



报告题目： **Final Exam Project\_\_**

姓 名： **颜 晶 15620161152292**

任课老师： **Wolfgang 教授\_\_\_\_**

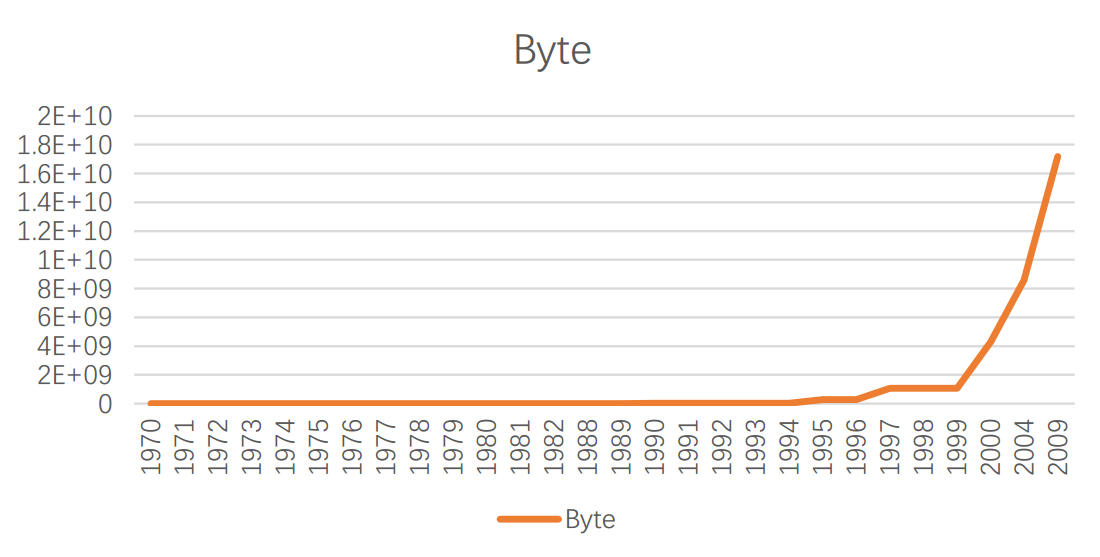
日 期：  **2017 年 11月 20 日\_**

**HW Unit1**

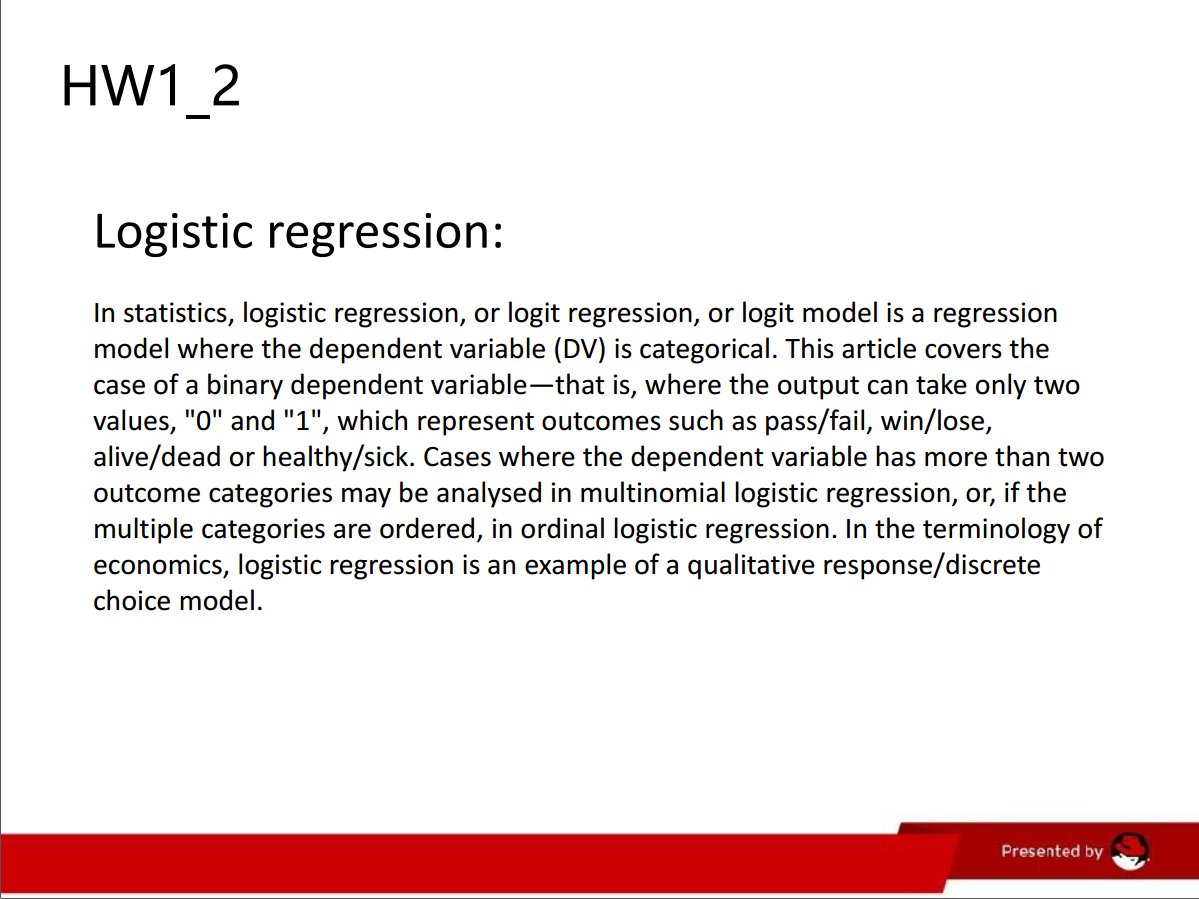
* 1. **Calculate the increase of memory of PCs over the last 30 years and check whether the FMRI analysis could have been done 20 years ago.**

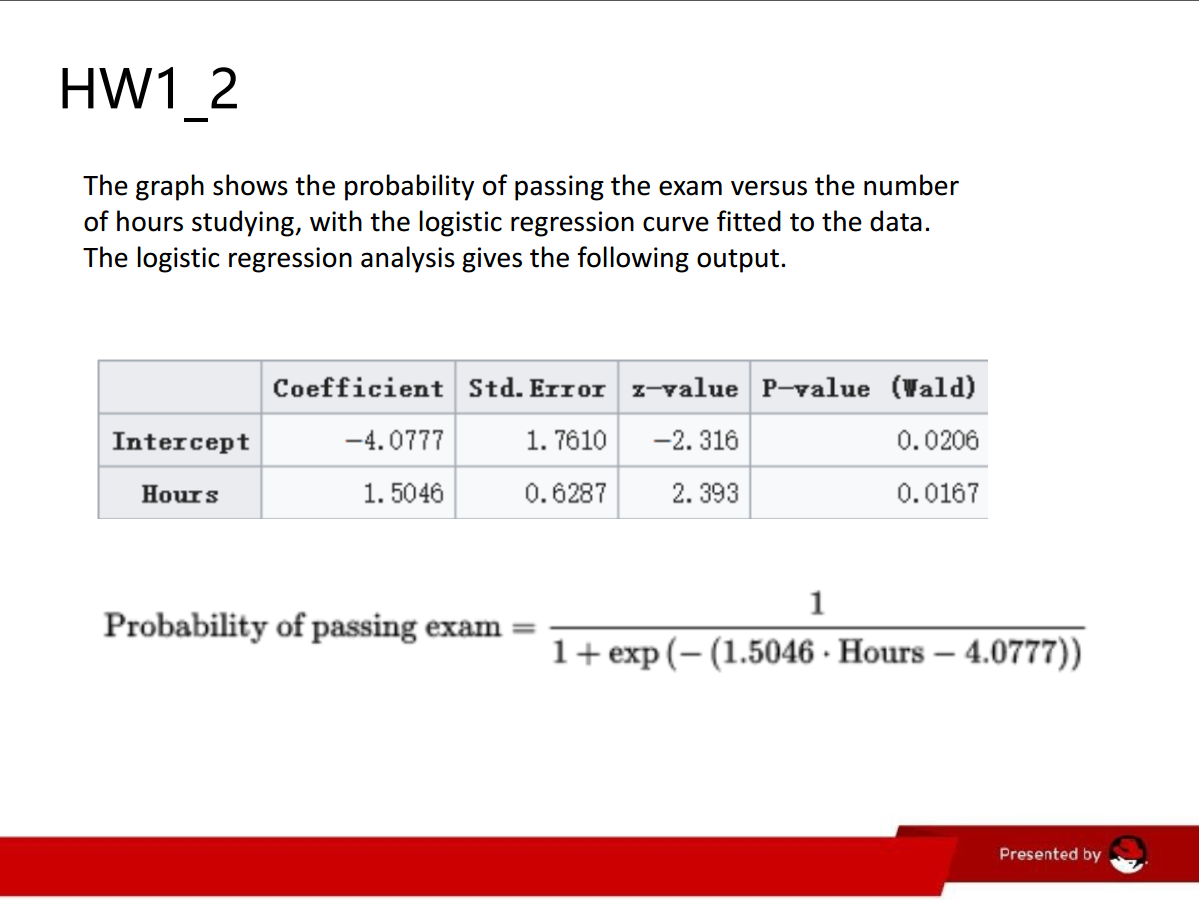
|  |  |
| --- | --- |
| **year** | **Byte** |
| 1970 | 262144 |
| 1971 | 262144 |
| 1972 | 262144 |
| 1973 | 262144 |
| 1974 | 262144 |
| 1975 | 262144 |
| 1976 | 262144 |
| 1977 | 262144 |
| 1978 | 262144 |
| 1979 | 262144 |
| 1980 | 262144 |
| 1981 | 262144 |
| 1982 | 262144 |
| 1988 | 2097152 |
|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

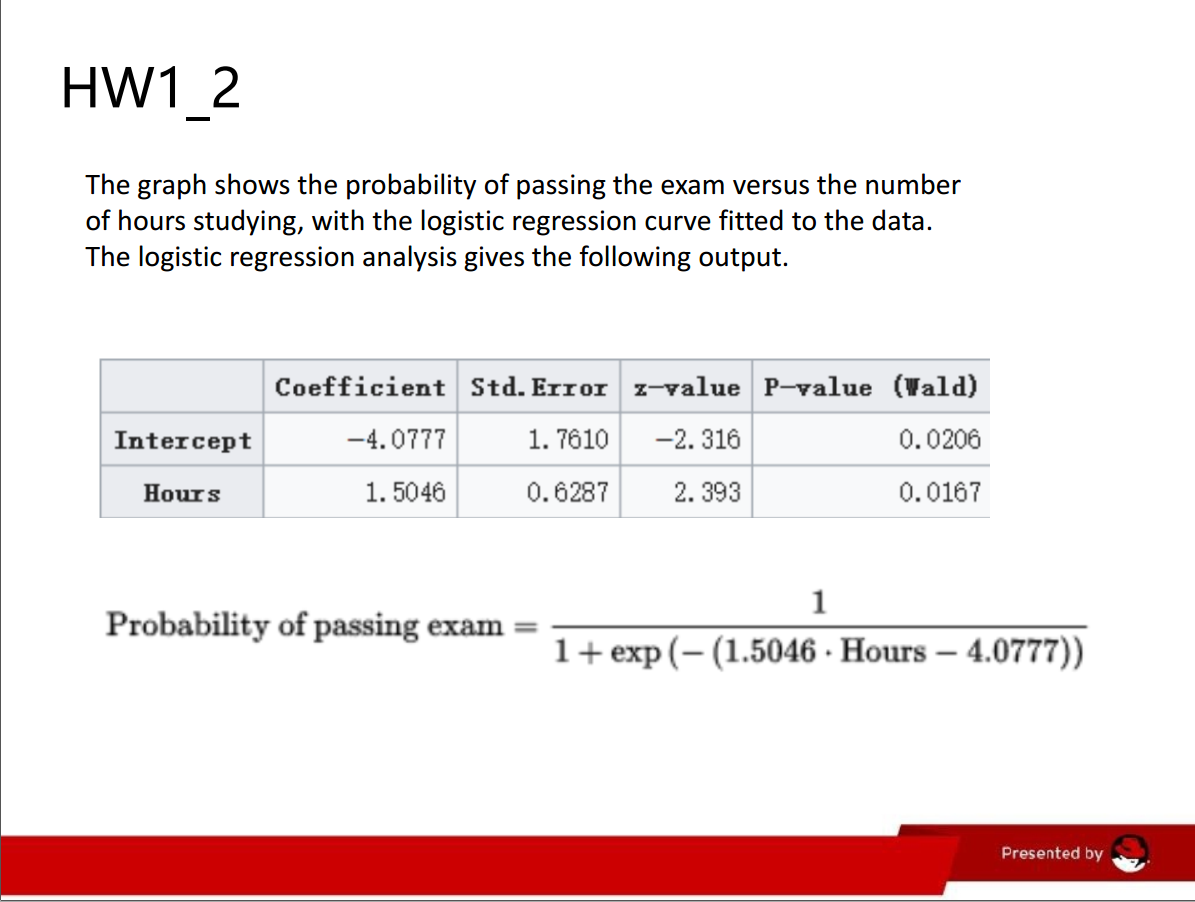
**FMRI analysis could have been done 20 years ago.**

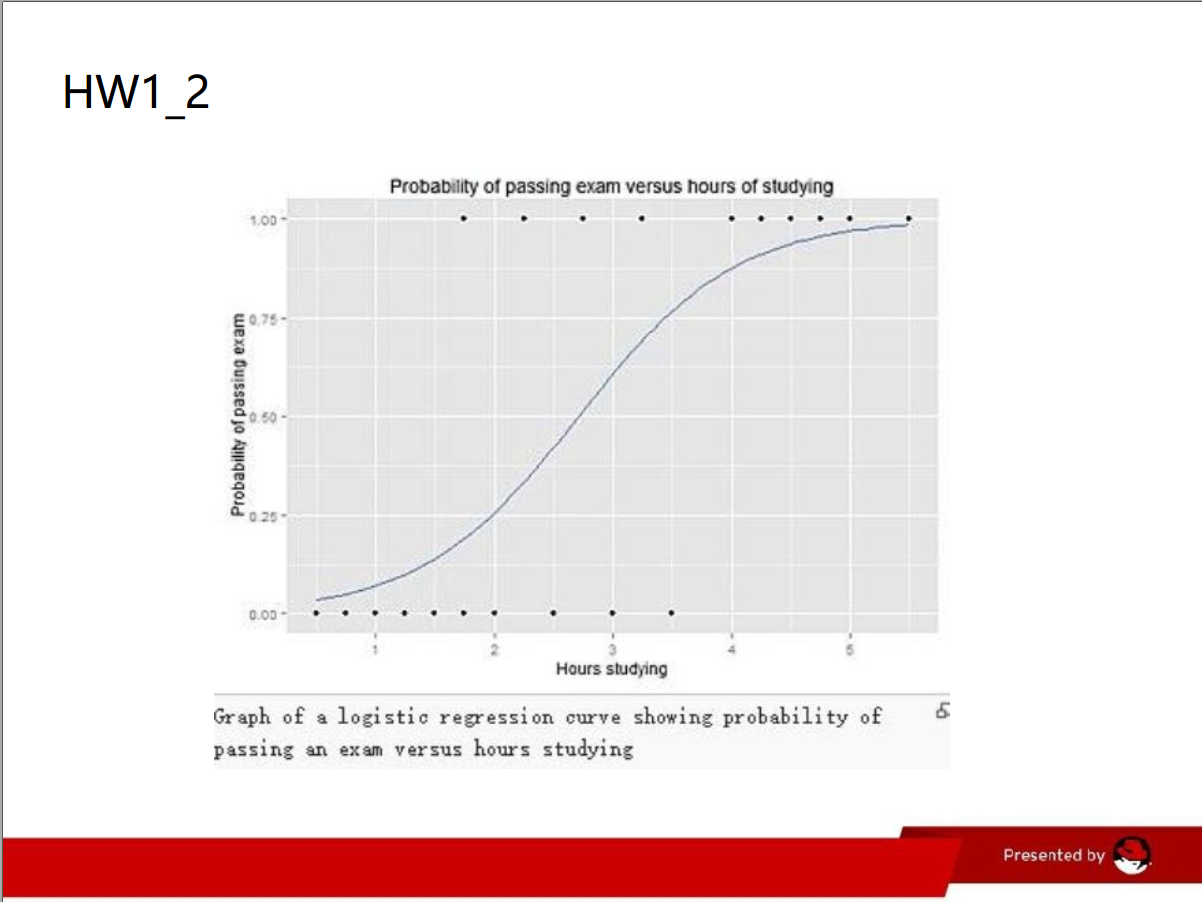


* 1. **Prepare 2-5 slides explaining logistic regression.**









**HW Unit 2**

**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.**

setwd("C:/Users/xiumei/Desktop/big data") 2.0read.csv("hw unit2.csv")  
plot(year,RAM,type ="o",col="red",main = "RAM of computer")

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

f<-read.csv("hw unit2.csv")  
year<-f$Year;RAM<-f$RAM  
plot(year,RAM,type ="o",col="black",main = "RAM of computer")  
require(datasets)  
require(class)  
require(grDevices)  
require(lattice)  
x= year  
y = RAM  
splines.reg.l1 = smooth.spline(x,y, spar = 0.2) # lambda = 0.2  
splines.reg.l2 = smooth.spline(x,y, spar = 1) # lambda = 1  
splines.reg.l3 = smooth.spline(x,y, spar = 2) # lambda = 2  
lines(splines.reg.l1, col = "red", lwd = 2) # regression line with lambda = 0.2

**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.**

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 )

**HW Unit 3**

**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.**

library("digest") 2.0# now do the hash code calculation  
digest("I learn a lot from this class when I am proper listening to  
the professor")  
digest("I do not learn a lot from this class when I am absent and  
playing on my Iphone")

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

WHAT IS DSA

Digital signatures are essential to verify the sender of a document’s identity. The  
signature is computer using a set of rules and algorithm such that the identity of  
the person can be verified.  
The signature is generated by the use of a private key that known only to the user.  
The signature is verified when a public key is corresponds to the private key. With  
every user having a public/private key pair, this is an example of public-key  
cryptography.  
Public keys, which are known by everyone, can be used to verify the signature of a  
user. The private key, which is never shared, is used in signature generation, which  
can only be done by the user .

WHAT CAN DSA DO?

Digital signatures are used to detect unauthorized modifications to data. Also, the  
recipient of a digitally signed document in proving to a third party that the  
document was indeed signed by the person who it is claimed to be signed by. This  
is known as nonrepudiation, because the person who signed the document cannot  
repudiate the signature at a later time.  
Digital signature algorithms can be used in e-mails, electronic funds transfer,  
electronic data interchange, software distribution, data storage, and just about any  
application that would need to assure the integrity and originality of data.。

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

>library(RJSONIO) 2.0> letter<-LETTERS[1:10]  
>country<-c("China","the US","the UK","Russia",  
"Korea","Japan","Italy","Brazil","India","Germany")  
> data<-data.frame(letter,country)  
> da<-as.matrix(data)  
>cat(toJSON(da))

[ {  
"letter": "A",  
"country": "China"  
},  
{  
"letter": "B",  
"country": "the US"  
},  
{  
"letter": "C",  
"country": "the UK"  
},  
{  
"letter": "D",  
"country": "Russia"  
},  
{  
"letter": "E",  
"country": "Korea"  
}

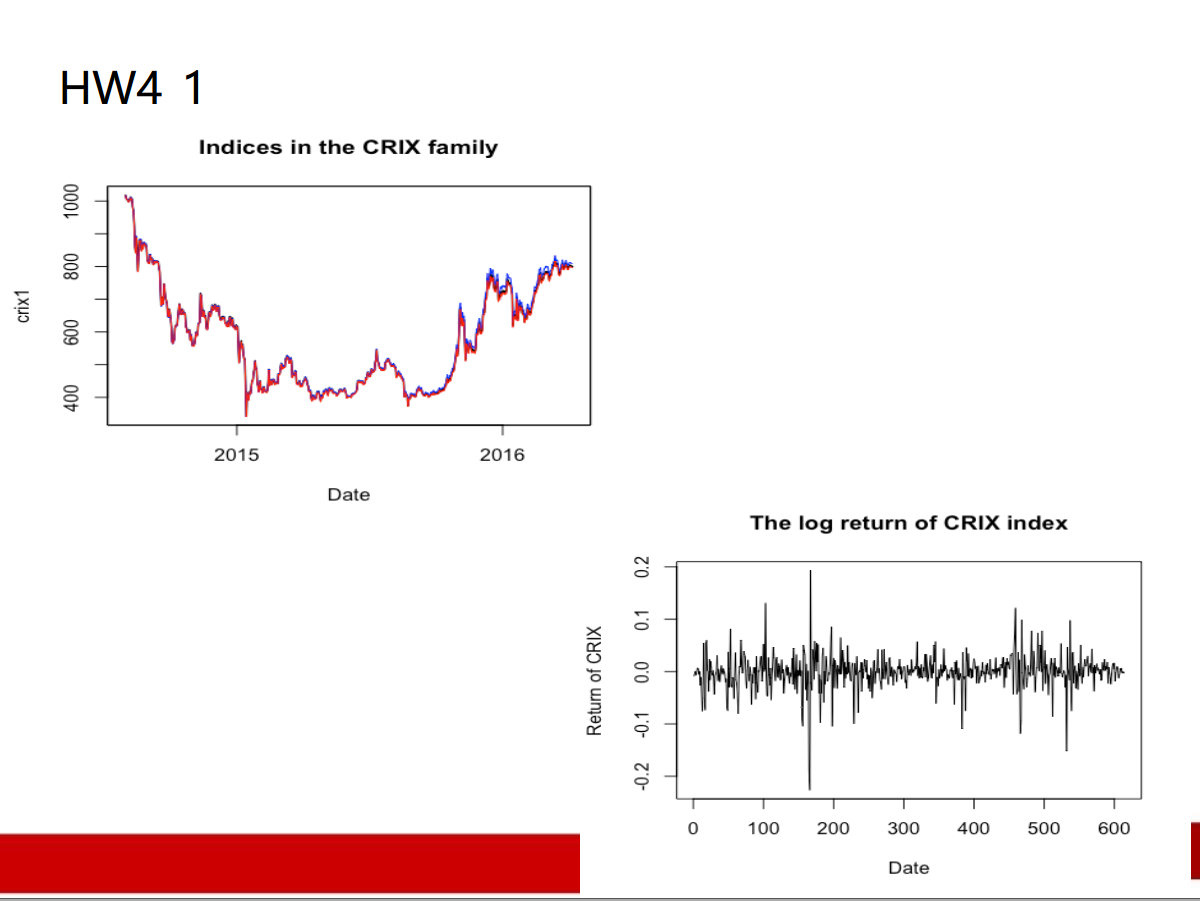
{  
"letter": "F",  
"country": "Japan"  
},  
{  
"letter": "G",  
"country": "Italy"  
},  
{  
"letter": "H",  
"country": "Brazil"  
},  
{  
"letter": "I",  
"country": "India"  
},  
{  
"letter": "J",  
"country": "Germany"  
} ]

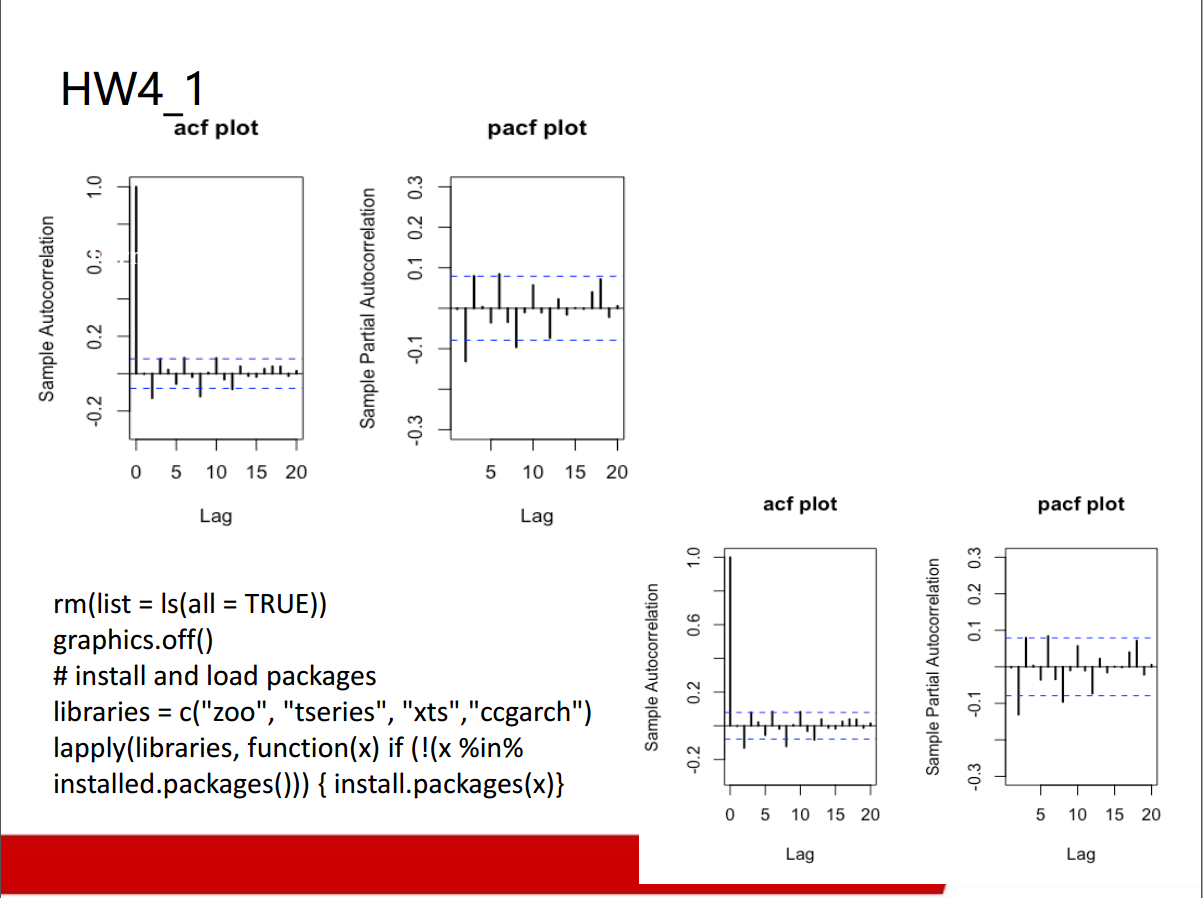
**3.4 Download the CRIX data and make a plot of the time series, analyse its properties, i.e. ﬁt ARMA, ARIMA etc. Is there a GARCH effect?**

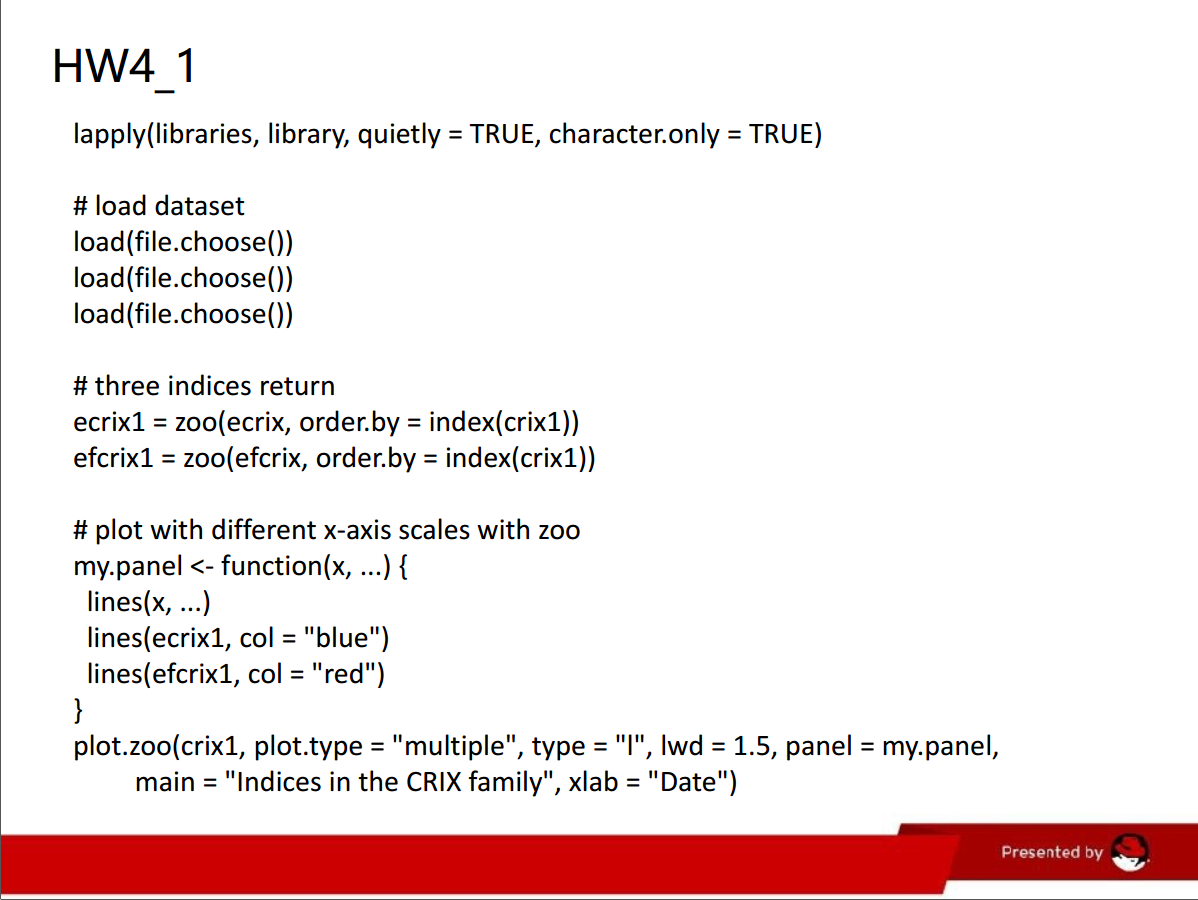
#install.packages("rjson", repos="http://cran.us.r-project.org")  
library(rjson)  
json\_file = "http://crix.hu-berlin.de/data/crix.json"  
json\_data = fromJSON(file=json\_file)  
crix <- Reduce(rbind,json\_data)  
crix\_data\_frame <- as.data.frame(crix)  
lst <- lapply(json\_data,function(x)  
{  
df<-data.frame(date=x$date,price=x$price)  
return(df)  
})  
crix\_data\_frame <- Reduce(rbind,lst)  
plot(crix\_data\_frame$date,crix\_data\_frame$price)  
#library(forecast)  
#library(tseries)  
plot(crix\_data\_frame)

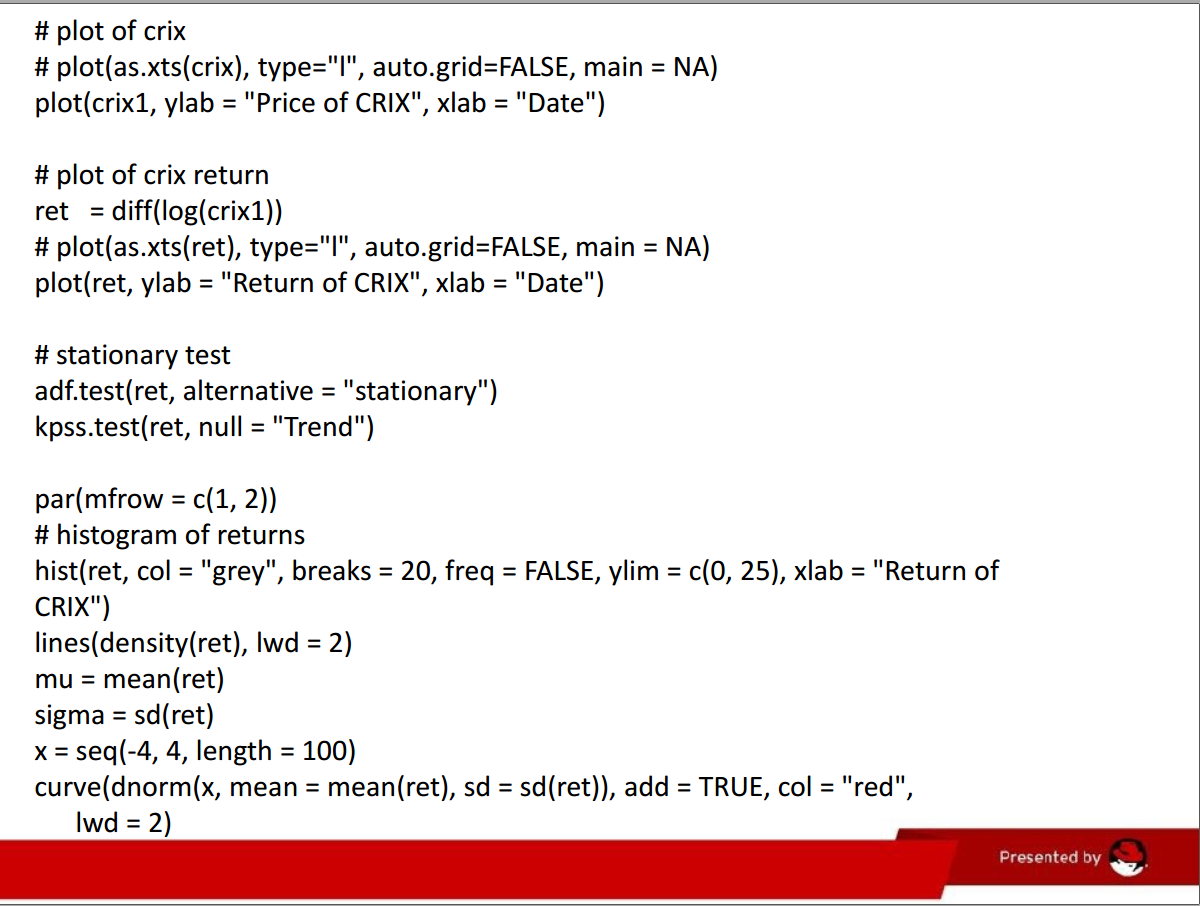
**Unit 4 HW**

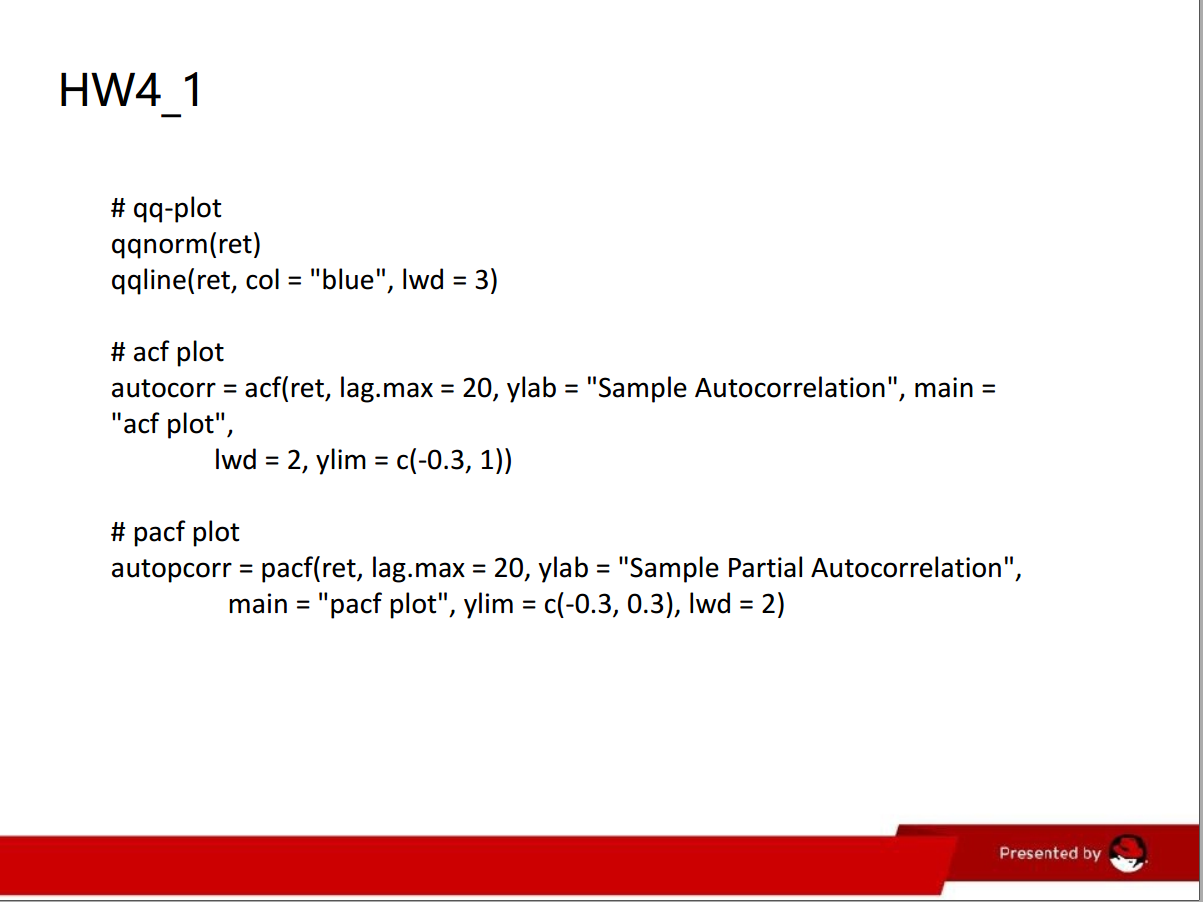
**4.1 Improve the R quantlets on GH (from CRIX directory on quantlet.de) and make excellent graphics that follow Fig 3,4,5,6 of the „Econometrics of CRIX“ paper.**



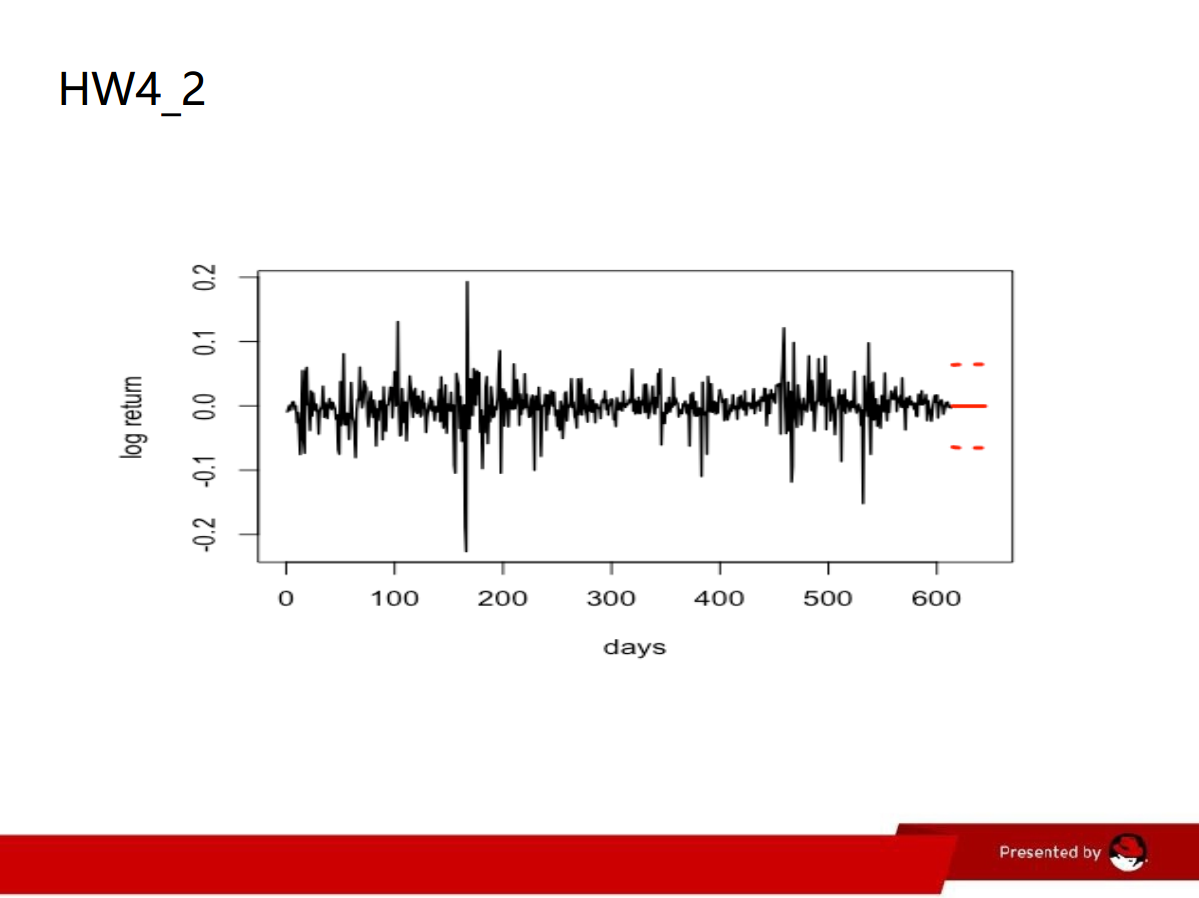


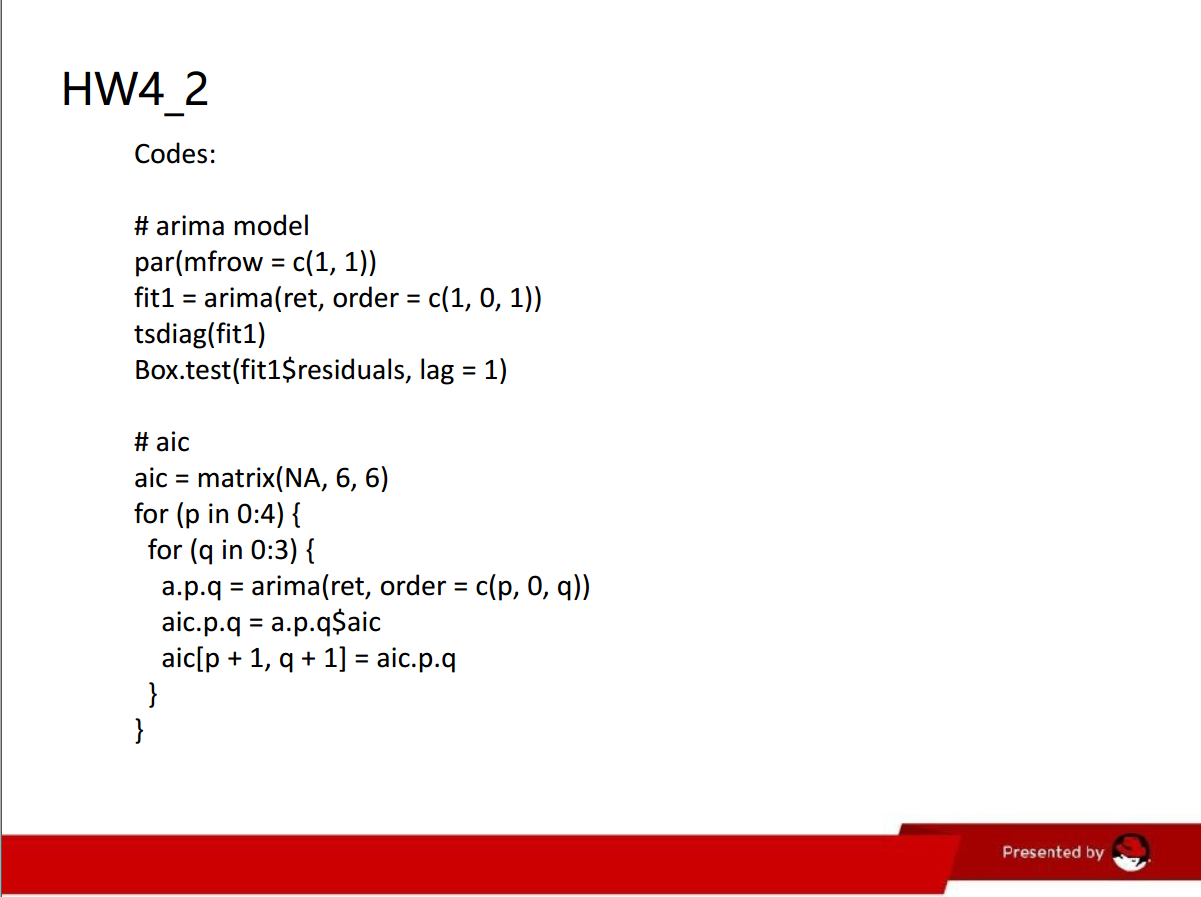


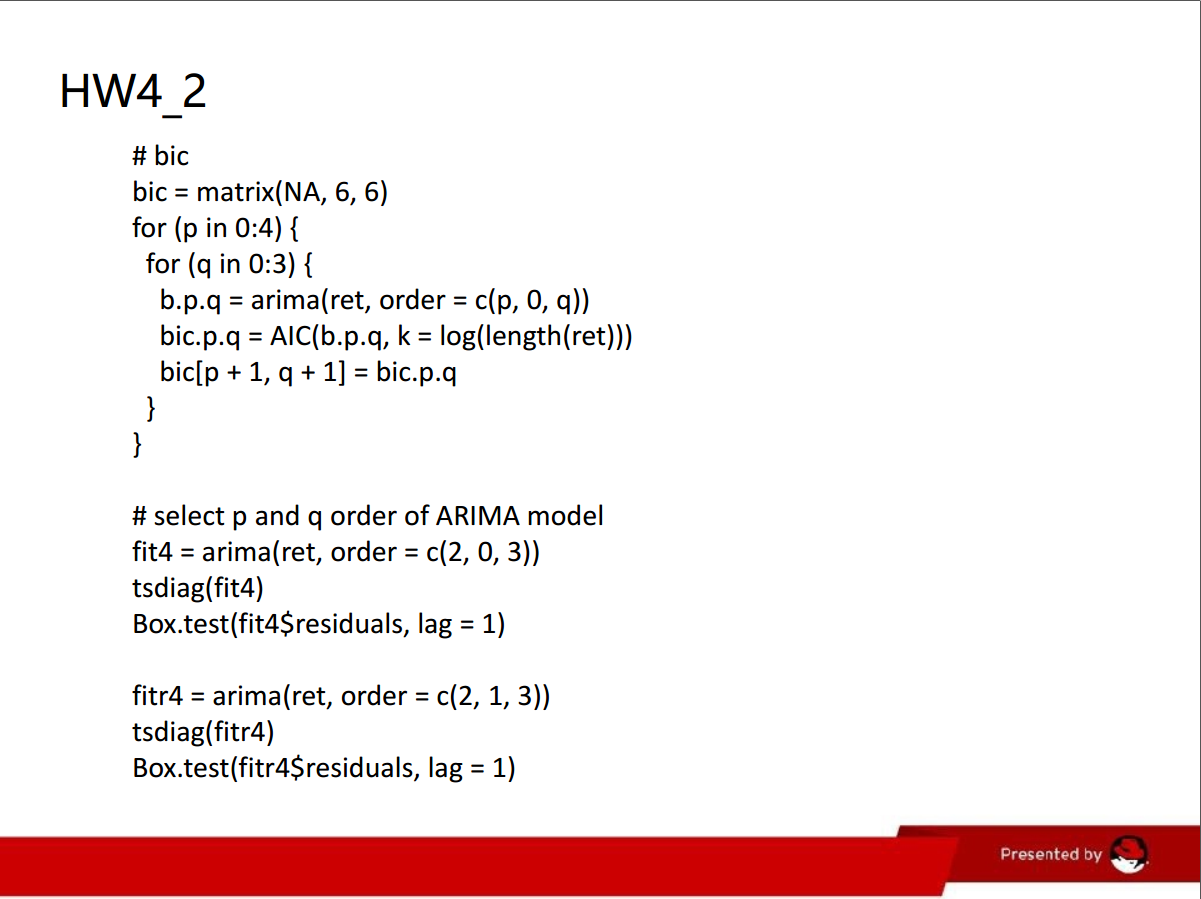


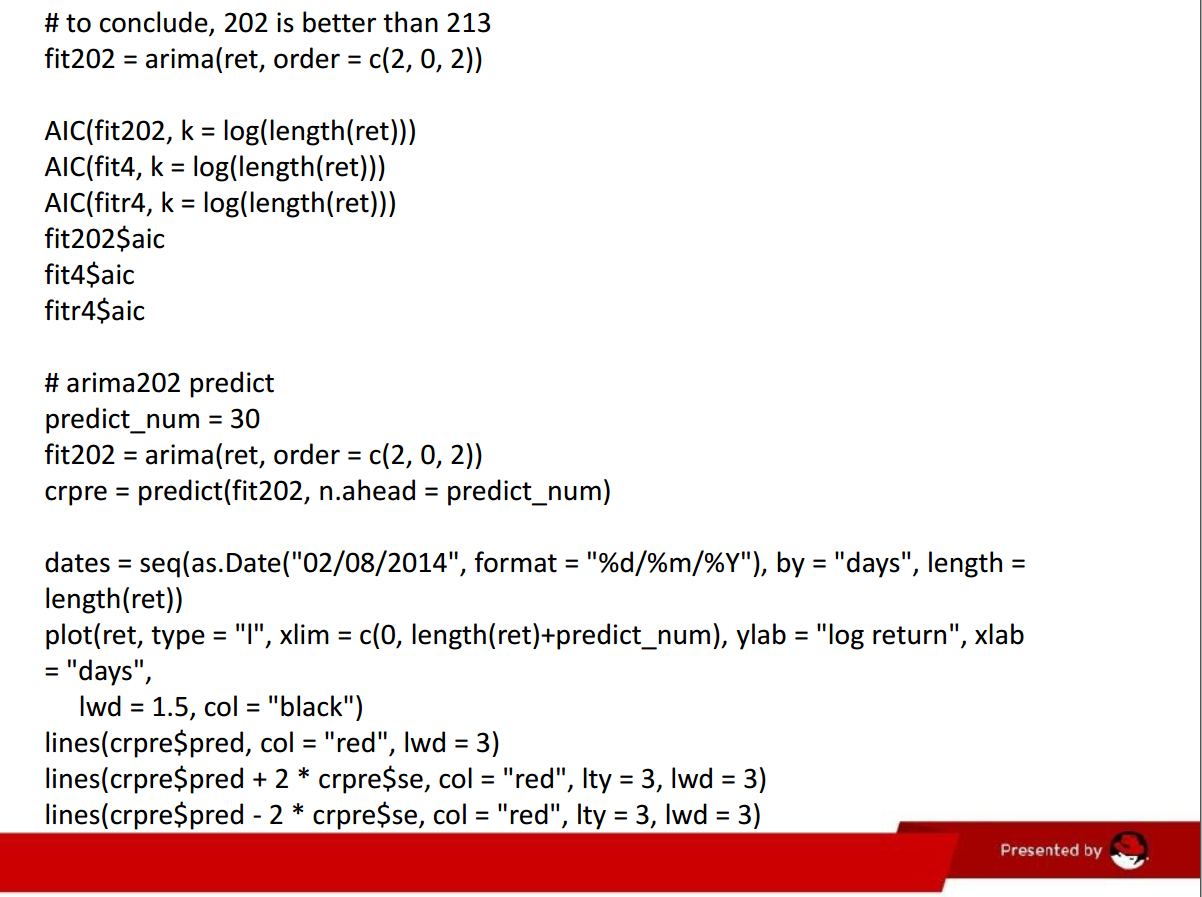


**4.2 Make your R code perfect as in the R examples on quantlet.de i.e. make sure that the code is „time independent“ by using actual dimensions of the data that you are collecting from crix.hu-berlin.de RecreateFig 7 from „Econometrics of CRIX“.**



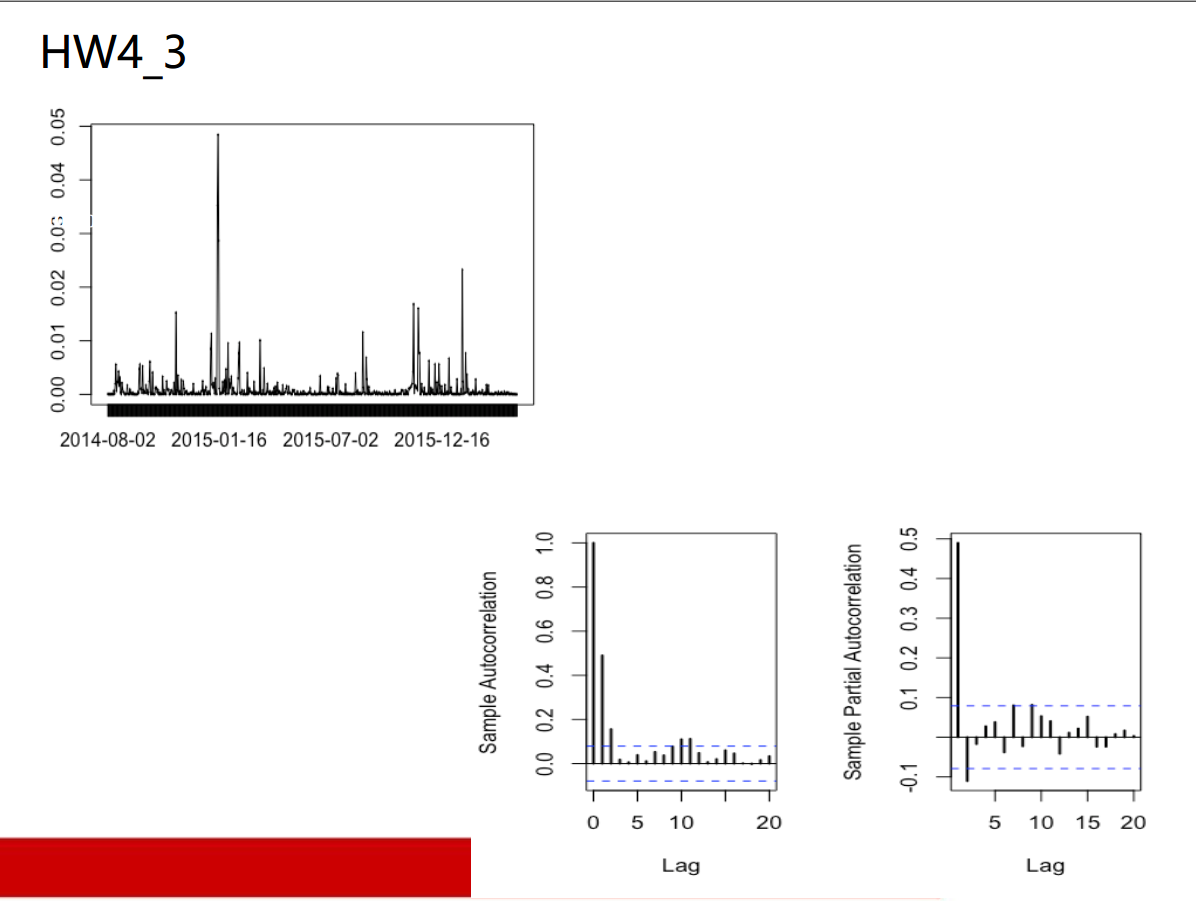


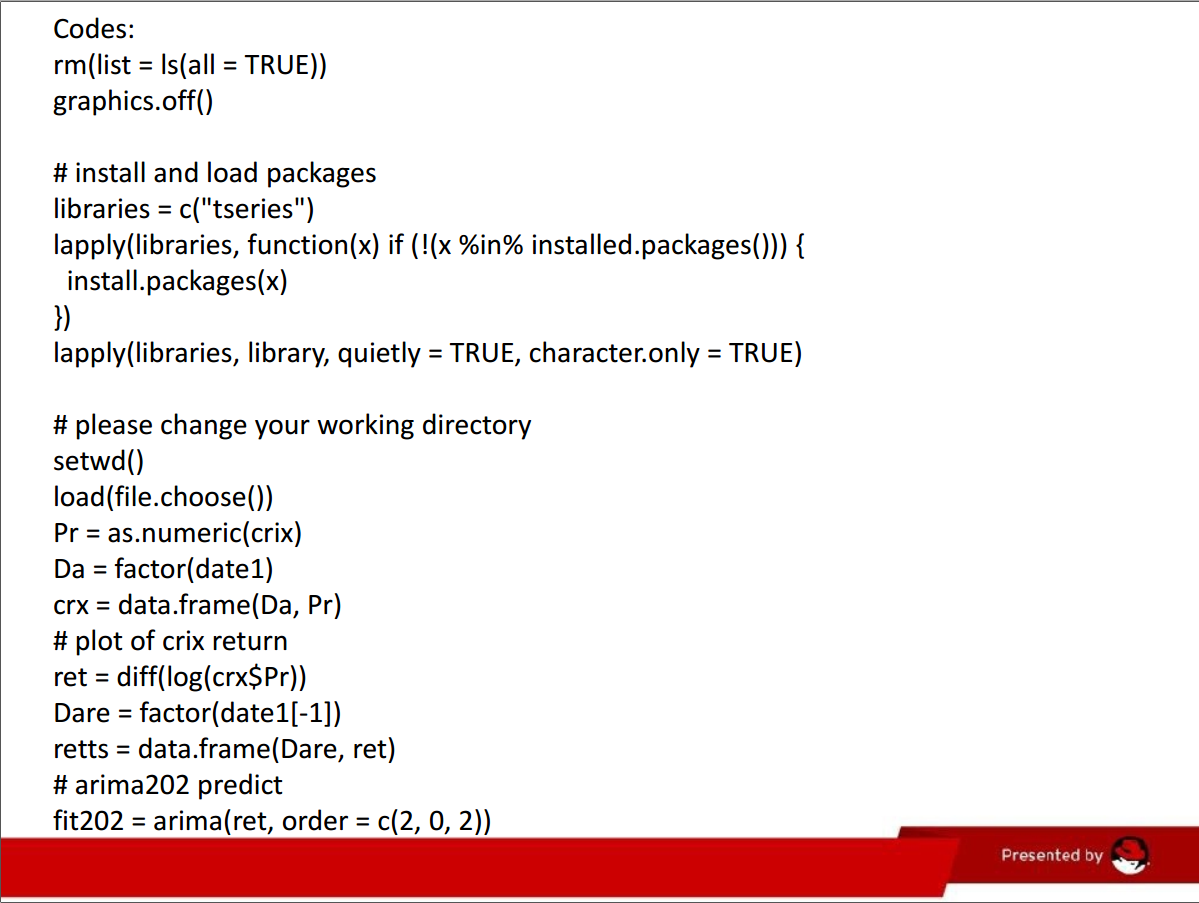


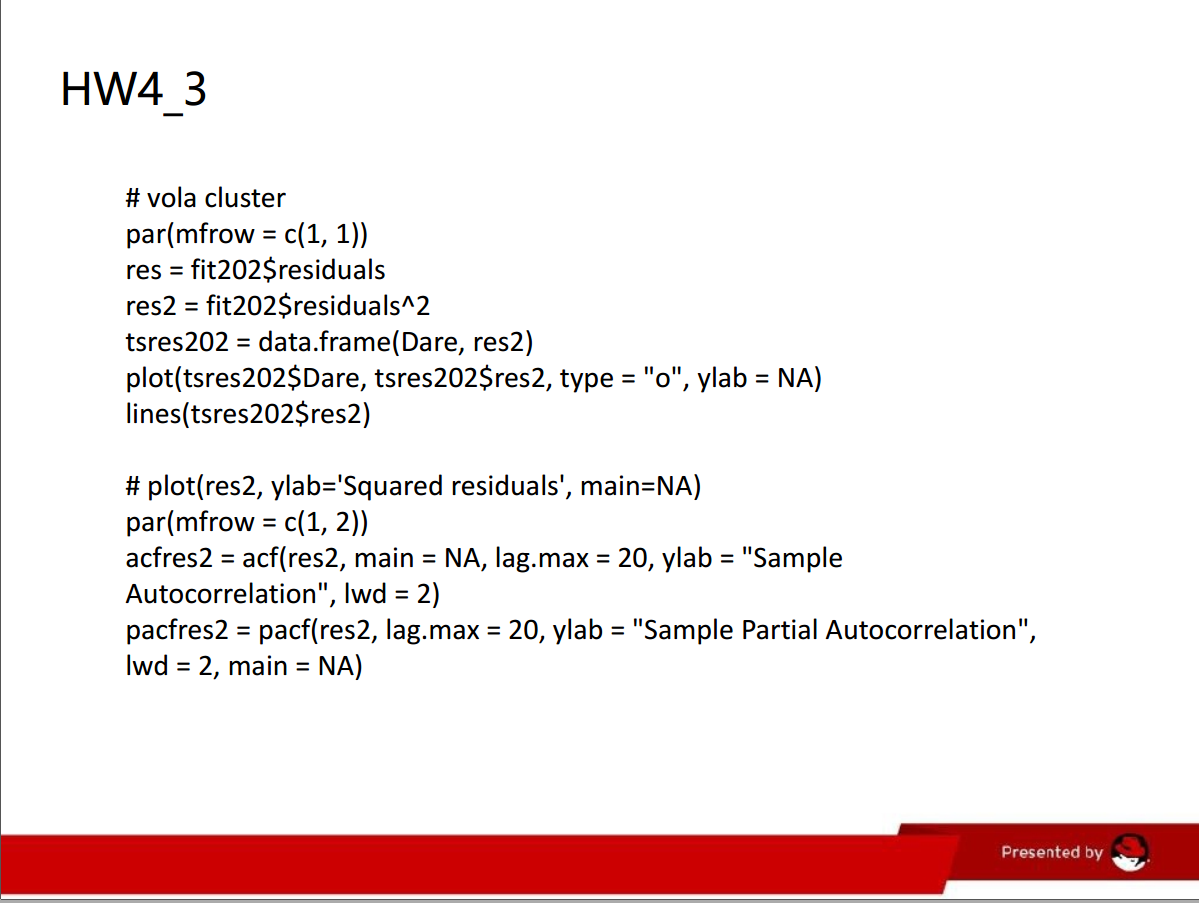


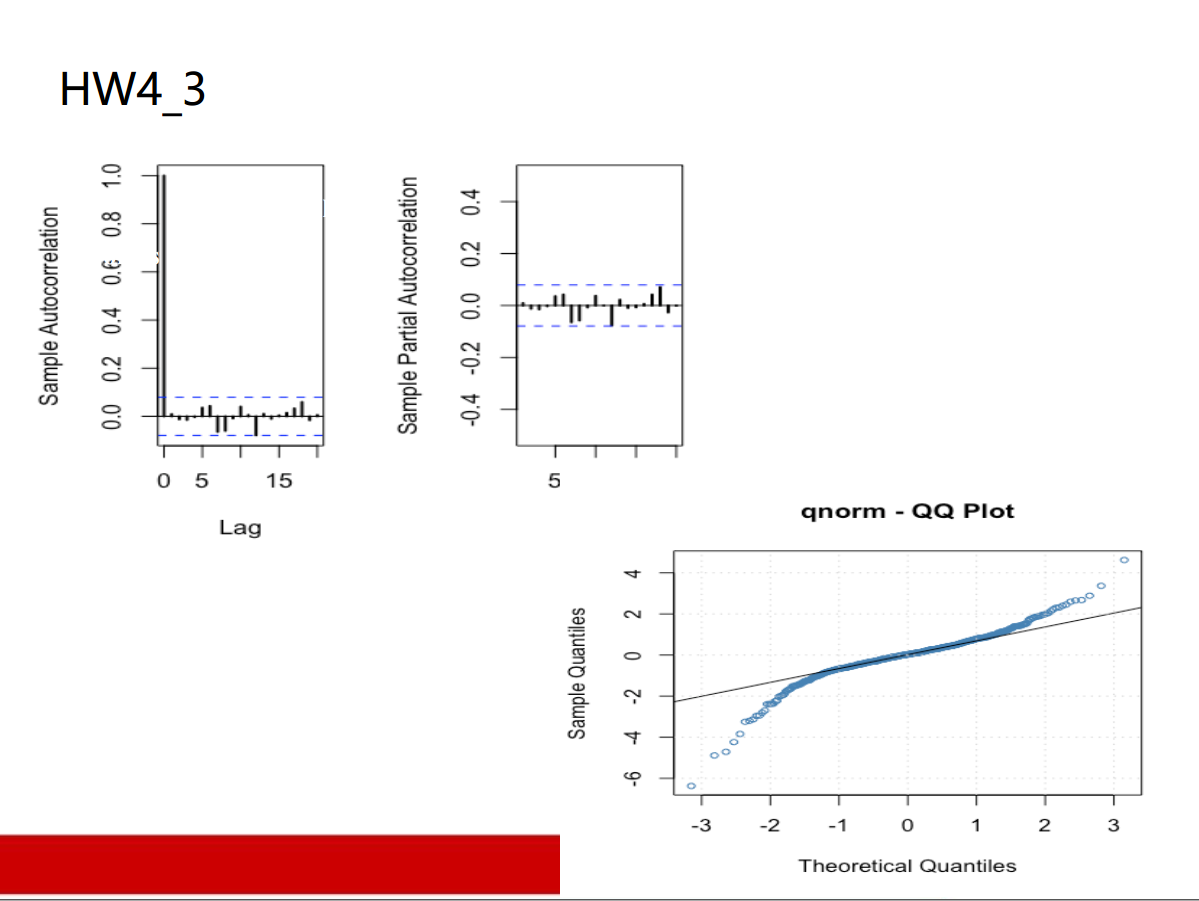
**4.3 Redo as many ﬁgures as you can.**

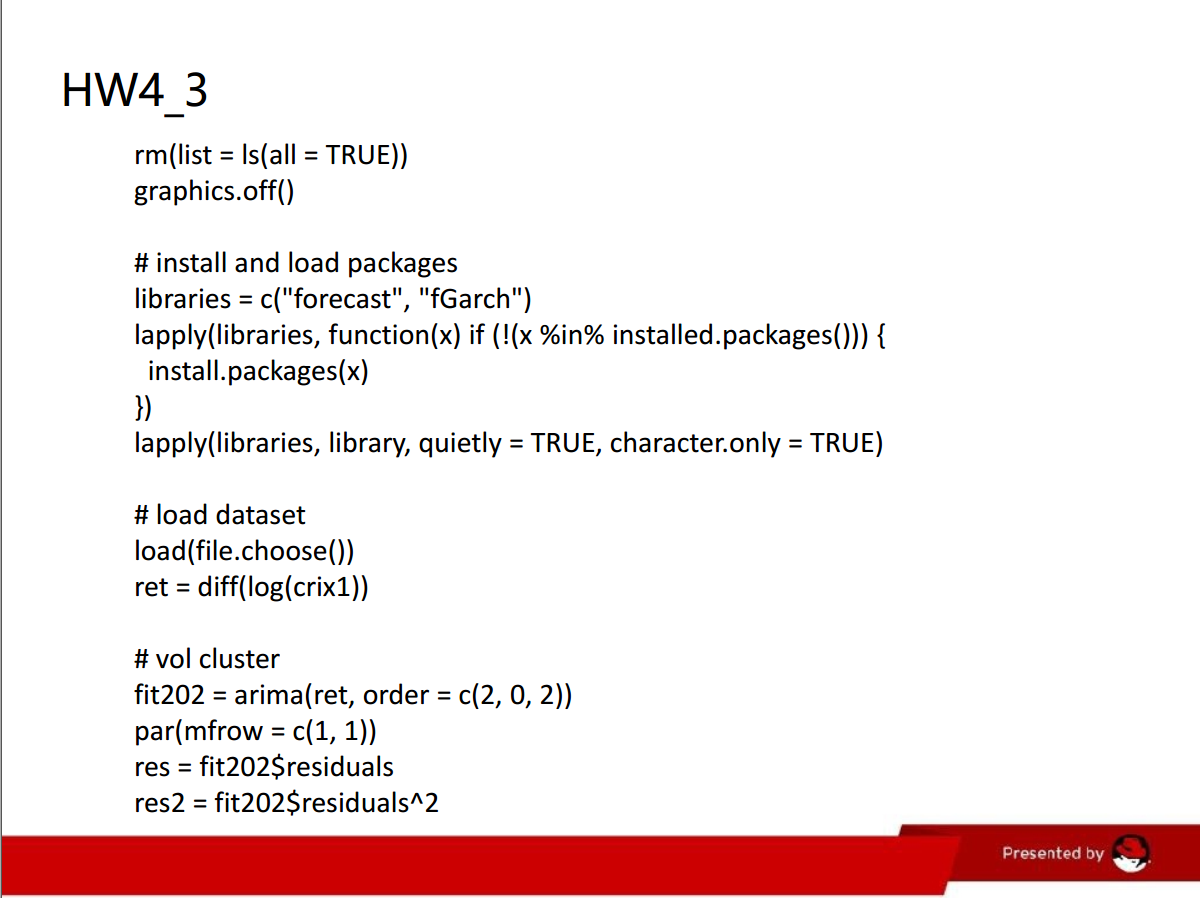
**(all this to be done on perfect PPTX slides)**

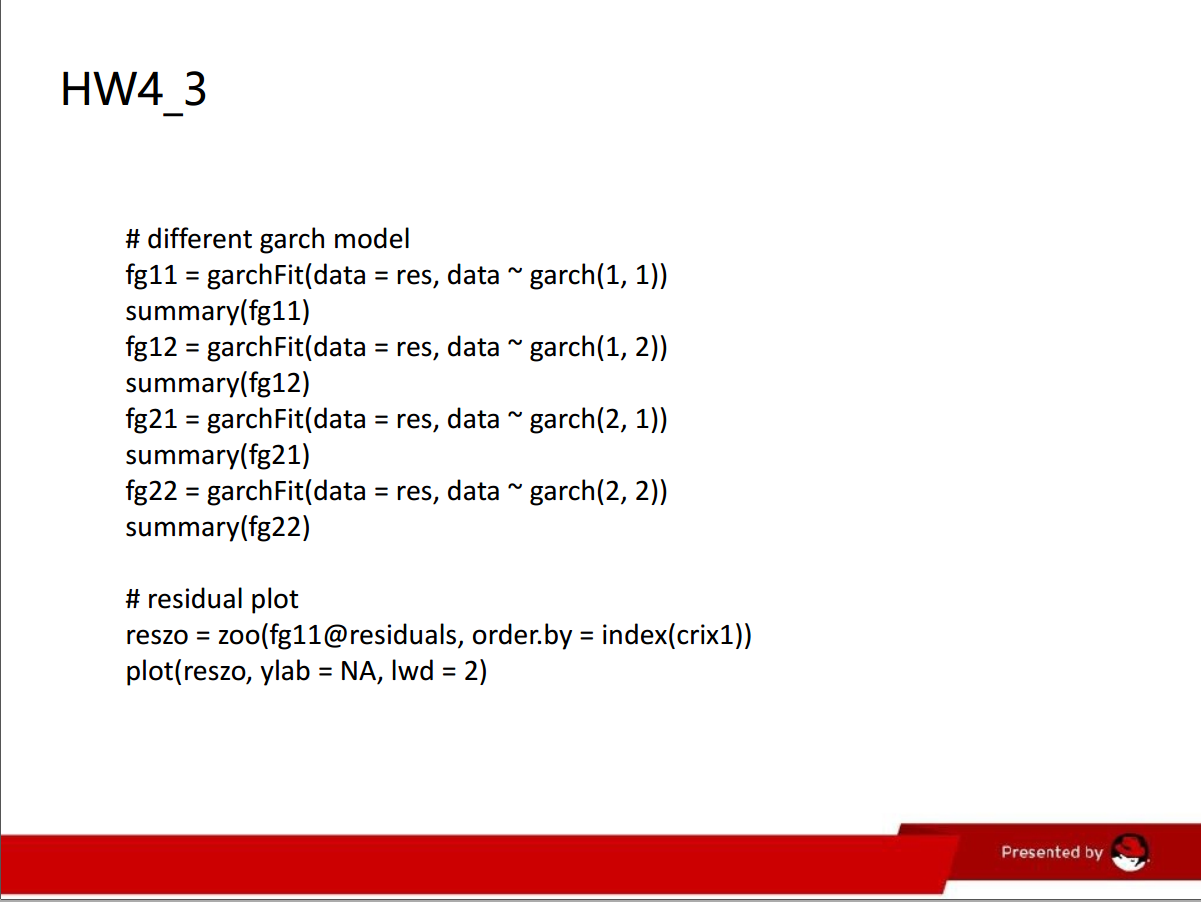


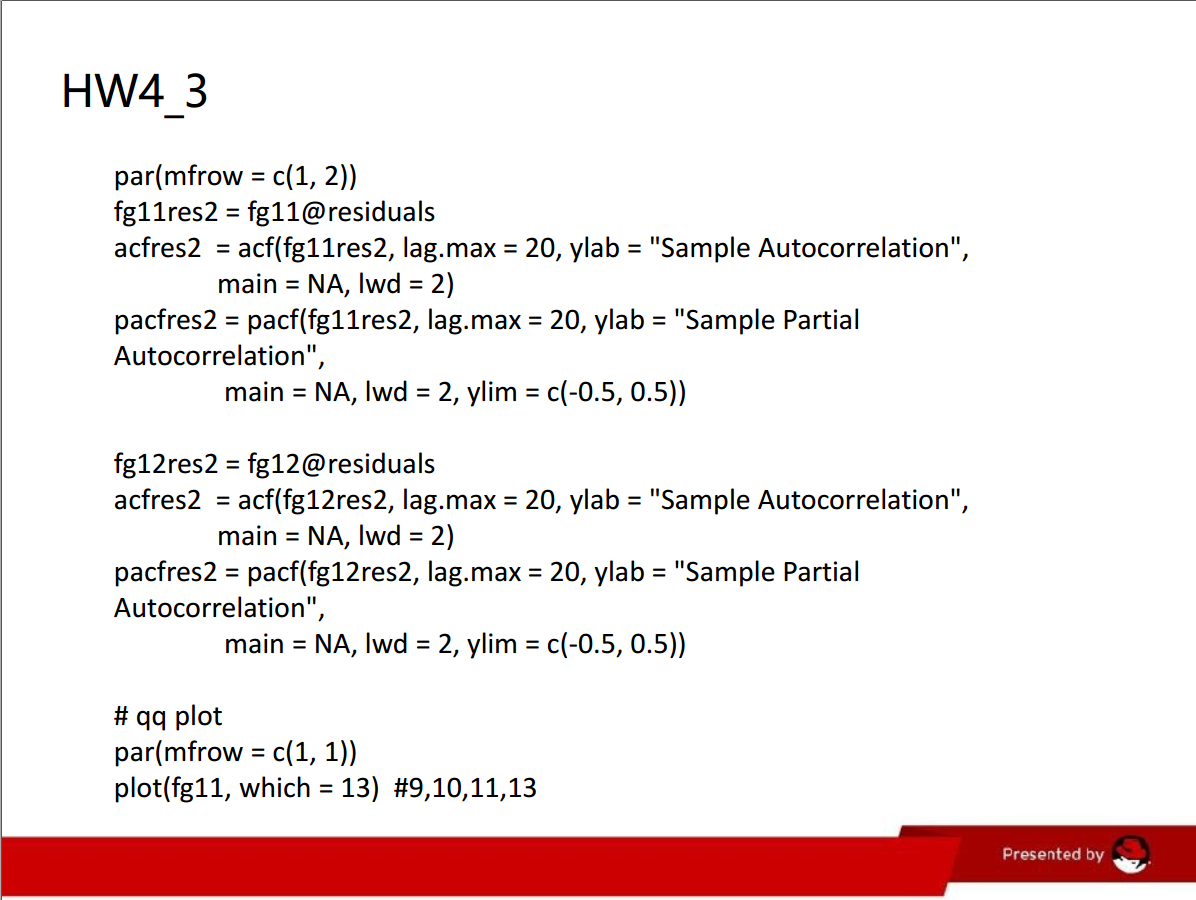


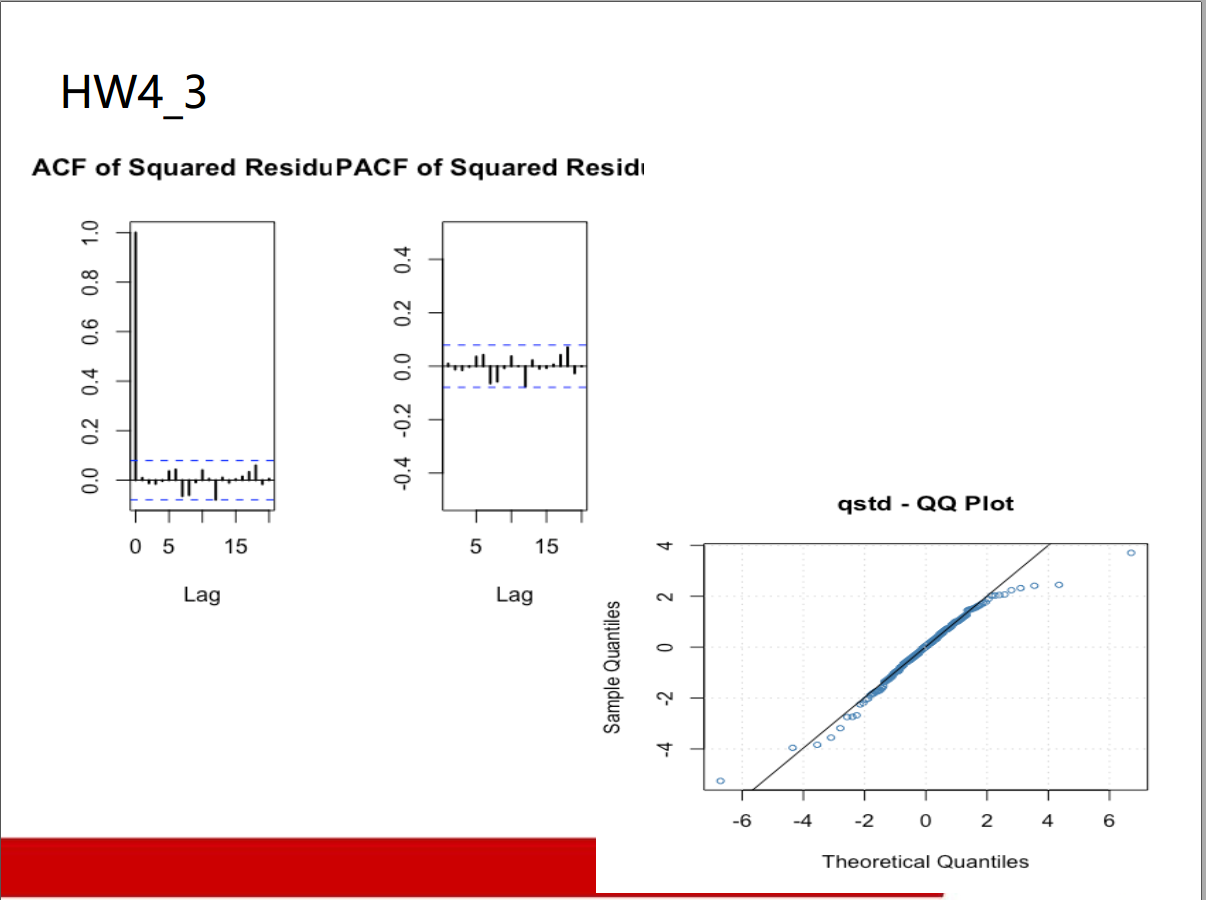


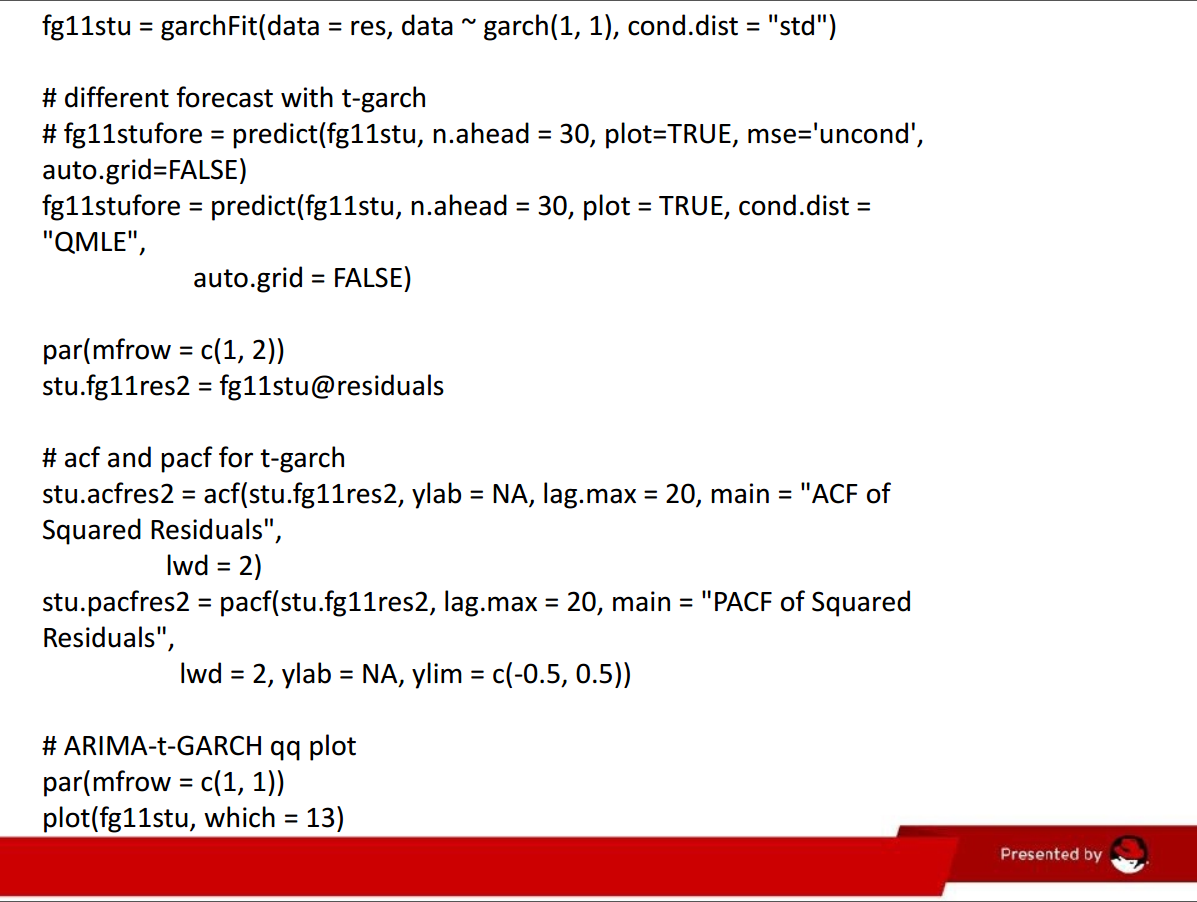












**Unit 5 HW**

**Q1**

rm(list = ls())

#install.packages("RCurl")

#install.packages("XML")

library(RCurl)

library(XML)

url1 = "http://shakespeare.mit.edu/romeo\_juliet/full.html"

url2 = "http://shakespeare.mit.edu/julius\_caesar/full.html"

url3 = "http://shakespeare.mit.edu/hamlet/full.html"

html1 = readLines(url1, encoding = "UTF-8")

html2 = readLines(url2, encoding = "UTF-8")

html3 = readLines(url3, encoding = "UTF-8")

html1 = htmlParse(html1, encoding = "UTF-8")

html2 = htmlParse(html2, encoding = "UTF-8")

html3 = htmlParse(html3, encoding = "UTF-8")

#install.packages("bitops")

#install.packages("stringr")

library(bitops)

library(stringr)

abs1 = lapply(url1, FUN = function(x) htmlParse(x, encoding = "Latin-1"))

abs2 = lapply(url2, FUN = function(x) htmlParse(x, encoding = "Latin-1"))

abs3 = lapply(url3, FUN = function(x) htmlParse(x, encoding = "Latin-1"))

clean\_txt = function(x) {

cleantxt = xpathApply(x, "//body//text()

[not(ancestor :: script)][ not(ancestor :: style)]

[not(ancestor :: noscript)] " ,xmlValue)

cleantxt = paste(cleantxt, collapse="\n")

cleantxt = str\_replace\_all(cleantxt, "\n", " ")

cleantxt = str\_replace\_all(cleantxt, "\r", "")

cleantxt = str\_replace\_all(cleantxt, "\t", "")

cleantxt = str\_replace\_all(cleantxt, "<br>", "")

return(cleantxt)

}

cleantxt1 = lapply(abs1,clean\_txt)

cleantxt2 = lapply(abs2,clean\_txt)

cleantxt3 = lapply(abs3,clean\_txt)

vec\_abs1 = unlist(cleantxt1)

vec\_abs2 = unlist(cleantxt2)

vec\_abs3 = unlist(cleantxt3)

###Text Mining

install.packages("tm")

install.packages("SnowballC")

library(tm)

library(SnowballC)

abs1 = Corpus(VectorSource(vec\_abs1))

abs2 = Corpus(VectorSource(vec\_abs2))

abs3 = Corpus(VectorSource(vec\_abs3))

abs\_dtm1 = DocumentTermMatrix(abs1, control = list(

stemming = TRUE, stopwords = TRUE, minWordLength = 3,

removeNumbers = TRUE, removePunctuation = TRUE))

abs\_dtm2 = DocumentTermMatrix(abs2, control = list(

stemming = TRUE, stopwords = TRUE, minWordLength = 3,

removeNumbers = TRUE, removePunctuation = TRUE))

abs\_dtm3 = DocumentTermMatrix(abs3, control = list(

stemming = TRUE, stopwords = TRUE, minWordLength = 3,

removeNumbers = TRUE, removePunctuation = TRUE))

##WordCloud

install.packages("ggplot2")

install.packages("wordcloud")

library(ggplot2)

library(wordcloud)

freq1 = colSums(as.matrix(abs\_dtm1))

freq2 = colSums(as.matrix(abs\_dtm2))

freq3 = colSums(as.matrix(abs\_dtm3))

wf1 = data.frame(word=names(freq1), freq=freq1)

wf2 = data.frame(word=names(freq2), freq=freq2)

wf3 = data.frame(word=names(freq3), freq=freq3)

#Romeo and Juliet

plot1 = ggplot(subset(wf1, freq>15), aes(word, freq1))

plot1 = plot1 + geom\_bar(stat="identity")

plot1 = plot1 + theme(axis.text.x=element\_text(angle=45, hjust=1))

freq1 = colSums(as.matrix(abs\_dtm1))

dark2\_1 = brewer.pal(6, "Dark2")

wordcloud(names(freq1), freq1, max.words=100, rot.per=0.2, colors=dark2\_1)

#Julius Caeser

plot2 = ggplot(subset(wf2, freq>15), aes(word, freq2))

plot2 = plot2 + geom\_bar(stat="identity")

plot2 = plot2 + theme(axis.text.x=element\_text(angle=45, hjust=1))

freq2 = colSums(as.matrix(abs\_dtm2))

dark2\_2 = brewer.pal(6, "Dark2")

wordcloud(names(freq2), freq2, max.words=100, rot.per=0.2, colors=dark2\_2)

#Hamlet

plot3 = ggplot(subset(wf3, freq>15), aes(word, freq3))

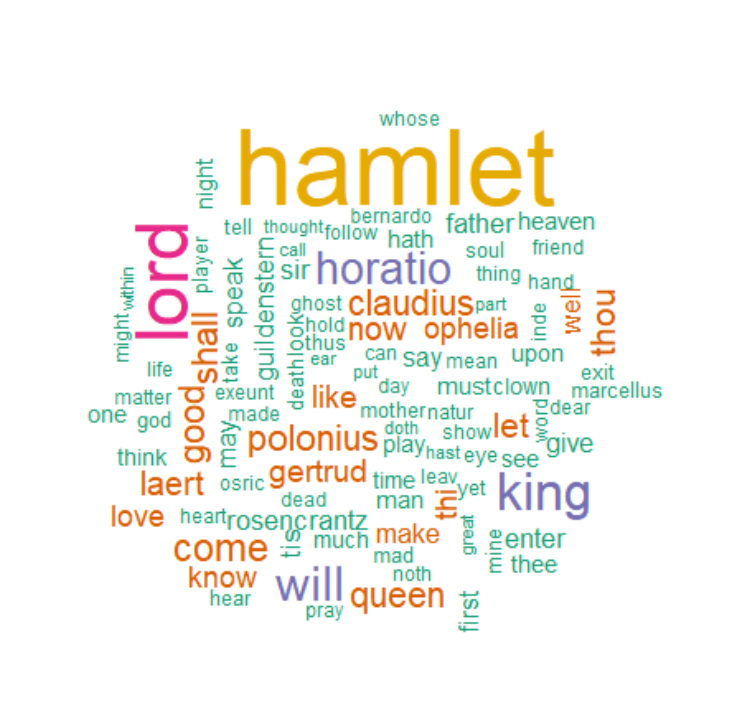
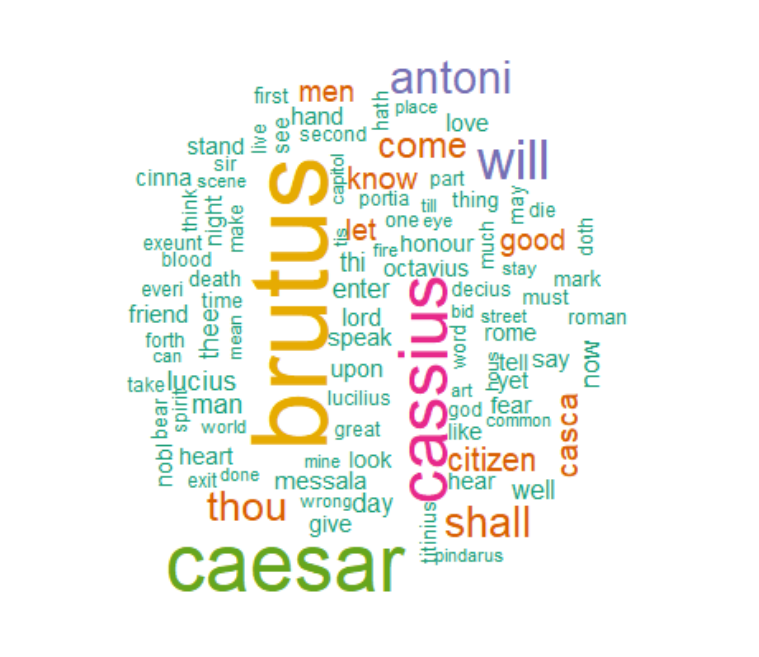
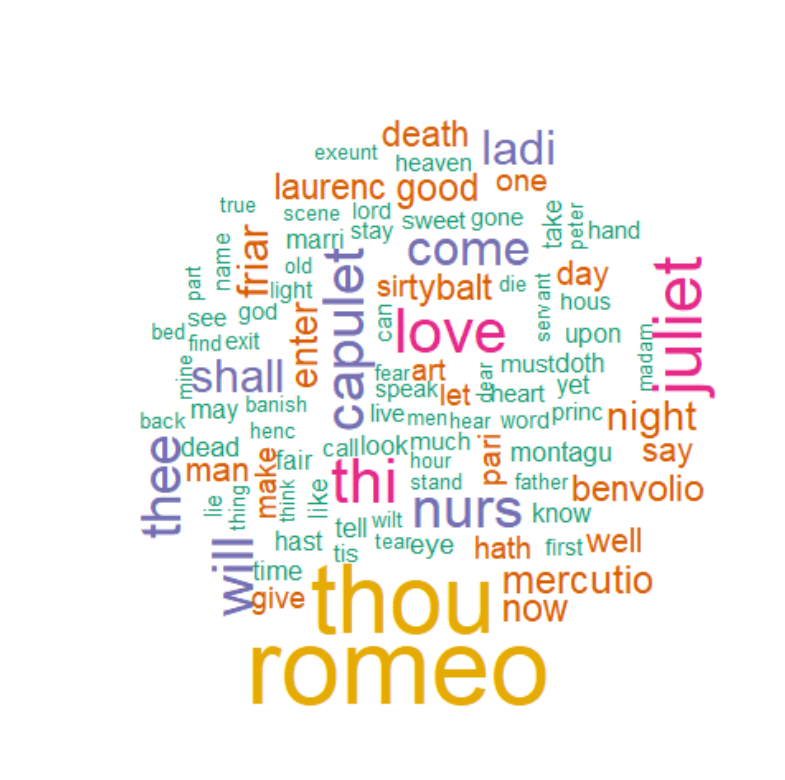
plot3 = plot3 + geom\_bar(stat="identity")

plot3 = plot3 + theme(axis.text.x=element\_text(angle=45, hjust=1))

freq3 = colSums(as.matrix(abs\_dtm3))

dark2\_3 = brewer.pal(6, "Dark2")

wordcloud(names(freq3), freq3, max.words=100, rot.per=0.2, colors=dark2\_3)



**Q2**

#Romeo and Juliet

wf1 <- wf1[order(-wf1$freq),]

wf1 <- wf1[c(1:20),]

p1 = ggplot(subset(wf1, freq > 15), aes(word, freq))

p1 = p1 + geom\_bar(stat = "identity")

p1 = p1 + theme(axis.text.x = element\_text(angle = 45, hjust = 1))

p1

#Julius Caeser

wf2 <- wf2[order(-wf2$freq),]

wf2 <- wf2[c(1:20),]

p2 = ggplot(subset(wf2, freq > 15), aes(word, freq))

p2 = p2 + geom\_bar(stat = "identity")

p2 = p2 + theme(axis.text.x = element\_text(angle = 45, hjust = 1))

p2

#Hamlet

wf3 <- wf3[order(-wf3$freq),]

wf3 <- wf3[c(1:20),]

p3 = ggplot(subset(wf3, freq > 15), aes(word, freq))

p3 = p3 + geom\_bar(stat = "identity")

p3 = p3 + theme(axis.text.x = element\_text(angle = 45, hjust = 1))

p3

