="Density Histogram")



- qqnorm(json_return$return)



Therefore, CRIX returns is not normally distributed.

HW 4

4.1 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.

```
#download data
#price
library("rjson")
#json_file="D:/研二/研二上/大数据与互联网金融/HW4/crix.json"
json_file="http://crix.hu-berlin.de/data/crix.json"
json_data <- fromJSON(paste(readLines(json_file), collapse=""))
json_df <- as.data.frame(c(json_data[[1]][1],json_data[[1]][2]))
for (i in 2:length(json_data)){
   json_df <- rbind(json_df,as.data.frame(c(json_data[[i]][1],json_data[[i]][2])))
}
json_df$date <- as.POSIXct(json_df$date)
# return
x<-json_df[,2]
return<-log(x[2:nrow(json_df)])-log(x[1:nrow(json_df)-1])
```

```
return<-c(NA,return)
json_return<-as.data.frame(cbind(json_df,return))
json_return<-json_return[-1,-2]
#save dataset
save(json_df,file="D:/研二/研二上/大数据与互联网金融/HW4/crix.RData")
save(json_return,file="D:/研二/研二上/大数据与互联网金融/HW4/return.RData")
#clear
rm(list = ls(all = TRUE)) #rm(list = ls())
graphics.off()
# install and load packages
libraries = c("zoo", "tseries", "xts")
lapply(libraries, function(x) if (!(x %in% installed.packages())) {
  install.packages(x)
})
lapply(libraries, library, quietly = TRUE, character.only = TRUE)
#load dataset
load(file = "D:/研二/研二上/大数据与互联网金融/HW4/crix.RData")
load(file = "D:/研二/研二上/大数据与互联网金融/HW4/return.RData")
#fig 3 in Econometrics of CRIX, plot of crix
#plot(json_df,xlab=NA,ylab=NA,type="l",col="red")
library(ggplot2)
library(scales)
ggplot(json_df)+
  geom_line(aes(x=date,y=price))+
  scale_x_datetime(breaks=date_breaks("4 month"),labels=date_format("%Y/%m"))
```

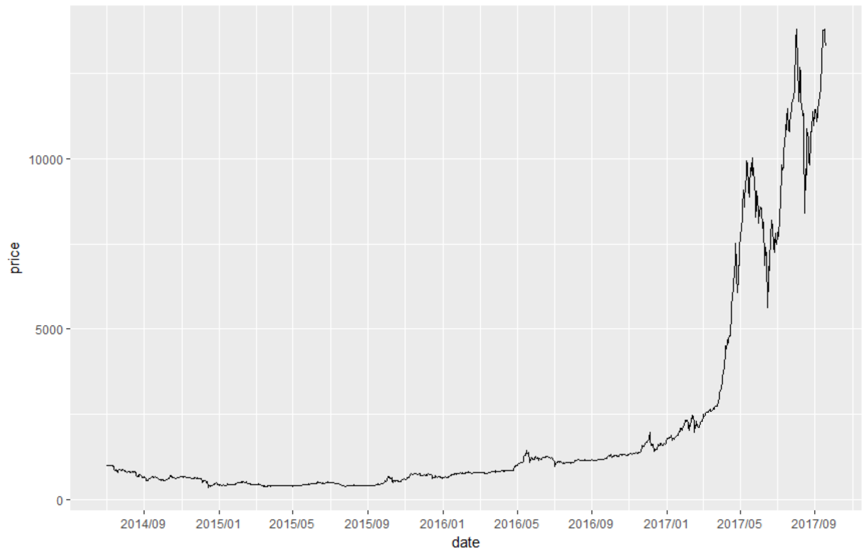


Figure 1 The daily value of CRIX index from 01/08/2014 to 19/10/2017

```
#fig 4 in Econometrics of CRIX ,plot of return
#plot(json_return,xlab=NA,ylab=NA,type="l")
ggplot(json_return)+
  geom_line(aes(x=date,y=return))+
```

```
scale_x_datetime(breaks=date_breaks("4 month"),labels=date_format("%Y/%m"))
```

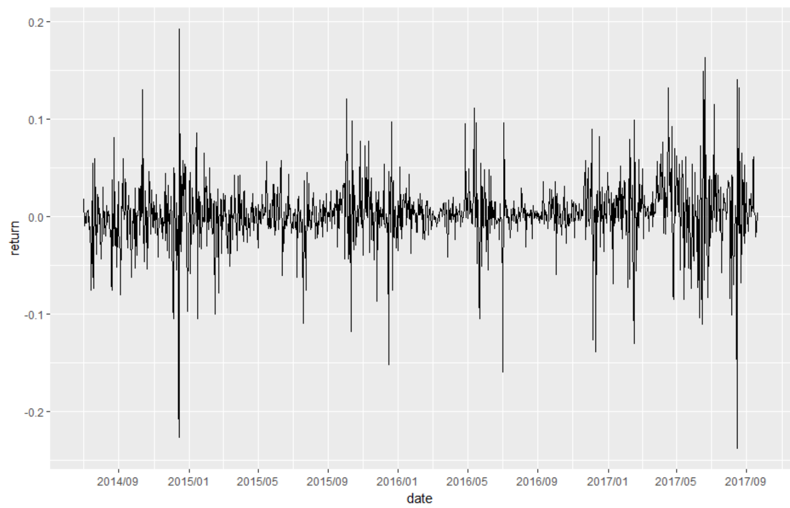


Figure 2 The log returns of CRIX index from 01/08/2014 to 19/10/2017

```
#fig 5 in Econometrics of CRIX,histogram of returns,qq-plot
return<-json_return[,2]
par(mfrow = c(1, 2))
# histogram of returns
hist(return, col = "grey", breaks = 20, freq = FALSE, ylim = c(0, 25), xlab = "return")
lines(density(return), lwd = 2)
mu = mean(return)
sigma = sd(return)
x = seq(-4, 4, length = 100)
curve(dnorm(x, mu, sigma), add = TRUE, col = "darkblue", lwd = 2)
# qq-plot
qqnorm(return)
qqline(return, col = "blue", lwd = 3)
```

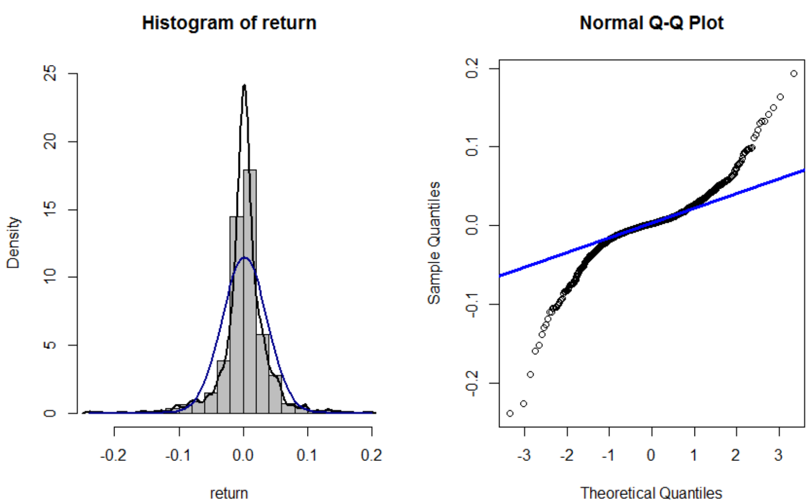


Figure 3 Histogram and QQ plot of CRIX returns from 01/08/2014 to 19/10/2017

```
#fig 6 in Econometrics of CRIX
par(mfrow = c(1, 2))
```

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

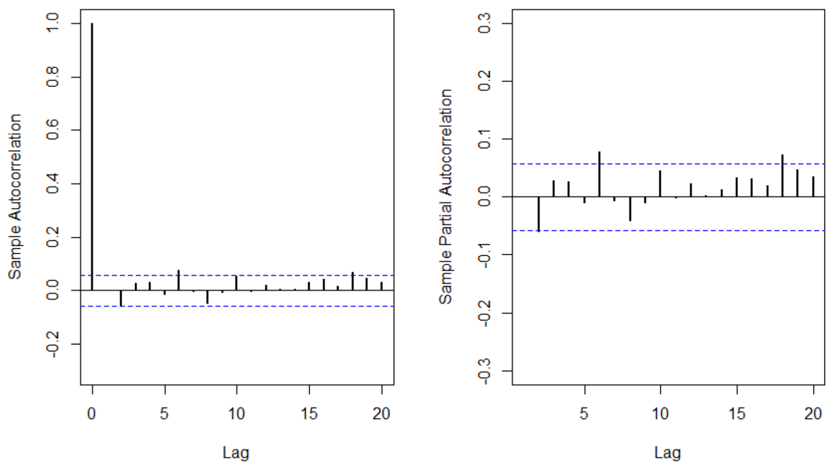


Figure 4 The sample ACF and PACF of CRIX returns from 01/08/2014 to 19/10/2017

4.2 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".

```
#fig 7 in Econometrics of CRIX, arima202 predict
graphics.off()
fit202 = arima(return, order = c(2, 0, 2))
crpre = predict(fit202, n.ahead = 30)
dates = seq(as.Date("01/08/2014", format = "%d/%m/%Y"), by = "days", length = length(return))
plot(return, type = "l", xlim = c(0, 1206), ylab = "return", xlab = "days", lwd = 1.5)
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)
```

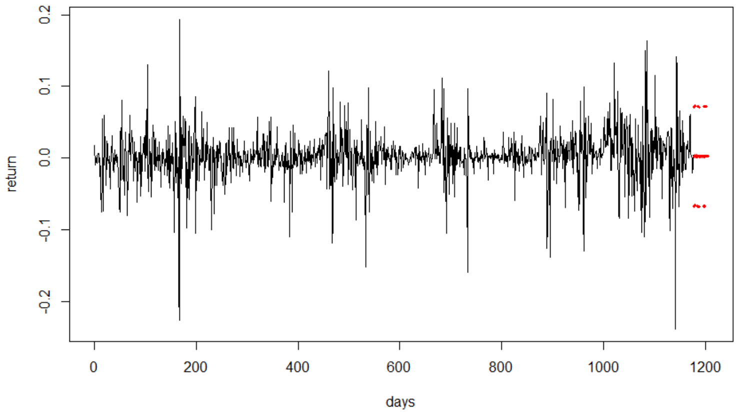


Figure 5 CRIX returns and predicted values

4.3 redo as many figures as you can.

```
# fig 8 in Econometrics of CRIX, Volatility cluster
graphics.off()
date=json_return$date
Volatility= fit202$residuals^2
tsres202 = data.frame(date,Volatility)
#plot(tsres202, type = "l",xlab="date", ylab = "Volatility")
ggplot(tsres202)+
  geom_line(aes(x=date,y=Volatility))+
  scale_x_datetime(breaks=date_breaks("4 month"),labels=date_format("%Y/%m"))
```

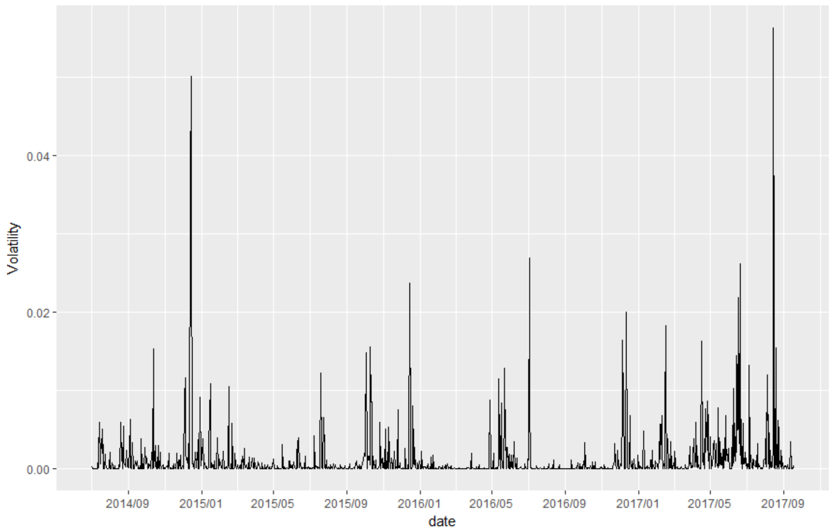


Figure 6 The squared ARIMA (2,0,2) residuals of CRIX returns