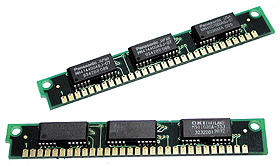
HW1

**The development of home computer**

**The root of the mountain -- SIMM memory**

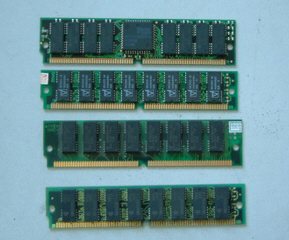
* Motherboard was first launched In 80286, memory chips used SIMM (Single In - lineMemory Modules, unilateral contact memory module) interface, the capacity of 30 pin, 256 KB, must be by 8 pieces of data and 1 piece of parity of a bank, because of this, 30 pin simms we see four are used together.Since the PC entered the civil market in 1982, the 30pin SIMM memory with the 80286 processor is the ancestor of the memory field

**The root of the mountain – 30Pin SIMM memory**



**Hovered -- EDO DRAM memory**

* EDO DRAM (Extended the Date Out of RAM, and the expansion of data model storage) memory, this is between 1991 to 1995 in memory chips, EDO - very similar to RAM with FP DRAM, it canceled the extension data output memory memory and transmission time interval between two storage cycle, at the same time of the data sent to the CPU to access to the next page, and 15-30% faster than ordinary DRAM.



* From 1991 to 1995 years, let's see an embarrassing situation, that is memory technology develops slowly over the past few years, almost stagnant, so at this point we see EDO RAM 72 pin, 168 pin and coexist, in fact EDO memory also belongs to the category of the 72 pin SIMM memory, but it USES a new way of addressing.



**Generational classic - SDRAM memory**

* SDRAM memory is developed from the early stage of 66MHz and developed later with 100MHz and 133MHz
* ****

**Rambus DRAM memory**

* In order to achieve the exclusive market, Intel and Rambus promote Rambus DRAM memory (called RDRAM memory) in the PC market.Unlike SDRAM, its simple adopted a new generation of high speed memory architecture, based on the kinds of RISC (Reduced Instruction Set Computing, Reduced Instruction Set computer) theory, this theory can reduce the complexity of the data, make the whole system performance improved.

****

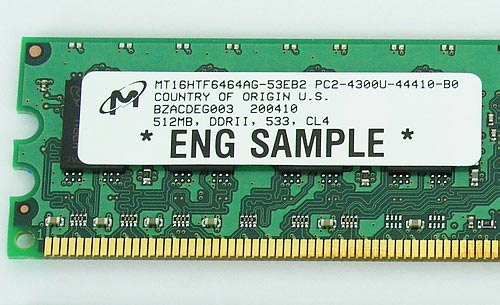
**Repeat the classic - DDR memory**

* DDR memory as a compromise between performance and cost solution, its purpose is to rapidly establish a solid market space, then rapid advances in frequency step by step, finally make up for the lack of memory bandwidth. First generation DDR200 specification has not gained popularity, the second generation PC266 DDR SRAM (133 MHZ clock x 2 x = 266 MHZ bandwidth) data transmission by PC133 SDRAM memory are derived, which brings the DDR memory to the first climax, there are still many cy young and K7 processors AMD are used DDR266 specifications of the memory, the subsequent DDR333 memory also belong to a kind of excessive, but DDR400 memory become the mainstream platform of choice at present, dual channel DDR400 memory has become the basic standard of the FSB 800 processor is tie-in, the subsequent DDR533 specification becomes overclocking user selection object.



**Today's star - DDR2 memory**

* DDR2 can provide a minimum of 400MB/s bandwidth per pin on the 100MHz frequency base, and its interface will run on 1.8V voltage, which further reduces heat output to improve the frequency. In addition, DDR2 will incorporate new performance indicators and interrupt instructions such as CAS, OCD, ODT, and increase the utilization of memory bandwidth. According to the DDR2 standard set out by JEDEC organizers, DDR2 memory in markets such as PCS will have a different clock frequency, such as 400, 533, 667MHz, and so on (see figure 16).High-end DDR2 memory will have 800, 1000MHz frequency. DDR2 memory will be packaged in FBGA with 200 -, 220 -, 240- stitches. The initial DDR2 memory will be produced with a production process of 0.13 micron, with a voltage of 1.8v and a capacity density of 512MB.

****

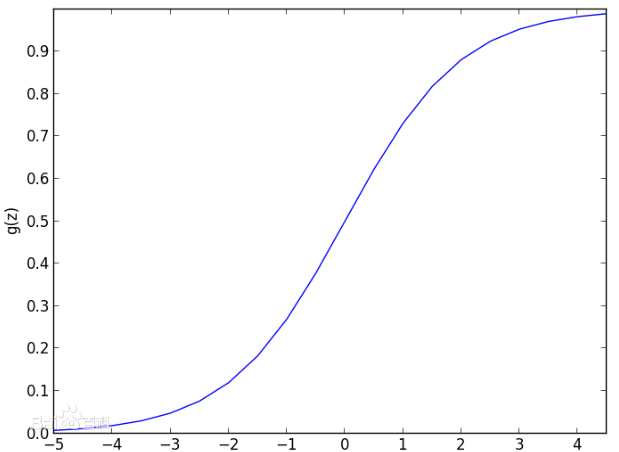
**logistic regression**

**The Introduction of logistic regression**

* In statistics, 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

**The Application of logistic regression**

* First, look for risk factorsto identify the risk factors for a disease.
* second, prediction, if we have established the logistic regression model, we can predict in the case of different independent variables, the probability of a disease or a certain situation.
* Third, discrimination, similar to prediction, according to the logistic model, judge the probability of a person's situation belongs to a certain desease.
* The test data is X（x0，x1，x2…xn），Θ（θ0，θ1，θ2…θn）is the parameter.
* Then Z= θ0x0 + θ1 x1  
  +· · ·+
* Transfer Z into two-valued variable
* g(Z)=1/(1+𝑒^(−𝑧) )

****

**HW2**

**Q1**

# clear variables and close windows

rm(list = ls(all = TRUE))

graphics.off()

# set working directory

setwd("")

# install and load packages

packages = c("lmtest", "sandwich")

invisible(lapply(packages, function(pkg) {

if (!is.element(pkg, installed.packages())) install.packages(pkg)

library(pkg, character.only = TRUE)

}))

# read data

cpu.df = read.csv2("cpu.csv")

# new column with year as integer

cpu.df$year.int = cpu.df$Date - min(cpu.df$Date)

# regression

reg.lm = lm(log10(cpu.df$Transistors) ~ cpu.df$year.int)

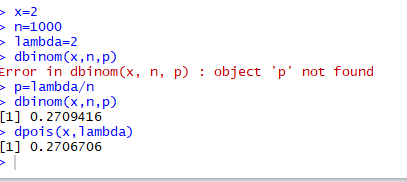
# coefficients

coeftest(reg.lm, vcov. = vcovHC)

# R^2

paste("R^2:", summary(reg.lm)$r.squared)

**Q3**

****

**HW3**

Answer 1

install.packages("digest")

library("digest")

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

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

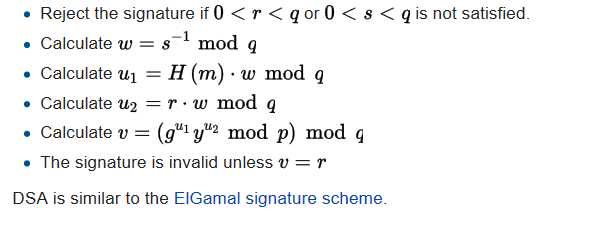
Answer2

DSA:

* Let H be the hashing function and m the message
* Generate a random per-message value where 1 < k < q
* Calculate r = ( g k mod p ) mod q
* In the unlikely case that r = 0, start again with a different random k
* Calculate s = k − 1 ( H ( m ) + x r ) mod q
* In the unlikely case that s = 0, start again with a different random k
* Sensitivity

1. With DSA, the entropy, secrecy, and uniqueness of the random signature value k are critical. It is so critical that violating any one of those three requirements can reveal the entire private key to an attacker.[11] Using the same value twice (even while keeping k secret), using a predictable value, or leaking even a few bits of k in each of several signatures, is enough to reveal the private key x.[12]
2. This issue affects both DSA and ECDSA – in December 2010, a group calling itself fail0verflow announced recovery of the ECDSA private key used by Sony to sign software for the PlayStation 3 game console. The attack was made possible because Sony failed to generate a new random k for each signature.[13]
3. This issue can be prevented by deriving k deterministically from the private key and the message hash, as described by RFC 6979. This ensures that k is different for each H(m) and unpredictable for attackers who do not know the private key x.
4. In addition, malicious implementations of DSA and ECDSA can be created where k is chosen in order to subliminally leak information via signatures. For example an offline private key could be leaked from a perfect offline device that only released innocent-looking signatures.[14]

Verifying:



Answer3

R-code:

>library(RJSONIO)

> 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))

Answer3

Answer 4

rm(list = ls(all = TRUE))

graphics.off()

# install and load packages #

libraries = c("zoo", "tseries")

lapply(libraries, function(x) if (!(x %in% installed.packages())) {install.packages(x)})

lapply(libraries, library, quietly = TRUE, character.only = TRUE)

# load dataset #

load(file = "F/wise/big data/crix.RData")

ret = diff(log(crix))

# d order #

Box.test(ret, type = "Ljung-Box", lag = 20)

# stationary test #

adf.test(ret, alternative = "stationary")

kpss.test(ret, null = "Trend")

par(mfrow = c(1, 2))

# acf plot #

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

# LB test of linear dependence #

print(cbind(autocorr$lag, autocorr$acf))

Box.test(ret, type = "Ljung-Box", lag = 1, fitdf = 0)

Box.test(autocorr$acf, type = "Ljung-Box")

# plot of pacf #

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

print(cbind(autopcorr$lag, autopcorr$acf))

# arima model#

par(mfrow = c(1, 1))

auto.arima(ret)

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

}

}

aic

# 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

}

}

bic

# 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))

tsdiag(fit202)

tsdiag(fit4)

tsdiag(fitr4)

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

fit202 = arima(ret, order = c(2, 0, 2))

crpre = predict(fit202, n.ahead = 30)

dates = seq(as.Date("02/08/2014", format = "%d/%m/%Y"), by = "days", length = length(ret))

plot(ret, type = "l", xlim = c(0, 644), ylab = "log 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)

# Produces GARCH estimation results using ARIMA model residuals

rm(list = ls(all = TRUE))

graphics.off()

# install and load packages

libraries = c("FinTS", "tseries", "forecast", "fGarch")

lapply(libraries, function(x) if (!(x %in% installed.packages())) {

install.packages(x)

})

lapply(libraries, library, quietly = TRUE, character.only = TRUE)

# load dataset

load(file = "C:/Users/xiumei/Desktop/big data/crix.RData")

ret = diff(log(crix1))

# vol cluster

fit202 = arima(ret, order = c(2, 0, 2))

par(mfrow = c(1, 1))

res = fit202$residuals

res2 = fit202$residuals^2

# different garch model

fg11 = garchFit(data = res, data ~ garch(1, 1))

summary(fg11)

fg12 = garchFit(data = res, data ~ garch(1, 2))

summary(fg12)

fg21 = garchFit(data = res, data ~ garch(2, 1))

summary(fg21)

fg22 = garchFit(data = res, data ~ garch(2, 2))

summary(fg22)

# residual plot

reszo = zoo(fg11@residuals, order.by = index(crix1))

plot(reszo, ylab = NA, lwd = 2)

par(mfrow = c(1, 2))

fg11res2 = fg11@residuals

acfres2 = acf(fg11res2, lag.max = 20, ylab = "Sample Autocorrelation",

main = NA, lwd = 2)

pacfres2 = pacf(fg11res2, lag.max = 20, ylab = "Sample Partial Autocorrelation",

main = NA, lwd = 2, ylim = c(-0.5, 0.5))

fg12res2 = fg12@residuals

acfres2 = acf(fg12res2, lag.max = 20, ylab = "Sample Autocorrelation",

main = NA, lwd = 2)

pacfres2 = pacf(fg12res2, lag.max = 20, ylab = "Sample Partial Autocorrelation",

main = NA, lwd = 2, ylim = c(-0.5, 0.5))

# qq plot

par(mfrow = c(1, 1))

plot(fg11, which = 13) #9,10,11,13

# kp test

set.seed(100)

x = rnorm(200)

# Do x and y come from the same distribution?

ks.test(x, fg11@residuals**)**

**HW4**

rm(list = ls(all = TRUE))

graphics.off()

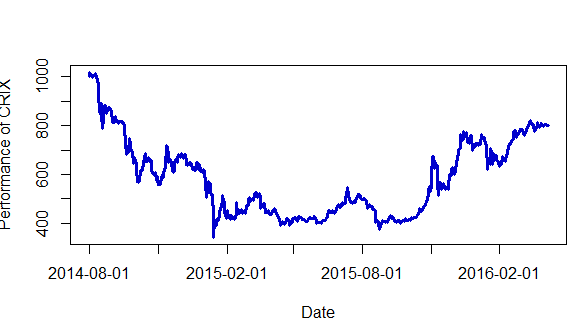
##

load("crix.RData")

plot(crix, type = "l", col = "blue3", xaxt = "n", lwd = 3, xlab = "Date",

ylab = "Performance of CRIX")

axis(1, at = c(2,94,186,275,367,459,551), label = names(crix)[c(2,94,186,275,367,459,551)])



rm(list = ls(all = TRUE))

graphics.off()

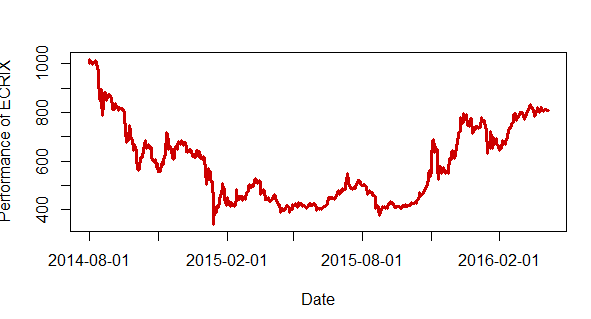
##

load("ecrix.RData")

plot(ecrix, type = "l", col = "red3", xaxt = "n", lwd = 3, xlab = "Date",

ylab = "Performance of ECRIX")

axis(1, at = c(2,94,186,275,367,459,551), label = names(ecrix)[c(2,94,186,275,367,459,551)])

****

rm(list = ls(all = TRUE))

graphics.off()

##

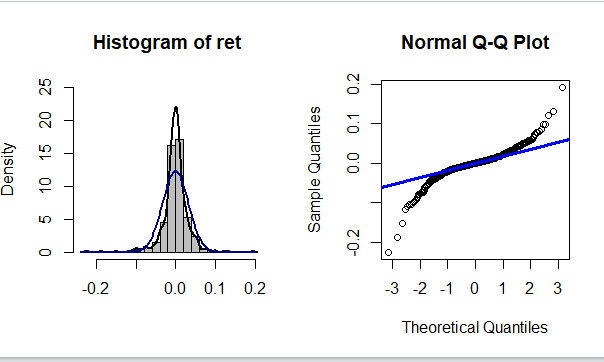
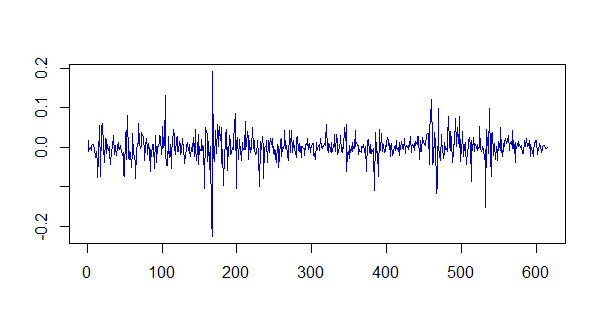
load("efcrix.RData")

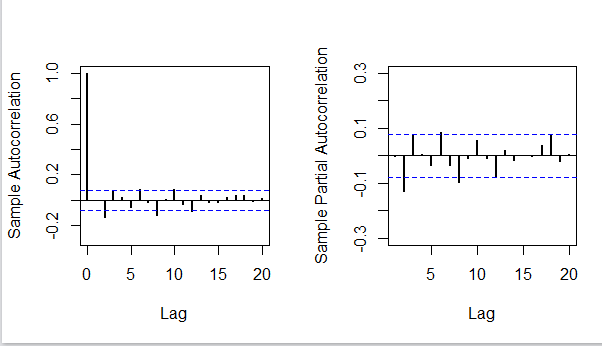
plot(efcrix, type = "l", col = "yellow3", xaxt = "n", lwd = 3, xlab = "Date",

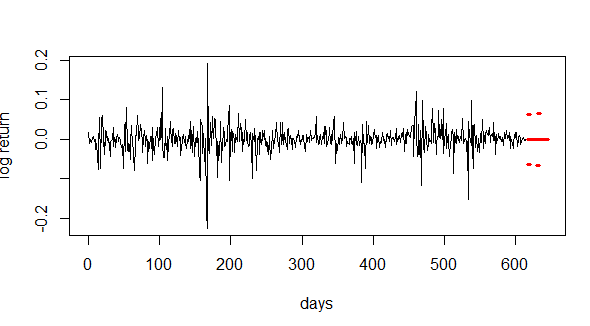
ylab = "Performance of EFCRIX")

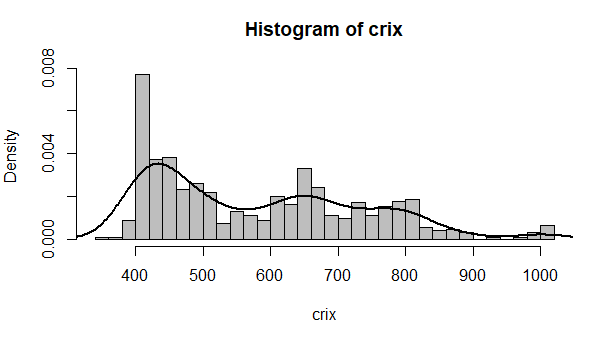
axis(1, at = c(2,94,186,275,367,459,551), label = names(efcrix)[c(2,94,186,275,367,459,551)])

****

****

****

****

**rm(list = ls(all = TRUE))**

**graphics.off()**

**load(file = "crix.RData")**

**hist(crix, col = "grey", breaks = 40, freq = FALSE)**

**lines(density(crix), lwd = 2)**

**par(mfrow = c(1, 2))**

**HW5**

**#HW5.1**

**library(RCurl)**

**library(XML)**

**library(bitops)**

**library(stringr)**

**url="http://www.romeo-and-juliet.org/Romeo-and-Juliet-Script/romeo-and-juliet-play-script-ACT-4-SCENE-1-Friar-Laurences-cell.htm"**

**abs= lapply(url, 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)**

**}**

**cleantxt = lapply(abs,clean\_txt)**

**vec\_abs = unlist(cleantxt)**

**install.packages("NLP")**

**library(tm)**

**library(SnowballC)**

**abs = Corpus(VectorSource(vec\_abs))**

**abs\_dtm = DocumentTermMatrix(abs, control = list(**

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

**removeNumbers = TRUE, removePunctuation = TRUE))**

**dim(abs\_dtm)**

**nstall.packages("wordcloud")**

**library(ggplot2)**

**library(wordcloud)**

**library(RColorBrewer)**

**freq = colSums(as.matrix(abs\_dtm))**

**wf = data.frame(word=names(freq), freq=freq)**

**plot = ggplot(subset(wf, freq>15), aes(word, freq))**

**plot = plot + geom\_bar(stat="identity")**

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

**plot**

**freq = colSums(as.matrix(abs\_dtm))**

**dark2 = brewer.pal(6, "Dark2")**

**wordcloud(names(freq), freq, max.words=100, rot.per=0.2, colors=dark2)**

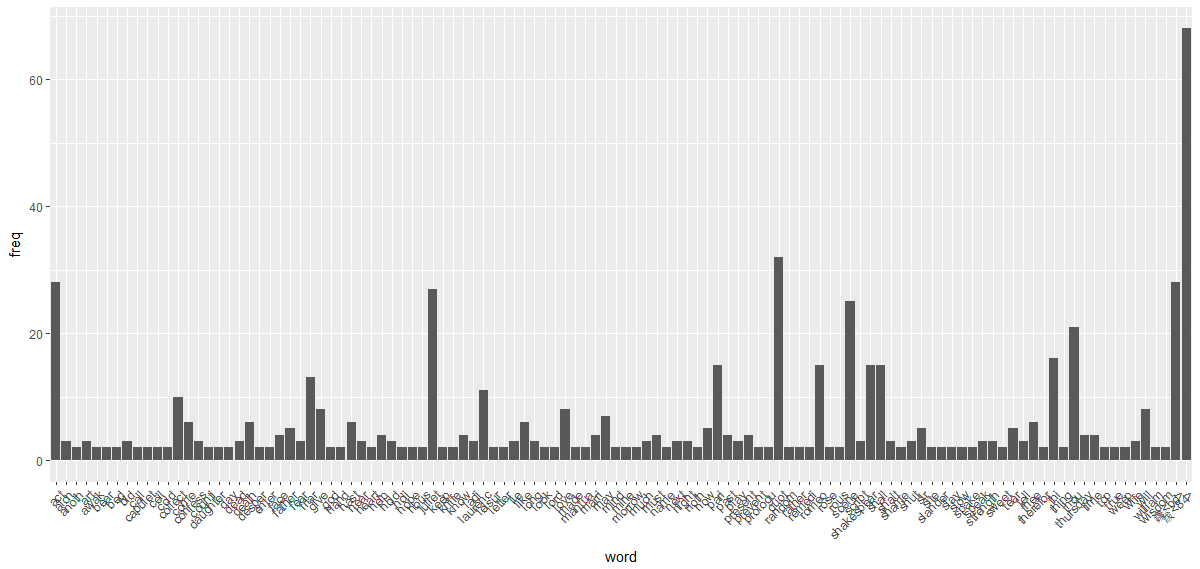
**#HW5.2**

**plot = ggplot(subset(wf, freq>1), aes(word, freq))**

**plot = plot + geom\_bar(stat="identity")**

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

**plot**

****