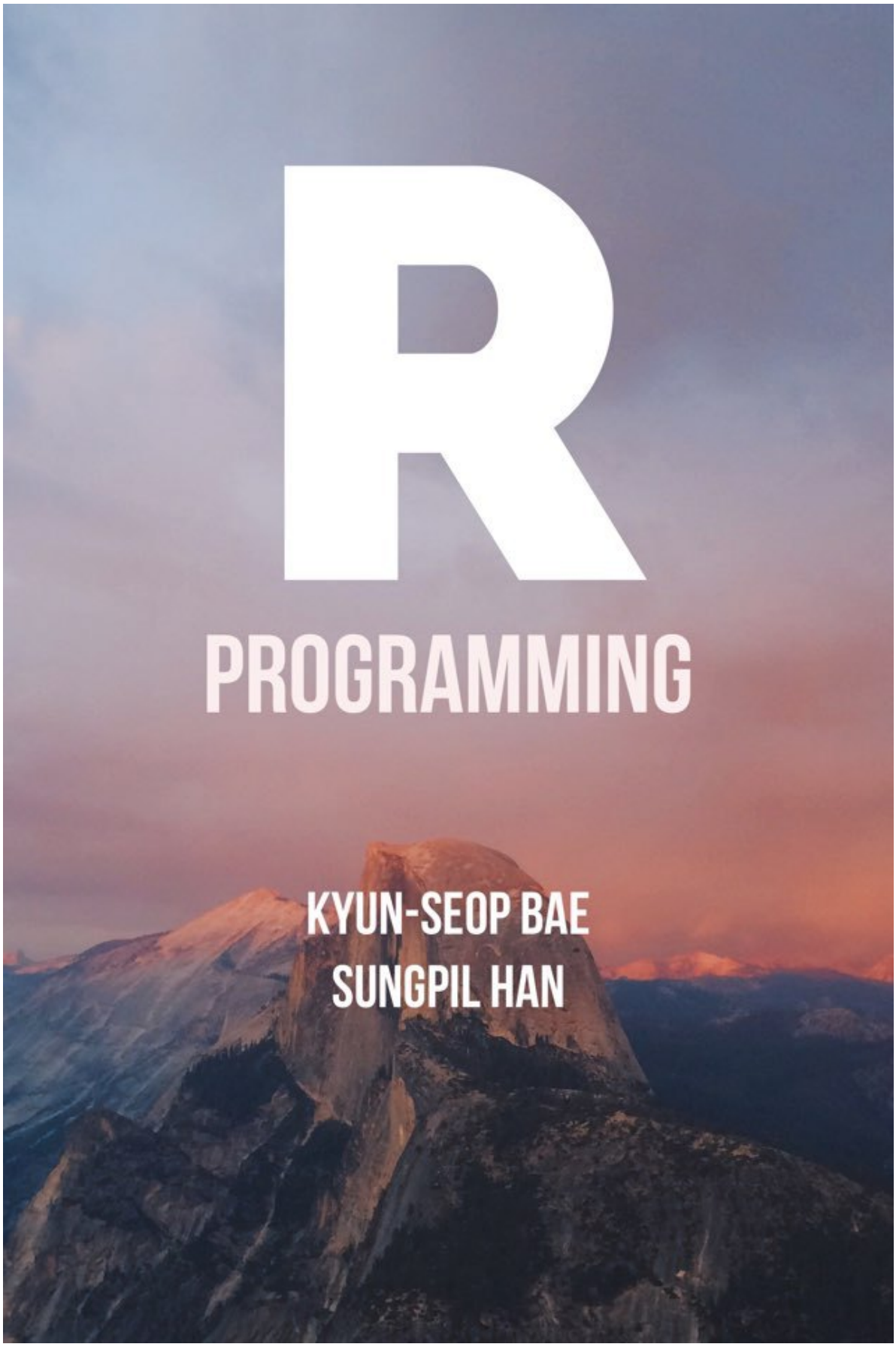
Kyun-Seop Bae, Sungpil Han

R Programming: Lecture Notes


R
PROGRAMMING
KYUN-SEOP BAE
SUNGPIL HAN

Contents

List of Tables

List of Figures

Preface

안녕하십니까?

2017년 1학기 울산대학교 의학과 대학원 수업 `R Programming` 과목 담당교수 배균섭입니다.

R은 http://cran.r-project.org 에서 다운로드받아 설치할 수 있습니다. 역시 같은 사이트에서 Manual이 나와 있으니 참고하시기 바랍니다. 구글에서 'R Programming pdf' 와 같은 키워드로 검색하시면 많은 자료를 보실 수 있습니다.

첨부한 R.stx[1] 파일은 AcroEdit이라는 editor에서 사용할 syntax highlighting용 구문 파일입니다. http://www.acrosoft.pe.kr 에서 다운로드 받아 설치하시기 바랍니다. AcroEdit 대신 notepad++를 선호하시는 분은 그대로 사용하셔도 됩니다.

저는 RStudio, tinnR 등을 이용해서 강의하지 않습니다만, 필요하신 분은 쓰셔도 괜찮습니다. 향후 R package 작성을 위해서는 MiKTeX와 Rtools를 설치하십시오.

추가로 말씀드리자면, http://www.coursera.org 에 많은 R 강좌가 개설되어 있습니다. Specialization course로 들어가면 유로이지만, (Specialization course는 여러 개의 과목이 합쳐져 있는 것입니다.) 개별 과목을 검색해서 들어가면, 무료로도 볼 수 있습니다. (대신 시험을 칠 수 없거나, certificate를 받을 수 없습니다.)

좋은 강좌가 많으니 많이 활용하시기 바립니다.

강의 장소에 불편함이 많은 것으로 생각되어, 다음과 같이 Skype 모임을 개설하였습니다. 사정상 원거리에서 오시기 불편한 분들은 활용하시기 바랍니다. 출석은 화면을 캡쳐하거나 휴대폰으로 찍은 뒤 `sec@acp.kr`, `shan@acp.kr` 로 보내주시면 출석으로 인정해 드립니다.

Skype 모임 참가 https://meet.lync.com/uucp-acp/ksbae/SKGJ3BNQ

2017년 3월, 배균섭 배상

[1] https://groups.google.com/a/acr.kr/group/r/attach/409db97bf453a/R.stx?part=0.1&authuser=0

[2] http://creativecommons.org/licenses/by-nc-sa/4.0/

FIGURE 1: Creative Commons License

Teaching Assistant

안녕하십니까? 서울아산병원 임상약리학과 전공의 한성필입니다. 수업과 관련된 여러 제반 업무를 담당하고 있습니다. 언제든 의문사항 있으면 `r@acr.kr` 로 전체 메일 보내시거나 교수님 `k@acr.kr` 혹은 제 개인 메일 `shan@acp.kr` 로 연락해 주십시오.

교수님께서 세우신 방침에 따라 수업시간에 출석을 부르지 않을 예정입니다. 수강하시는 화면(Skype)을 휴대폰으로 사진 찍으시거나 강의실의 스크린을 사진으로 촬영하셔서 `sec@acp.kr` / `shan@acp.kr` 로 동시에 보내주시면 됩니다. 가급적 "2017-03-31 한성필 출석"과 같은 식의 제목을 유지해 주시면 처리하는데 큰 도움이 될 것 같습니다.

출석 체크를 위해 전체메일을 사용하지 말아주십시오!

아울러 수업 중에 사용한 코드/스크립트를 사용하여 R의 패키지인 `bookdown`을 사용해 웹북을 제작 중에 있습니다. (Xie, 2017) 여러분이 읽고 있는 이 책 자체가 R 코드의 일종인 `Rmarkdown`의 결과물이라고 보시면 됩니다. Github 저장소[3]가 있으니 소스 코드를 보실 수 있습니다. 누구나 소스를 편집하여 `Pull Request`를 요청할 수 있으므로 혹시 Github를 사용하셔서 웹북의 질을 높이고자 하시는 수강생 선생님들께서는 도움을 주십시오.

감사합니다.

2017년 3월, 한성필 올림

FAQ

접속 관련

> Q. 스카이프를 한번도 안써봐서 이참에 사용법을 배우고있는데, 수업시작 시에 상대방을 어떻게 검색해서 들어가면 될지 알려주시면 감사하겠습니다.

[3] https://github.com/asancpt/Rprogramming

Q. 온라인 수강시 접속하는 스카이프 주소는 무엇인지요?

---

https://meet.lync.com/uucp-acp/ksbae/SKGJ3BNQ

Chrome 등 웹브라우저에서 위 주소를 입력하면 직접 대화방으로 연결됩니다. (검색할 필요 없습니다.) 처음 설치시에는 Add-on이 설치될 수 있습니다. MacOS Sierra, Win7, Win10에서 Chrome, Internet Explorer 등을 사용하여 테스트해 보았고 모두 잘 동작하였습니다. 대부분의 경우 Skype For Business 계정이 없을 것으로 생각되는데 따로 로그인할 필요 없습니다.

수업 시작 30분 전부터 대화방을 개설해 놓도록 하겠습니다.

https://groups.google.com/a/acr.kr/d/msg/r/nUkrE37W2kQ/waG-FkM_BgAJ 교수님께서 처음 보낸 메일을 참고해 주십시오.

Q. 앞으로 수업은 지난 첫수업처럼 계속 온라인 수강이 가능한 것인가요?

네, 계속 온라인으로 가능합니다.

Q. 저도 웹캠을 설치하여야 하여야 하나요?

설치할 필요 없습니다. 오히려 수강자의 웹캠의 전원을 꺼두시길 권고드립니다.

Q. 수강전 온라인 강의 테스트 해볼 수 있나요?

수업 시작 30분 전부터 대화방을 개설하여 놓도록 하겠습니다.

출석관련

> Q. 미국학회 참석으로 수업시간이 귀국행 비행기 기내에 있을거같아 출석이 안될것 같습니다. 방법이 있을지요?

결석 사유서를 제출해 주시면 출석 처리 하겠습니다. 대학원 홈페이지 참고 바랍니다.[4] 이 링크로 들어가시면 가장 위에 있습니다. (결석사유서.hwp) 참고로 수업 영상은 녹화하여 Youtube에 비공개 링크를 만들 예정이라서 추후에 관련 영상을 시청할 수 있을 것 같습니다. 결석사유서를 제출한다고 100% 출석이 인정되는 것은 아닙니다. 이것이 기본적으로는 offline강의이기 때문에 강의시간에 강의실에 있든지, 또는 온라인으로 접속해 있어야 합니다. 출석사유서를 제출하거나, 추후 동영상 시청을 해서 그 증거(사진)을 제출하는 경우에 감점을 줄여드릴 수 있습니다. 예를 들어, 결석시에는 2점 감점인데, 결석사유서를 제출하면 1점만 감점한다는지, 동영상을 보면 0.5점만 감점한다는지 하는 것입니다. 결석 사유서 제출 시 출석 처리 원칙에 대한 설명을 드리오니, 참고하시길 바랍니다.

과제 관련

> Q. 과제물이 있다고 들었는데 언제 assign하게 되는지요?

과제물은 빨라야 5주차 이후에 나갑니다.

> 수업계획서가 변경되었다고하셨는데, 과정을보니 시험을 몇째주에 보는지 기재되어있지 않아서요, 성적평가에는 중간고사, 기말고사, 과제가 모두 적혀있는데 어떤것이 맞는 것 일까요?

[4] http://www.medulsan.ac.kr/graduate/?mid=72&curpage=files

중간 기말 고사는 따로 없습니다. 대신 강의를 합니다. (수업계획서 참고)

Coursera 관련

> Q. 첫 수업 때, certification 관련 말씀을 하셨는데, 정확히 coursera 사이트에서 어떤 것을 듣고, 제출을 해야하는지 궁금합니다. (비슷한 내용이 많아, 어떤것을 들어야하는지 헷갈립니다.)

Coursera는 꼭 어느 것을 들어야 하는 것은 아니고, R programming과 관련된 것이라면 자유로이 골라서 들으면 됩니다. 대표적인 두 가지만 들자면 다음과 같습니다.

- https://www.coursera.org/learn/r-programming
- https://www.coursera.org/learn/r-programming-environment

> Q. Coursera 강의를 듣고 증명서를 내면 출석을 얼마나 커버할 수 있을런지요?

Coursera는 출석 커버보다는 grade를 올려 주기 위한 것입니다. 출석은 Skype로 커버해야 합니다. 출석의 성적 반영비율은 25%이지만, 규정상 4회 이상 결석이면 성적이 나갈 수 없습니다.

Syllabus

2017-04-10 현재 개정된 수업계획서입니다.

2017-1학기 수업계획서(Course Outline)

년도-학기 (year-semester)	2017-1	과목명 (course name)	R 프로그래밍 R Programming		
과목번호-분반 (courseNo-classNo)	WA5493 - 01	학점(강의-실습) (credit)	3학점(3-0)	ABEEK(설계학점) (Abeek credit)	
담당교수 (professor)	배균섭 Bae, Kyun Seop	연구실 번호 (office phone)	02-3010-4611	학부(과)사무실 (Dept. office phone)	02-3010-4217
개설학과-학년 (department-year)	의학과			이수구분 (type of course requirement)	공통 common
E-MAIL	ksbae@amc.seoul.kr	강좌구분 (type of lecture)	일반강좌 general lecture		
홈페이지 (Homepage)		성적평가방법(method of grade evaluation)	절대평가 / 등급 absolute evaluation		

1.교과목 개요(course description)

Data Science의 가장 기본적인 tool인 R로 어떻게 프로그래밍을 하는지에 기술을 익힌다. 자신의 세부 전공분야에 상관없이 적용할 수 있는 공통적인 부분을 중심으로 학습할 것이다. Students will learn how to program using R which is the basic tool for the data science. The subjects will be common ones regardless of their specific majors.

2.교수목표(goal of instruction)

1.R을 이용하여 자신의 문제를 해결할 수 있다.
2.R package를 개발하여 다른 사람의 문제 해결을 돕는다.
3.정답이 알려져 있지 않는 문제에 대하여, 체계적인 시행착오로 최적의 해답을 찾는 습관을 들인다.
4.자신이 가진 Tool들의 한계와 장단점을 이해하고 자신의 문제에 적용한다.

3.주요 학습내용 및 수업진행방법(main contents & methods of teaching)

강의, 동영상 시청

4.학습 성과 평가방법(evaluation criteria)

과제 (중간, 기말고사 대신 과제를 제출해야 하며, 중간, 기말고사 기간에도 강의가 있습니다.)

평가항목(evaluation	출석 (attendance)	중간고사(mid term exam)	기말고사 (final exam)	리포트 (report)	발표(presentation)	퀴즈 (quiz)	Term Project	기타 (etc.)
성적반영 비율(percentage)	25.00	25.00	25.00	25.00	0.00	0.00	0.00	0.00

5.교재 및 참고 문헌(textbook & reference books)

1. [부교재] [도서관] (데이터 고급 분석과 통계 프로그래밍을 위한) 빅데이터 분석 도구 R 프로그래밍 노만 매트로프 지음 옮김 의왕 : 에이콘 9788960773332; 9788960772793(set)
2. [부교재] [도서관] R Cookbook : 데이터 분석과 통계, 그래픽스를 위한 실전 예제 폴 티터 지음;이제원 옮김 서울: 인사 9788966260379
3. [부교재] [수기입력] Software for Data Analysis Chambers JM Springer
4. [부교재] [수기입력] Advanced R Wickham H CRC Press
5. [부교재] [도서관] [ebook]The Basics of S-PLUS [electronic resource] Krause A, Olson M New York, NY : Sp New York 9780387283906
6. [부교재] [도서관] [ebook]Introduction to Scientific Programming and Simulation Using R, Second Edition [electronic resource] Jones O, Maillardet R, Hoboken : CRC Press 9781466570016
7. [부교재] [수기입력] R Programming for Data Science Peng R. lulu.com
8. [주교재] [수기입력] The R Manuals R Core Team http://cran.r-project.org

6.주별 진도계획 및 학습자료(weekly plan & study materials)

제1주(week 1)	[주별진도(topic)] Course introduction: textbooks, resources, installation, …	[학습자료(materials)] R-admin
제2주(week 2)	[주별진도(topic)] Data objects: basic types, vector, matrix, list, data frame	[학습자료(materials)] R-intro
제3주(week 3)	[주별진도(topic)] Plotting and graphics	[학습자료(materials)] Handout
제4주(week 4)	[주별진도(topic)] Data manipulation: loading, subsetting, merging, saving	[학습자료(materials)] R-data

울산대학교 *UWIN* 1/2

FIGURE 2: Syllabus page 1

제5주(week 5)	[주별진도(topic)] Basic commands and functions	[학습자료(materials)] R-lang
제6주(week 6)	[주별진도(topic)] Elements of programming style	[학습자료(materials)]
제7주(week 7)	[주별진도(topic)] Developing R package without Rstudio	[학습자료(materials)] R-exts
제8주(week 8)	[주별진도(topic)] Some useful packages 1 - RODBC, rtf	[학습자료(materials)]
제9주(week 9)	[주별진도(topic)] (석가탄신일 휴무)	[학습자료(materials)]
제10주(week 10)	[주별진도(topic)] Rstudio and some useful packages 2 - ggplot2	[학습자료(materials)]
제11주(week 11)	[주별진도(topic)] Some useful packages 3 - dplyr, tidyr	[학습자료(materials)]
제12주(week 12)	[주별진도(topic)] Some useful packages 4 - lubridate, stringr	[학습자료(materials)]
제13주(week 13)	[주별진도(topic)] Handling date, time, and string	[학습자료(materials)]
제14주(week 14)	[주별진도(topic)] Functional and object-oriented programming in R	[학습자료(materials)]
제15주(week 15)	[주별진도(topic)] 과제 solution 예시 & feedback	[학습자료(materials)]
제16주(week 16)	[주별진도(topic)] Pitfalls and limitations of R	[학습자료(materials)]

FIGURE 3: Syllabus page 1

1

R language

2017-03-15 배균섭 교수님 강의

R Language Definition[1]의 초반 내용에 대해 설명하였습니다.

[1] https://cran.r-project.org/doc/manuals/r-release/R-lang.pdf

2

Graphics

2017-03-22 임형석 교수님 강의

R을 사용해 그림 그리는 방법에 대해 알아보겠습니다.

2.1 Introduction

R에서 상위수준 그림 함수는 그림을 생성합니다. 반면 하위수준 그림 함수는 기존의 그림에 그림을 추가합니다.

2.2 상위수준 그림 함수

2.2.1 상위수준 그림 함수의 주요 인자 (arguments)

- main : 제목
- xlab/ylab : x축 및 y축 레이블
- xlim/ylim : x축 및 y축 범위
- col : 색깔
- lty : 선 모양
- pch : 점 모양
- cex : 그림 성분의 크기
- lwd : 선 굵기
- type : 그림 타입

```
dta <- read.csv("PK.csv")
head(dta)
```

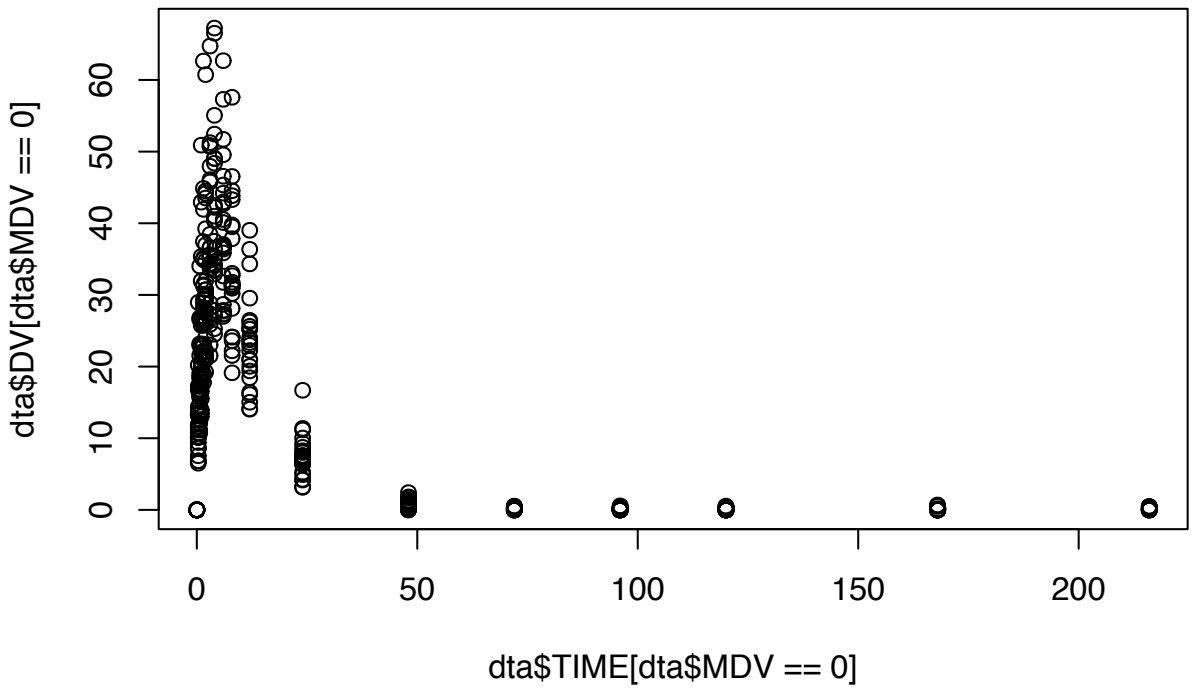
```
##   ID TIME AMT    DV MDV
## 1  1 0.00   0  0.00   0
## 2  1 0.00   4  0.00   1
## 3  1 0.33   0  9.40   0
## 4  1 0.66   0 13.71   0
## 5  1 1.00   0 16.52   0
## 6  1 1.50   0 29.36   0
```

```
str(dta)
```

```
## 'data.frame':    456 obs. of  5 variables:
##  $ ID  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ TIME: num  0 0 0.33 0.66 1 1.5 2 3 4 6 ...
##  $ AMT : num  0 4 0 0 0 0 0 0 0 0 ...
##  $ DV  : num  0 0 9.4 13.7 16.5 ...
##  $ MDV : num  0 1 0 0 0 0 0 0 0 0 ...
```

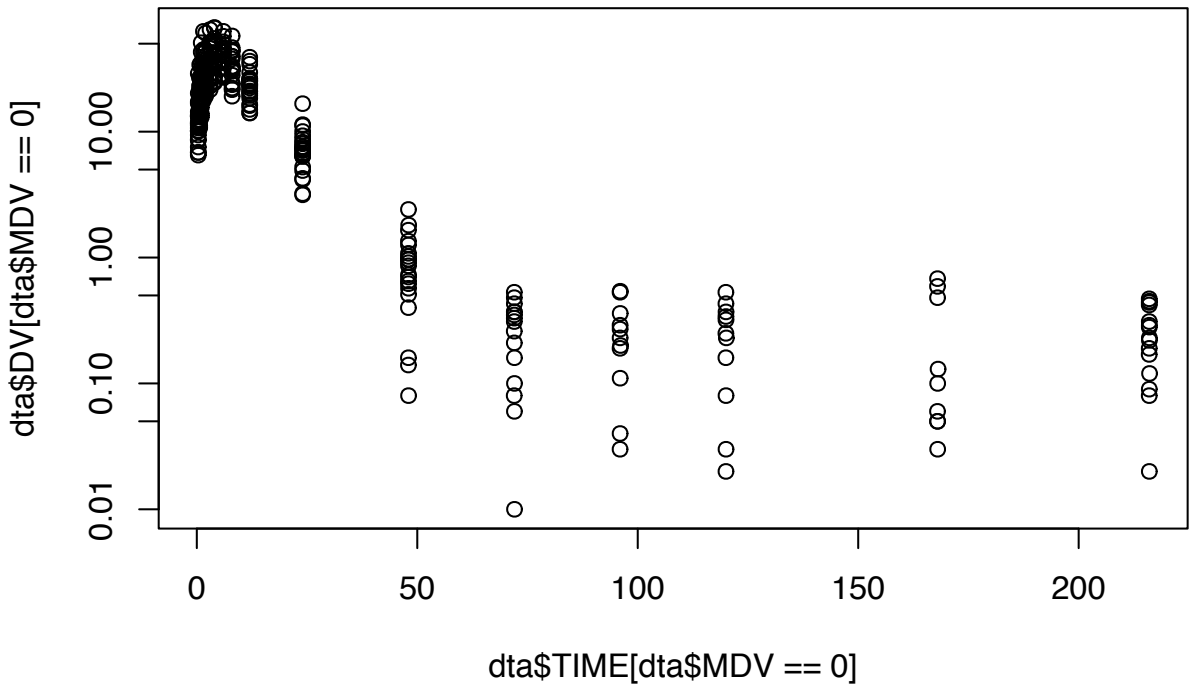
2.2.2 scatter plot

```
plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0])
```

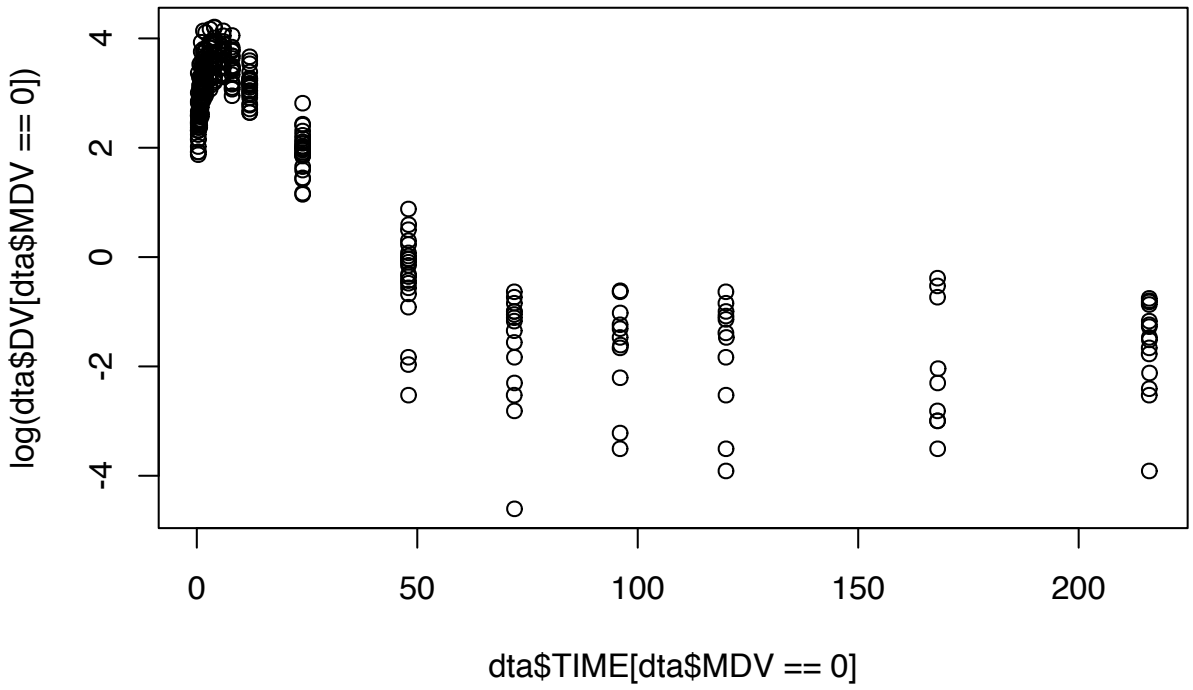


```
plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], log="y")
```

```
## Warning in xy.coords(x, y, xlabel, ylabel, log): 86 y
## values <= 0 omitted from logarithmic plot
```

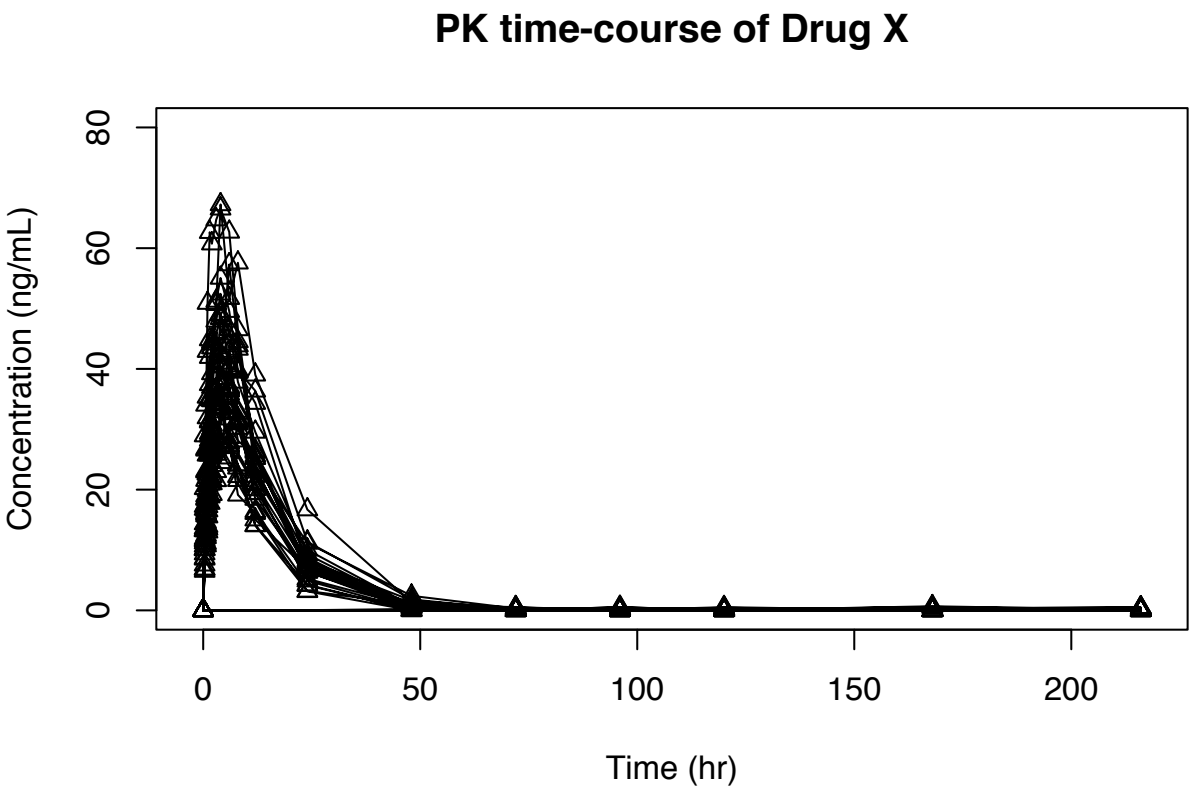



```
plot(dta$TIME[dta$MDV==0], log(dta$DV[dta$MDV==0]))
```

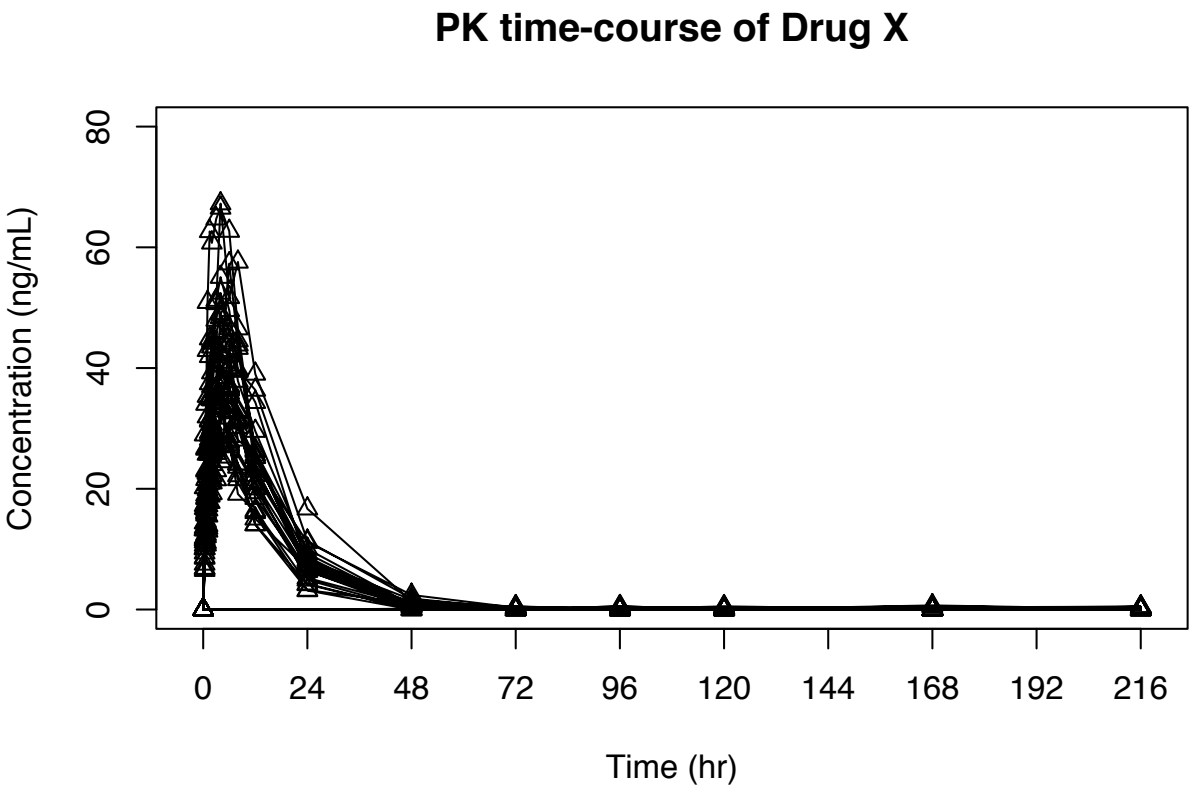



```
plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0]
     , xlab="Time (hr)", ylab="Concentration (ng/mL)"
```

```
      , type="o", pch=2, col=1, main="PK time-course of Drug X"
      , xlim =c(-2,218), ylim=c(0,80))
```

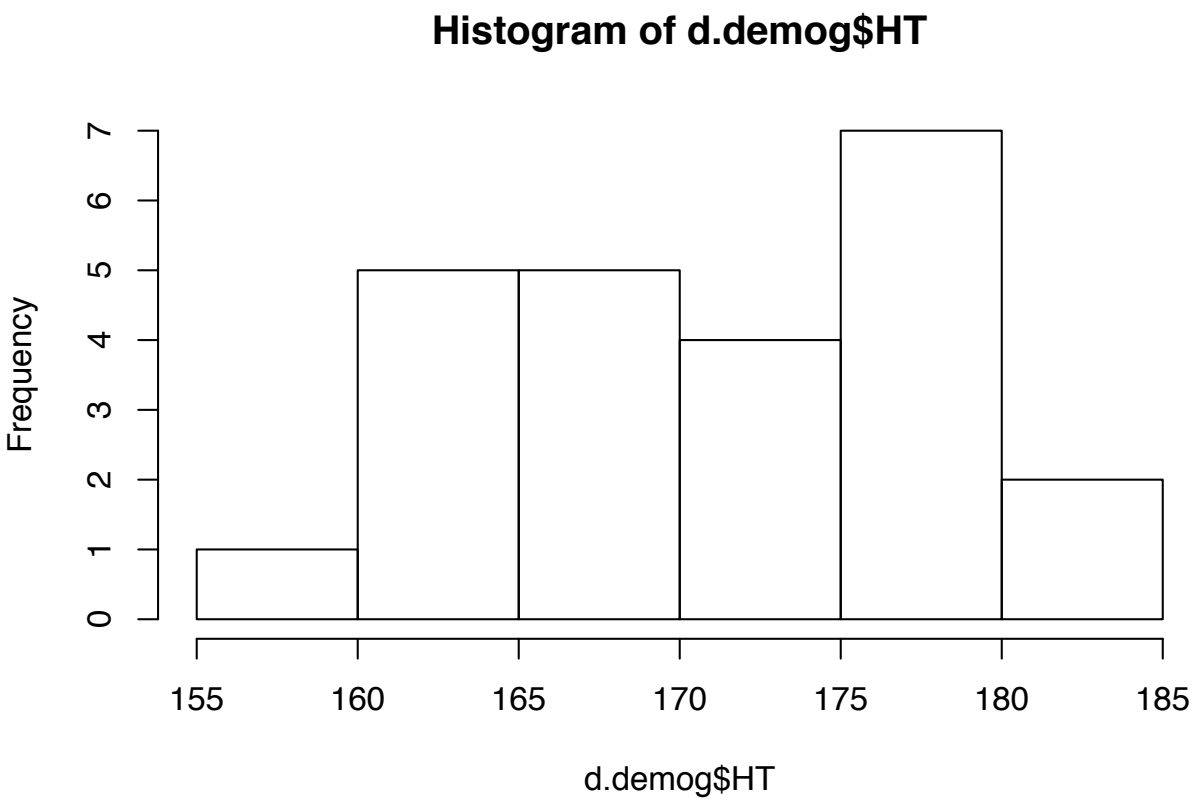



```
plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], axes=F,
      , xlab="Time (hr)", ylab="Concentration (ng/mL)"
      , type="o", pch=2, col=1, main="PK time-course of Drug X"
      , xlim =c(-2,218), ylim=c(0,80))
axis(1, at=seq(0, 218, 24))
axis(2)
box()
```

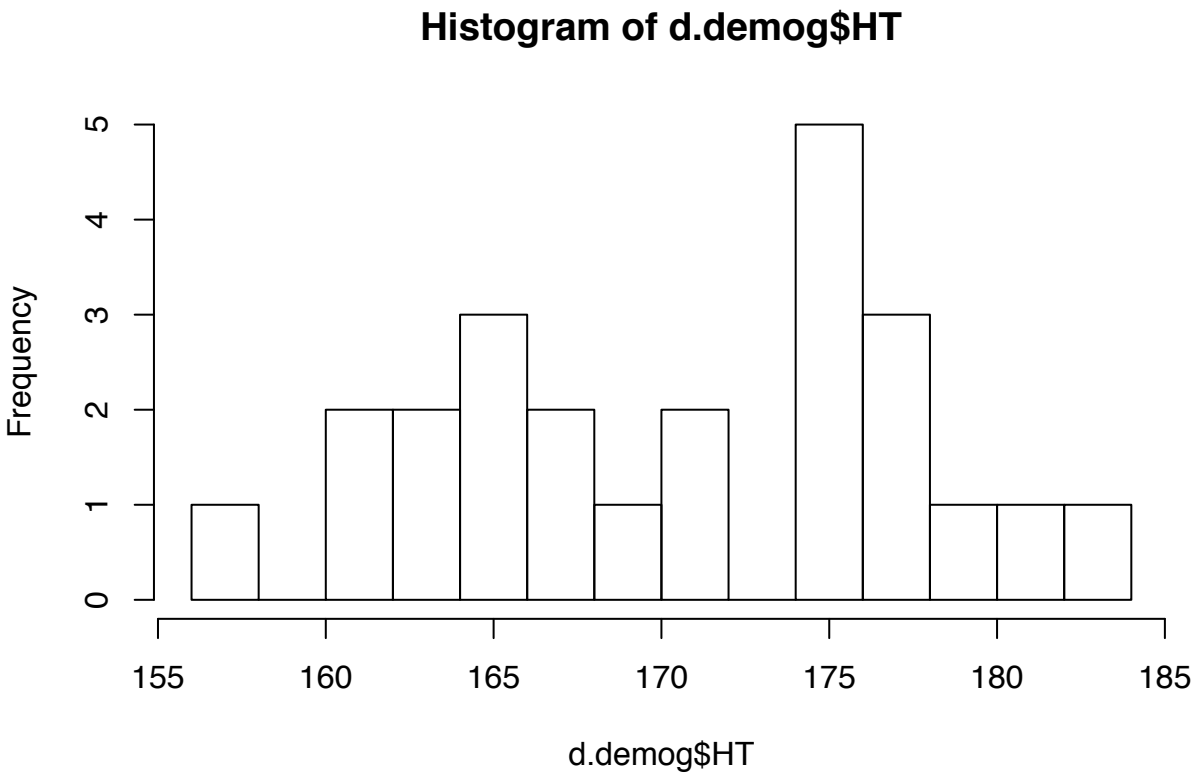


2.2.3 Histogram

```
d.demog <- read.csv("DEMOG.csv")

hist(d.demog$HT)
```

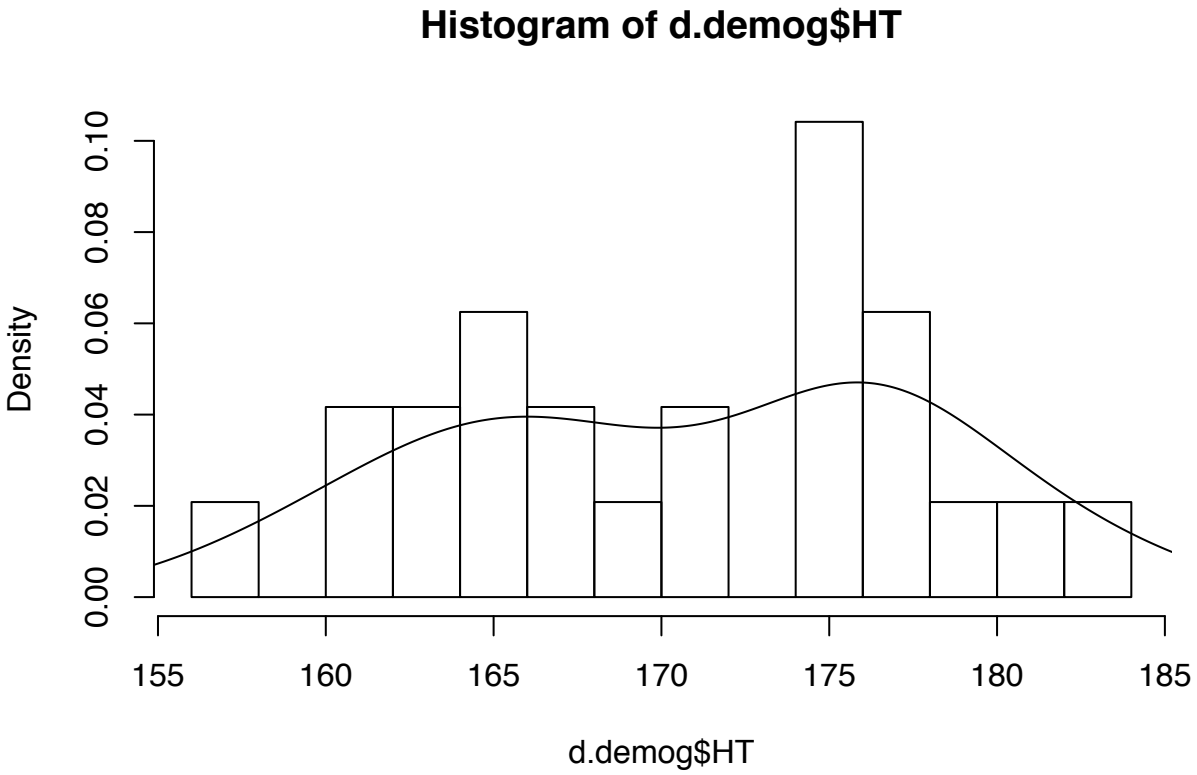



```
hist(d.demog$HT, breaks=10)
hist(d.demog$HT, nclass=10)
```

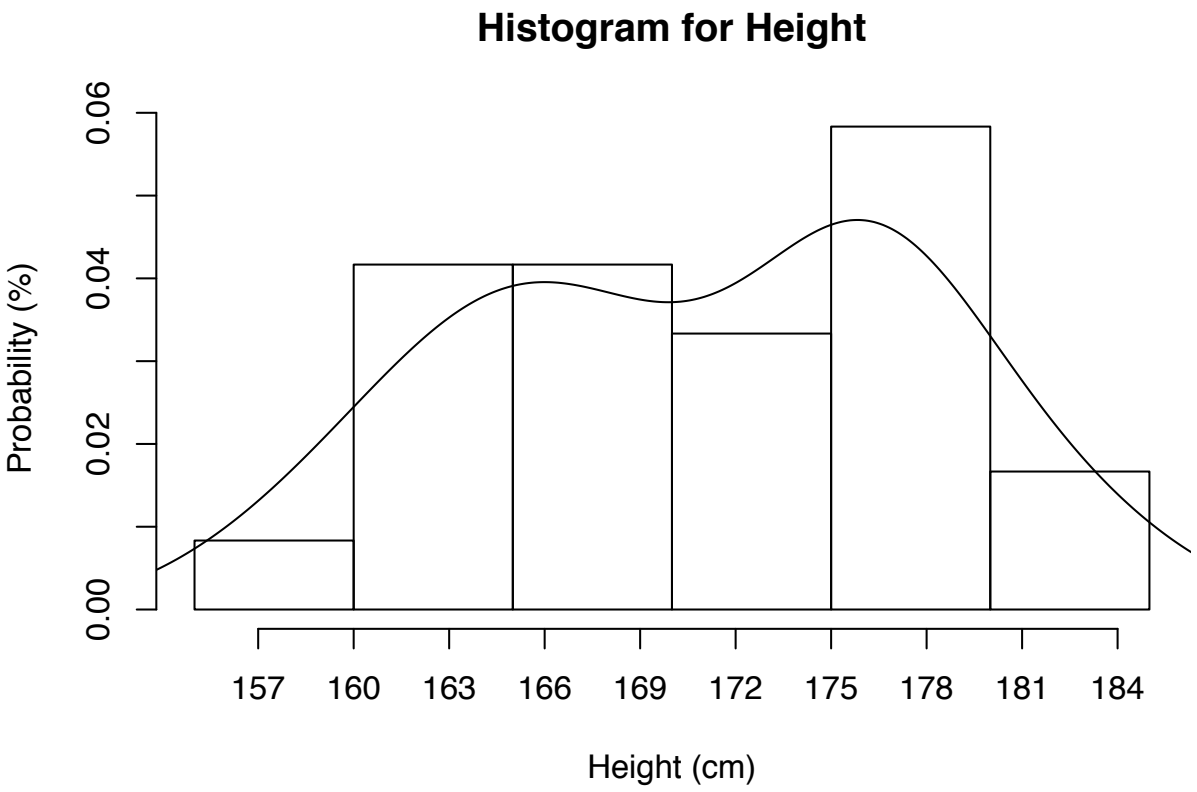


2.2.3.1 with density line

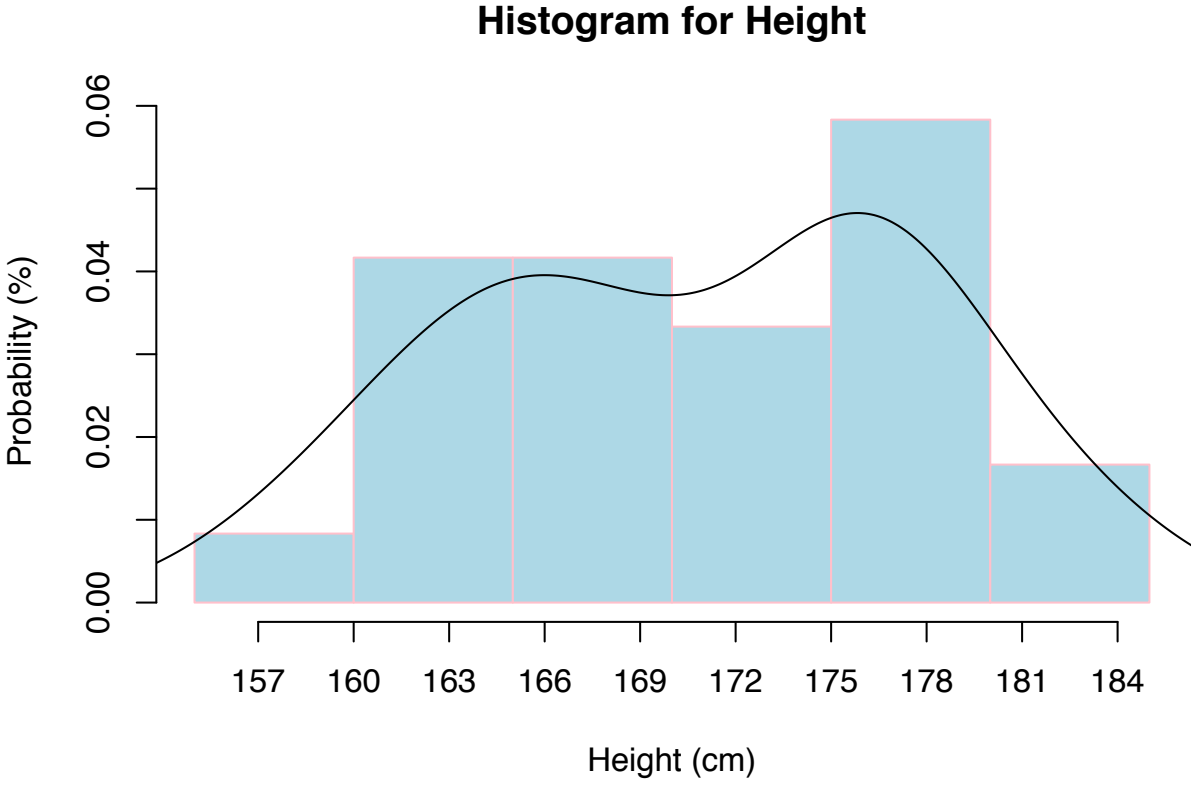
```
hist (d.demog$HT, probability=TRUE, breaks=10)
lines(density(d.demog$HT))
```




```
hist (d.demog$HT, probability=TRUE, breaks=9, xaxt="n"
      , main="Histogram for Height", xlab="Height (cm)", ylab="Probability (%)")
axis(1, at=seq(min(d.demog$HT), max(d.demog$HT), 3))
lines(density(d.demog$HT))
```

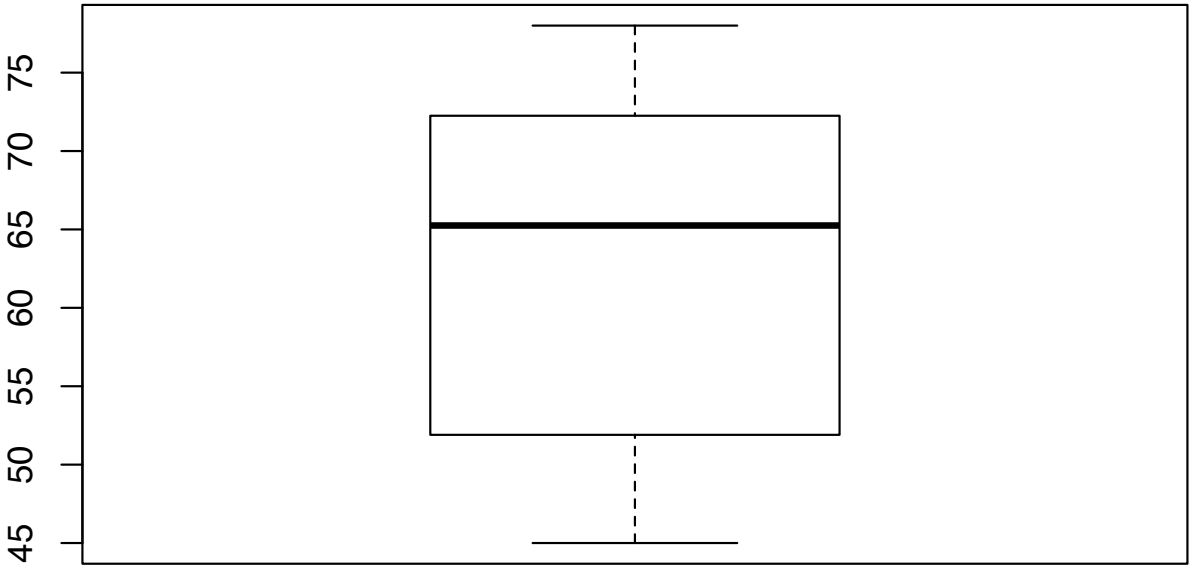



```
hist (d.demog$HT, probability=TRUE, breaks=9, xaxt="n"
      , main="Histogram for Height", xlab="Height (cm)", ylab="Probability (%)"
      , col = "lightblue", border = "pink")
axis(1, at=seq(min(d.demog$HT), max(d.demog$HT), 3))
lines(density(d.demog$HT))
```

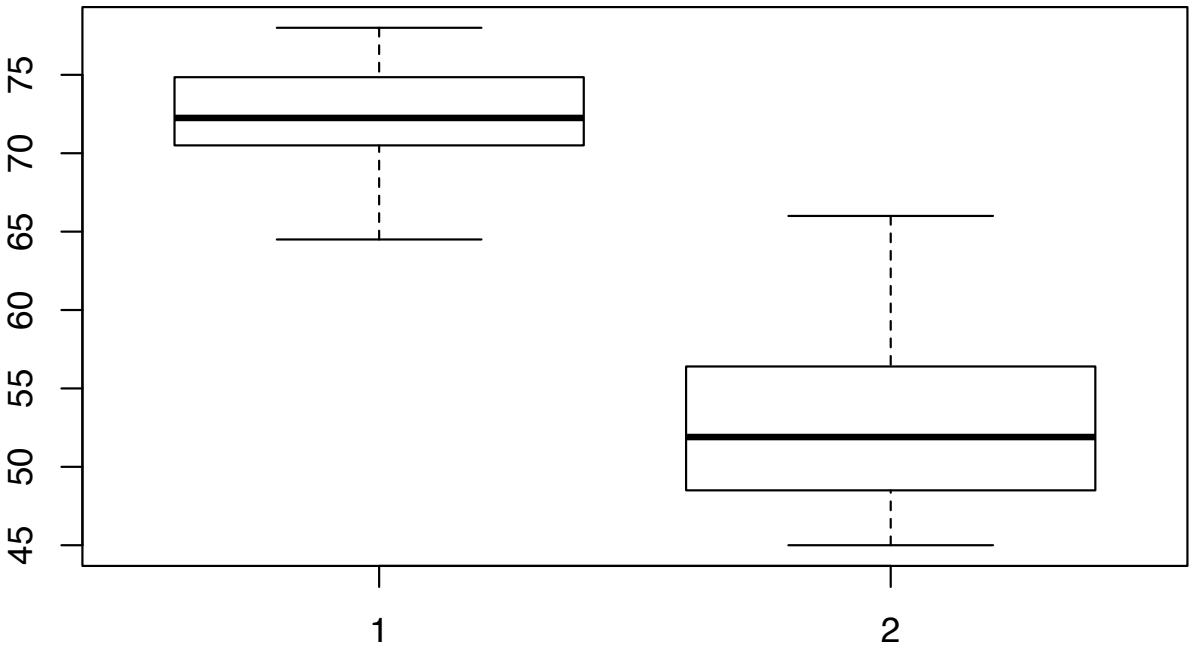


2.2.4 Box-Whisker Plot

```
boxplot(d.demog$WT)
```

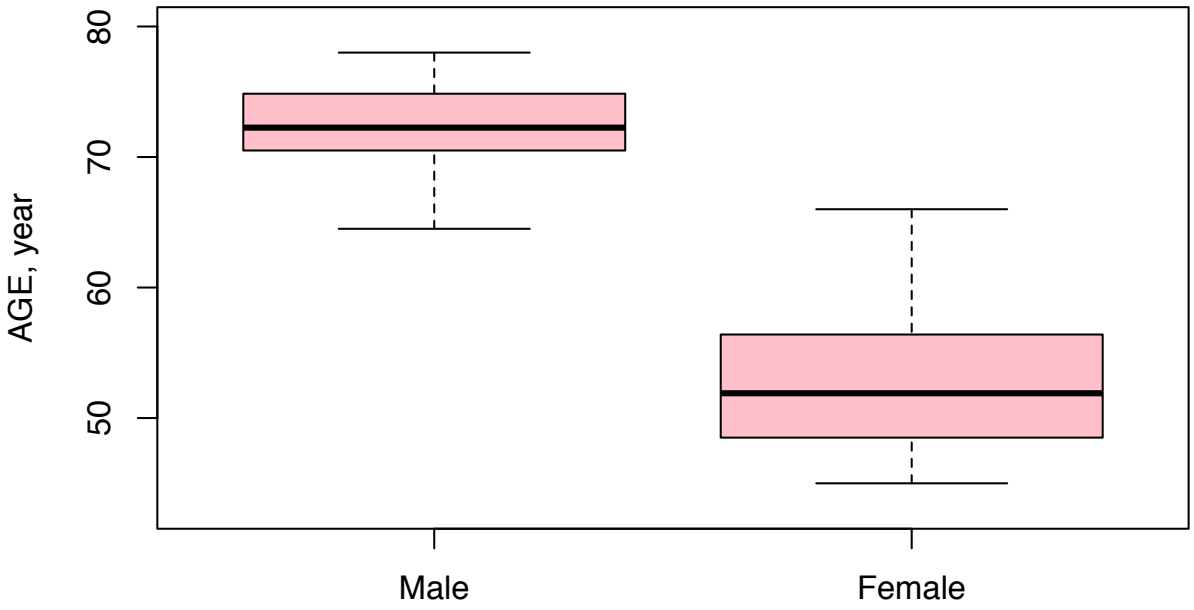



```
boxplot(d.demog$WT ~ d.demog$SEX)

boxplot(split(d.demog$WT, d.demog$SEX))
```

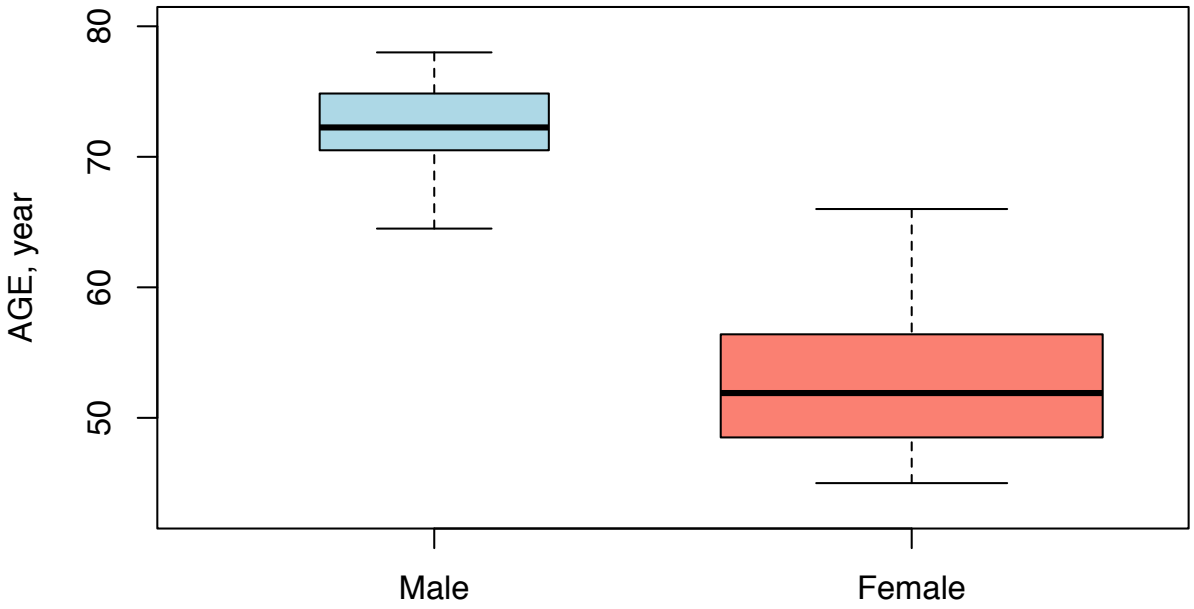



```
boxplot(WT ~ SEX, data=d.demog)

boxplot(d.demog$WT ~ d.demog$SEX
```

```
        , names=c("Male","Female"), ylab="AGE, year", ylim=c(min(d.demog$WT)-2, max(d.demog$WT)
        , col="pink")
```

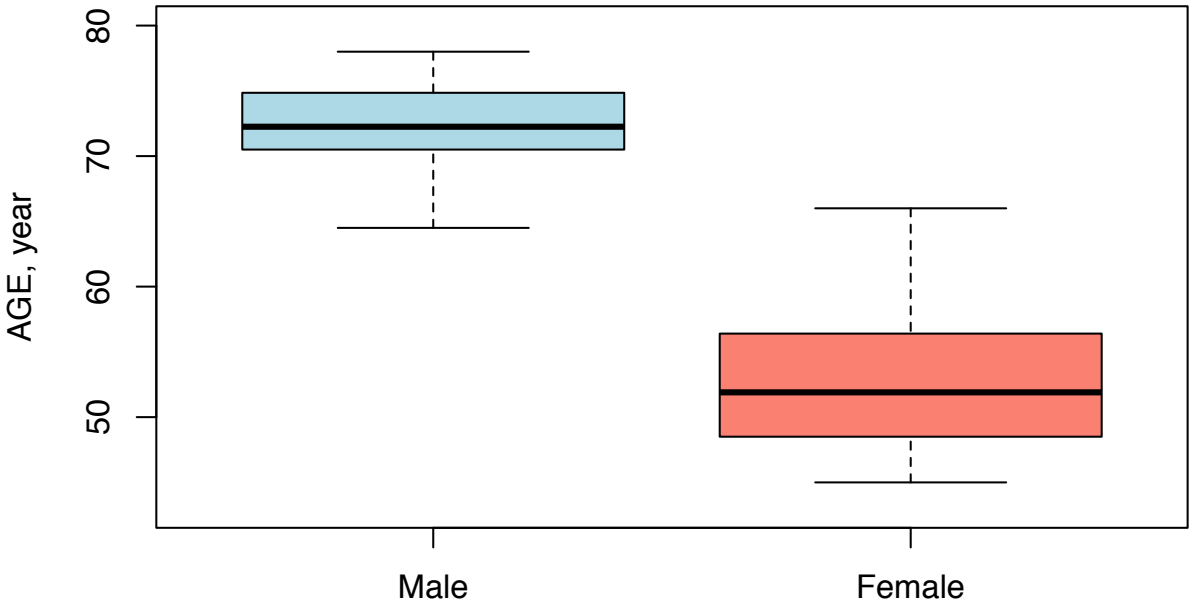



```
boxplot(d.demog$WT ~ d.demog$SEX
        , names=c("Male","Female"), ylab="AGE, year", ylim=c(min(d.demog$WT)-2, max(d.demog$WT)
        , col=c("lightblue", "salmon"), width=c(0.6, 1))
```



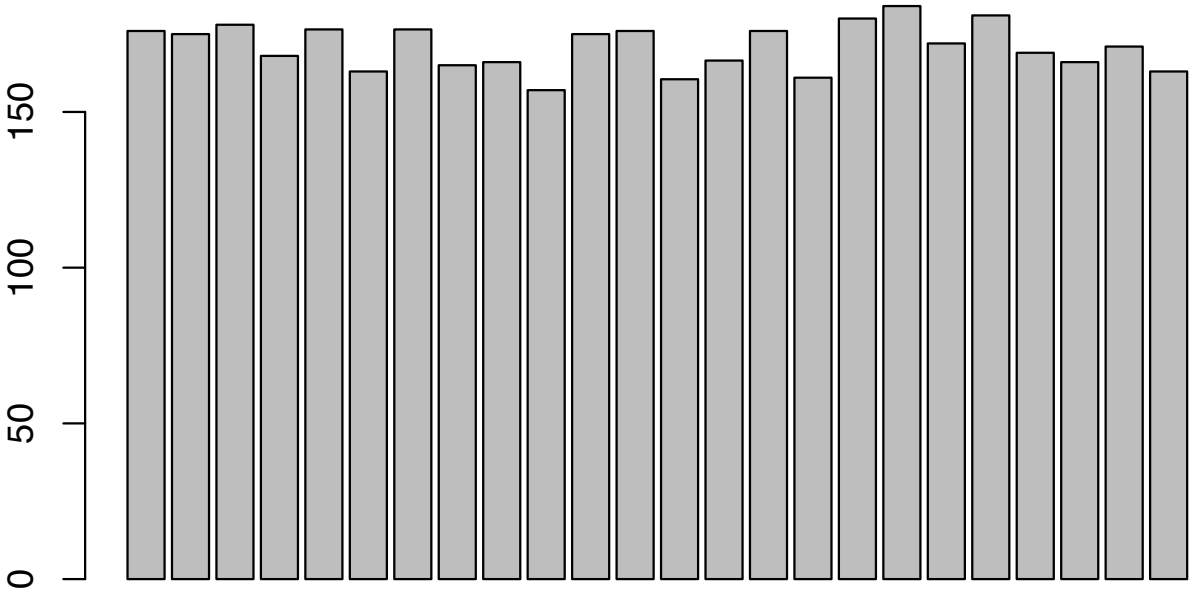
-varwidth: if varwidth is TRUE, the boxes are drawn with widths proportional to the square-roots of the number of observations in the groups.

```
boxplot(d.demog$WT ~ d.demog$SEX
        , names=c("Male","Female"), ylab="AGE, year", ylim=c(min(d.demog$WT)-2, max(d.demog$WT)
        , col=c("lightblue", "salmon")
        , varwidth=TRUE)
```



2.2.5 Bar Plot

```
barplot(d.demog$HT)
```

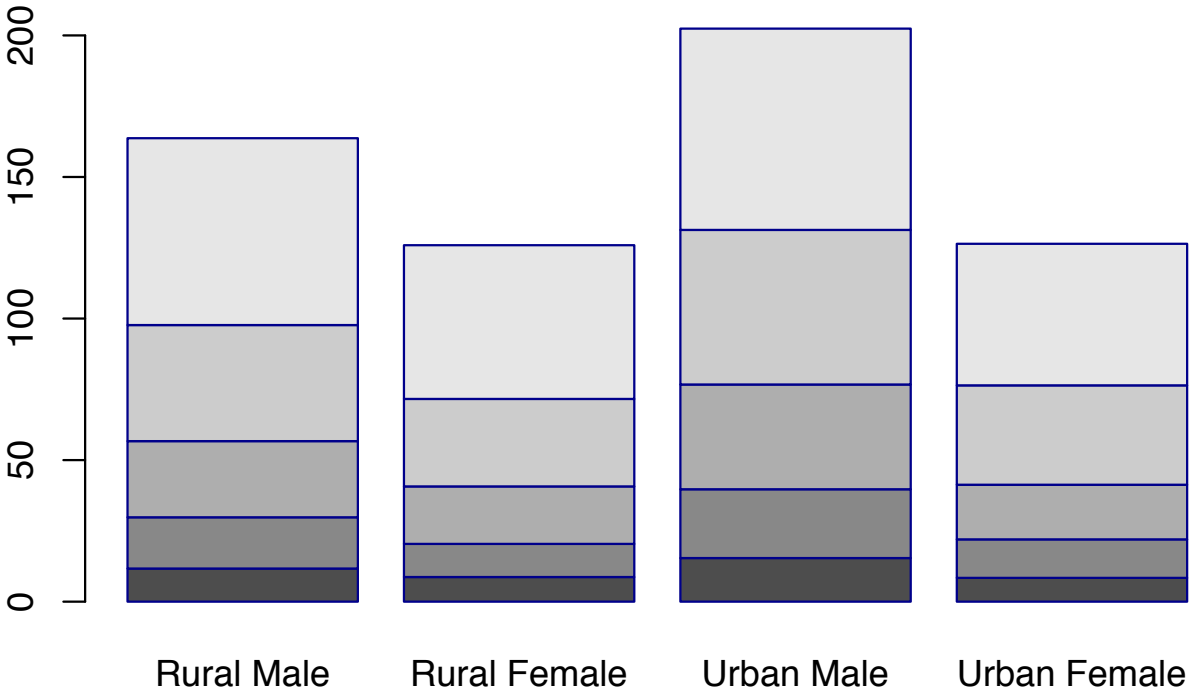



```
VADeaths
```

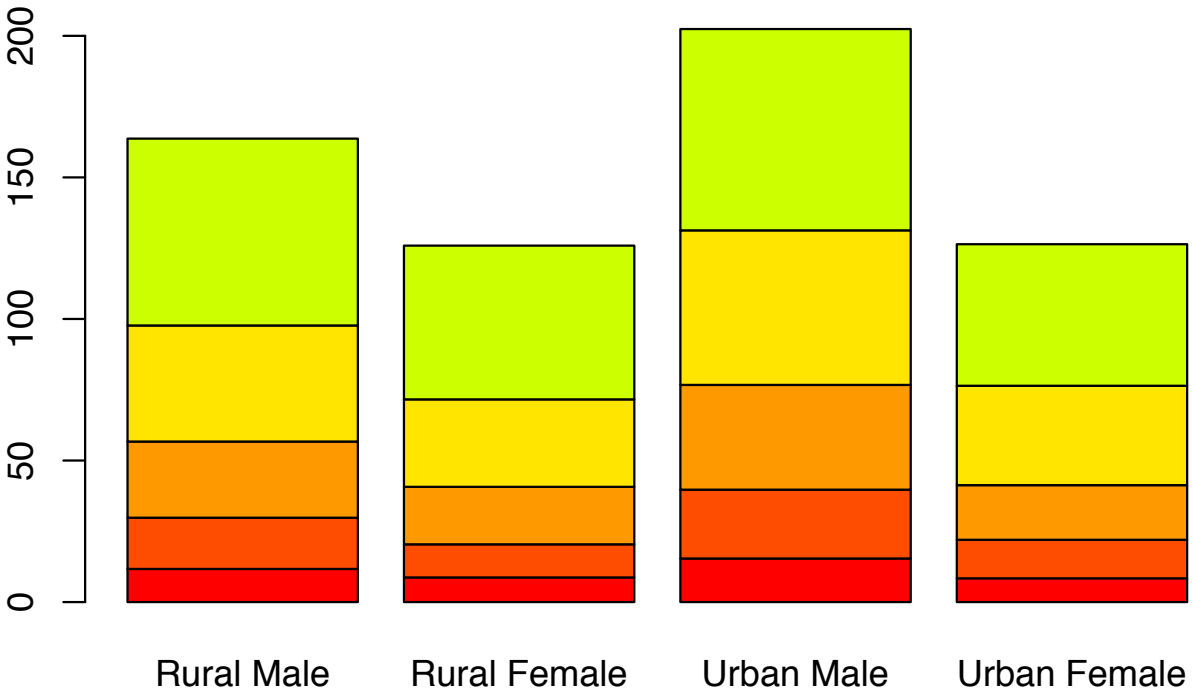
```
##       Rural Male Rural Female Urban Male Urban Female
## 50-54       11.7          8.7       15.4          8.4
## 55-59       18.1         11.7       24.3         13.6
## 60-64       26.9         20.3       37.0         19.3
```

```
## 65-69       41.0          30.9        54.6          35.1
## 70-74       66.0          54.3        71.1          50.0
```

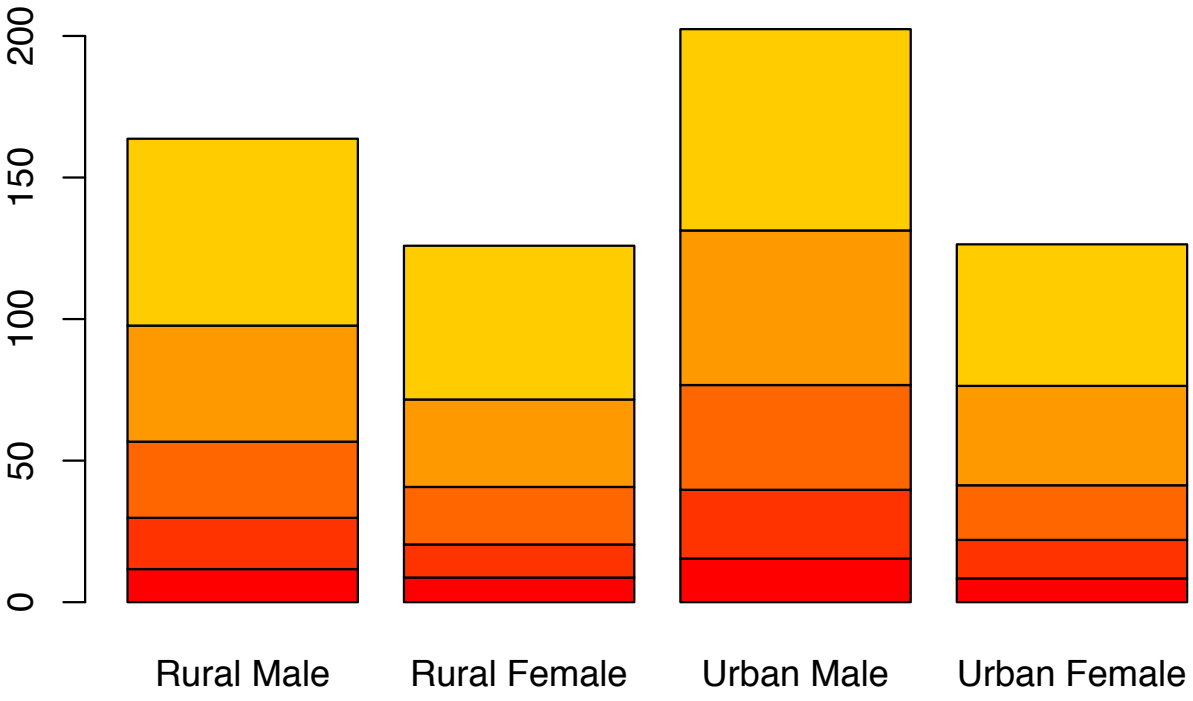
```
barplot(VADeaths, border = "dark blue")
```



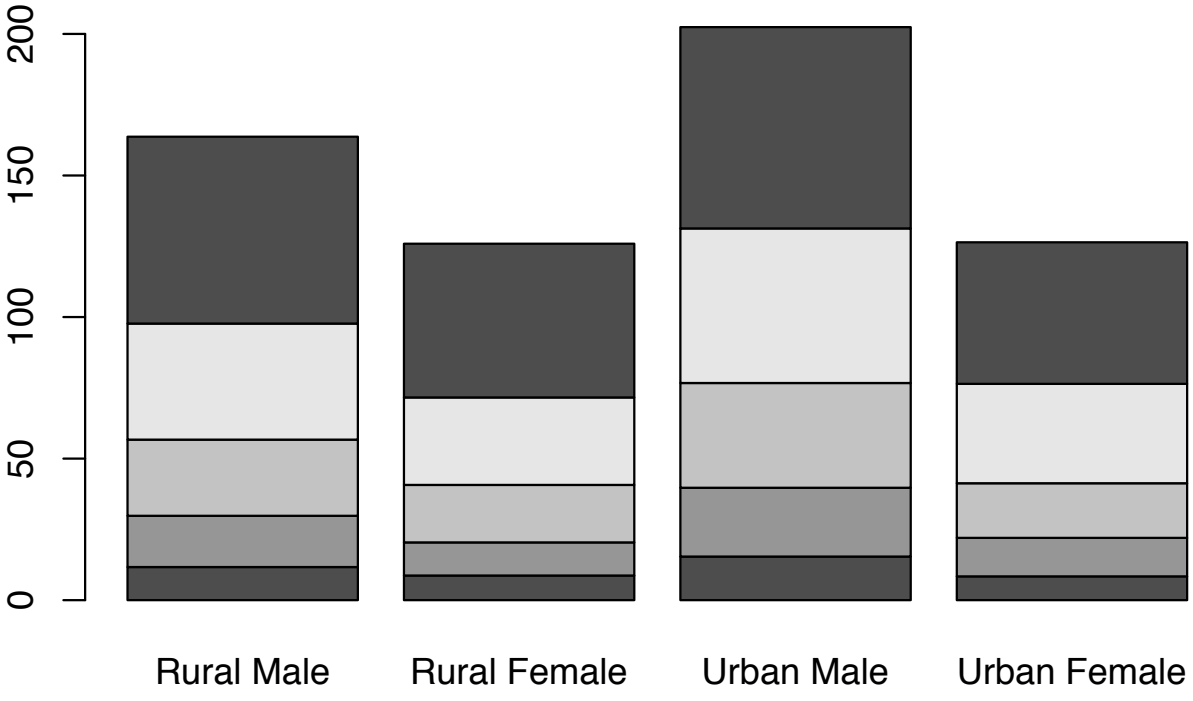

```
barplot(VADeaths, col = rainbow(20))
```




```
barplot(VADeaths, col = heat.colors(8))
```



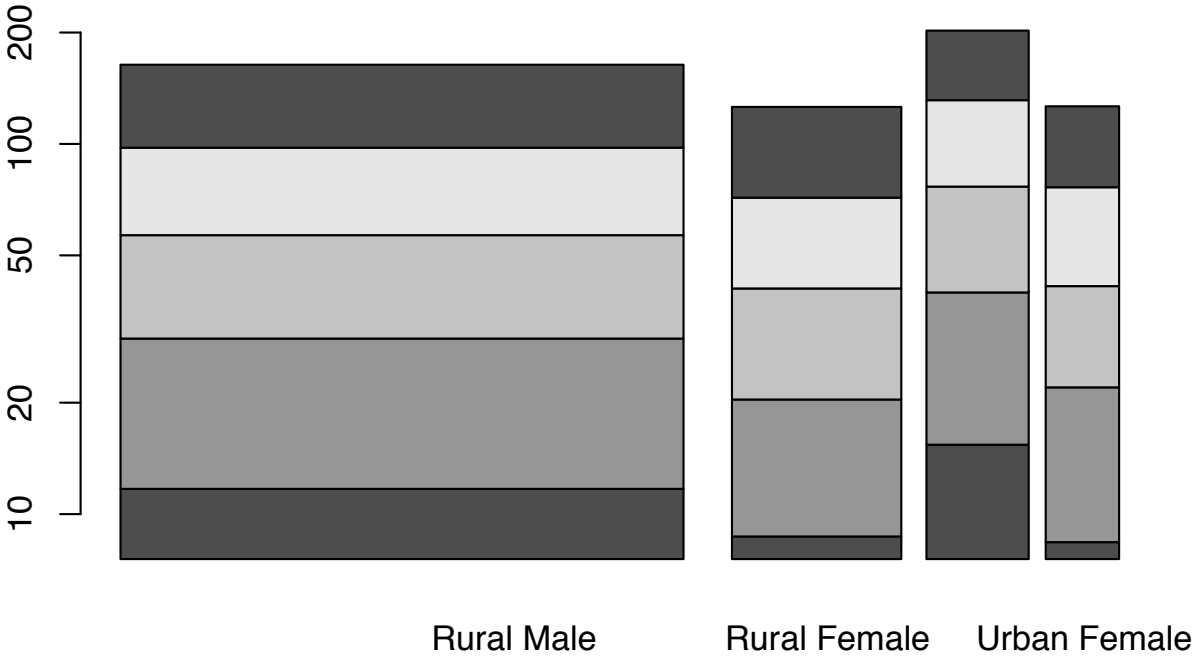

```
barplot(VADeaths, col = gray.colors(4))
```




```
barplot(VADeaths, col = gray.colors(4), log="x")
```

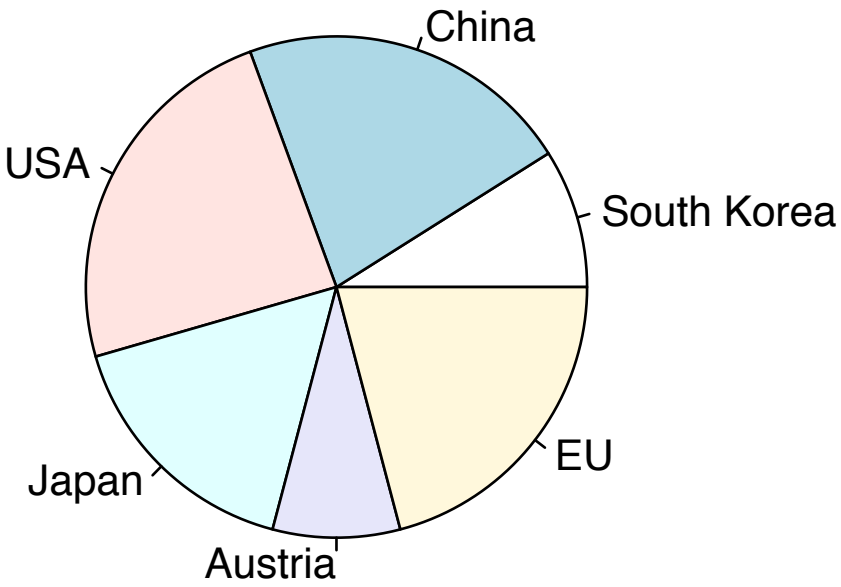

```
barplot(VADeaths, col = gray.colors(4), log="y")
```


```
barplot(VADeaths, col = gray.colors(4), log="xy")
```



2.2.6 pie chart

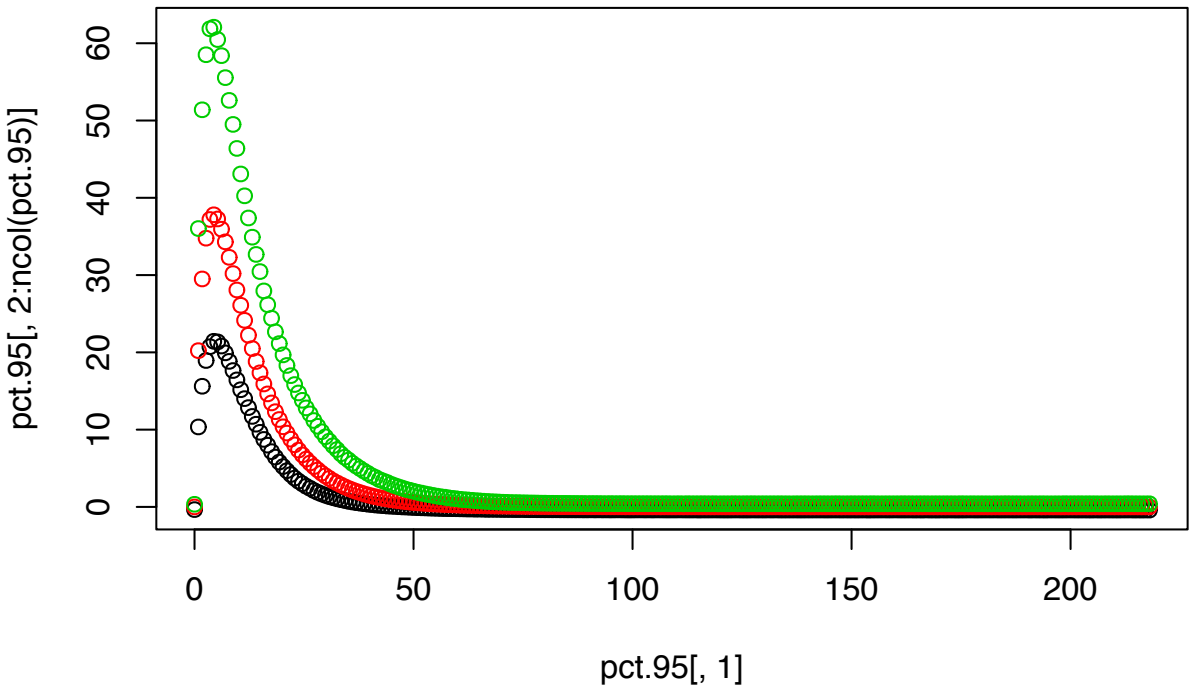
```
drug.X.market <- c(0.12, 0.29, 0.32, 0.22, 0.11, 0.28)
names(drug.X.market) <- c("South Korea","China","USA","Japan","Austria","EU")
pie(drug.X.market)
```



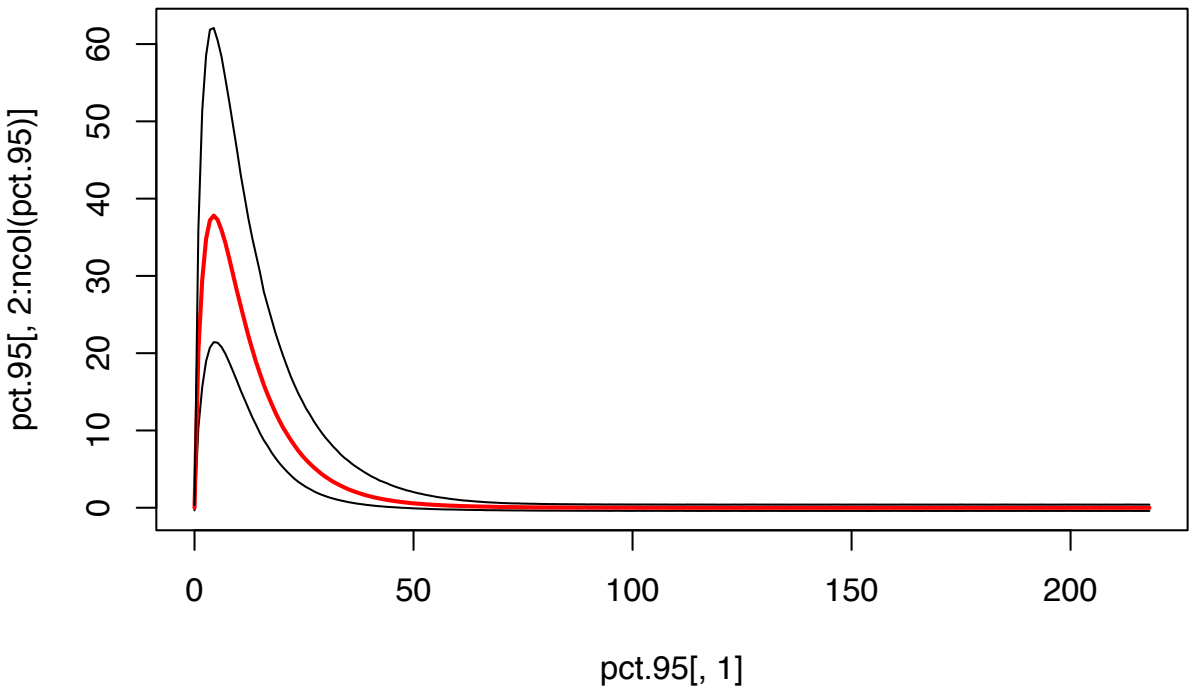
2.2.7 matplot 함수

2.2.7.1 matrix와 column 사이의 그림

```
pct.95 <- read.csv("pct95.csv")
matplot(pct.95[,1], pct.95[,2:ncol(pct.95)], pch=1)
```

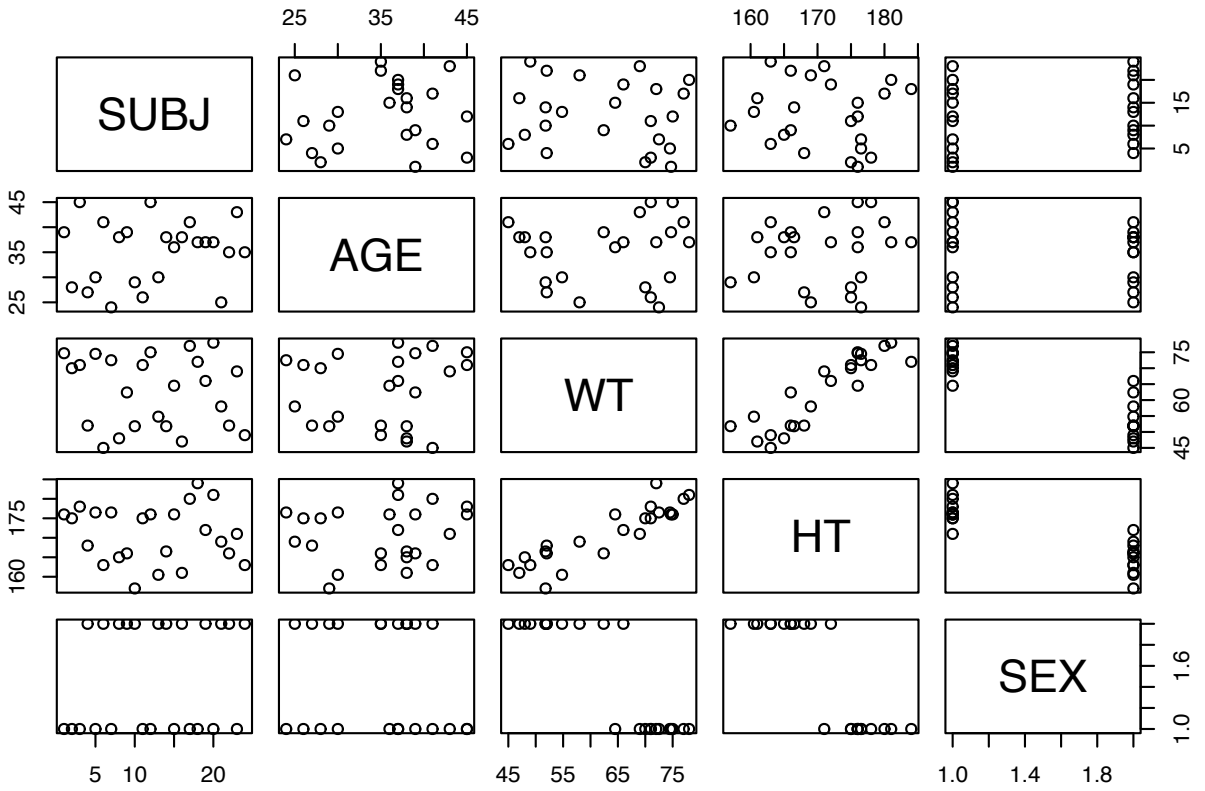



```
matplot(pct.95[,1], pct.95[,2:ncol(pct.95)], pch=1, col=c(1,2,1), type="l", lty=1, lwd=c(1,2,1)
```



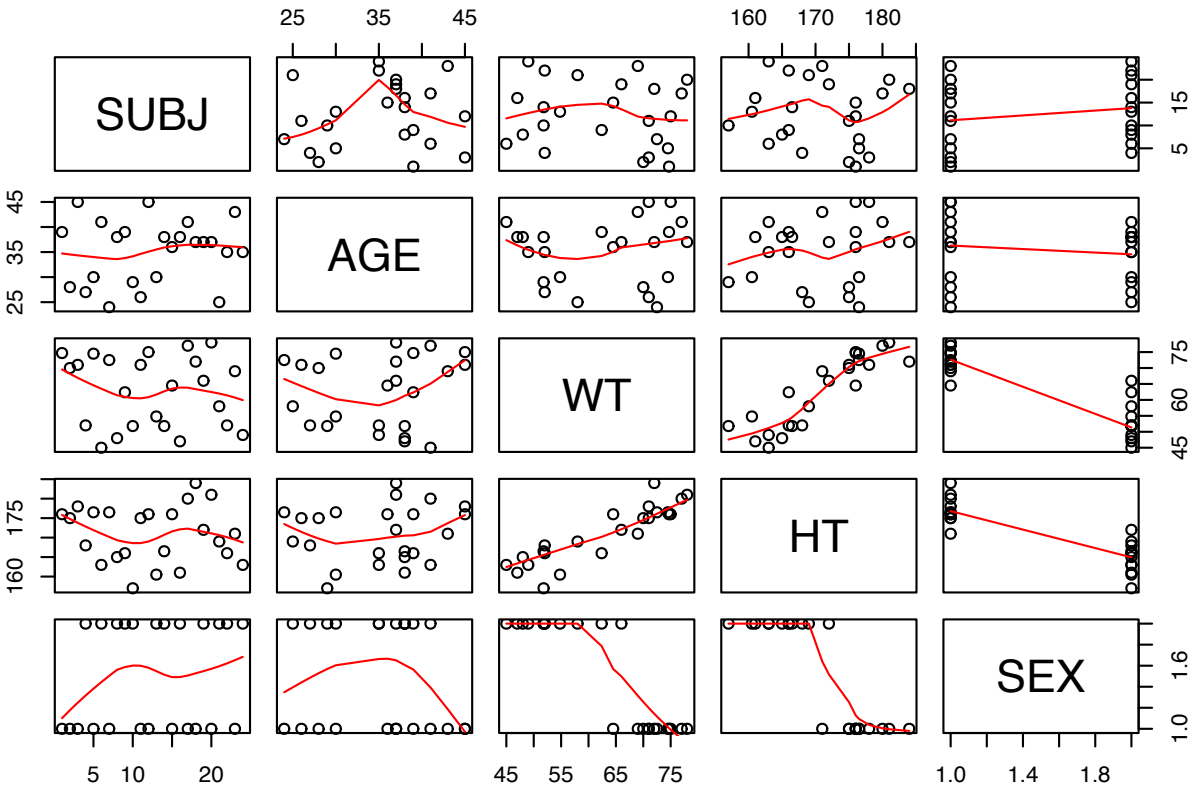
2.2.8 Scatter plot matrices (pairs plots)

```
pairs(d.demog)
```

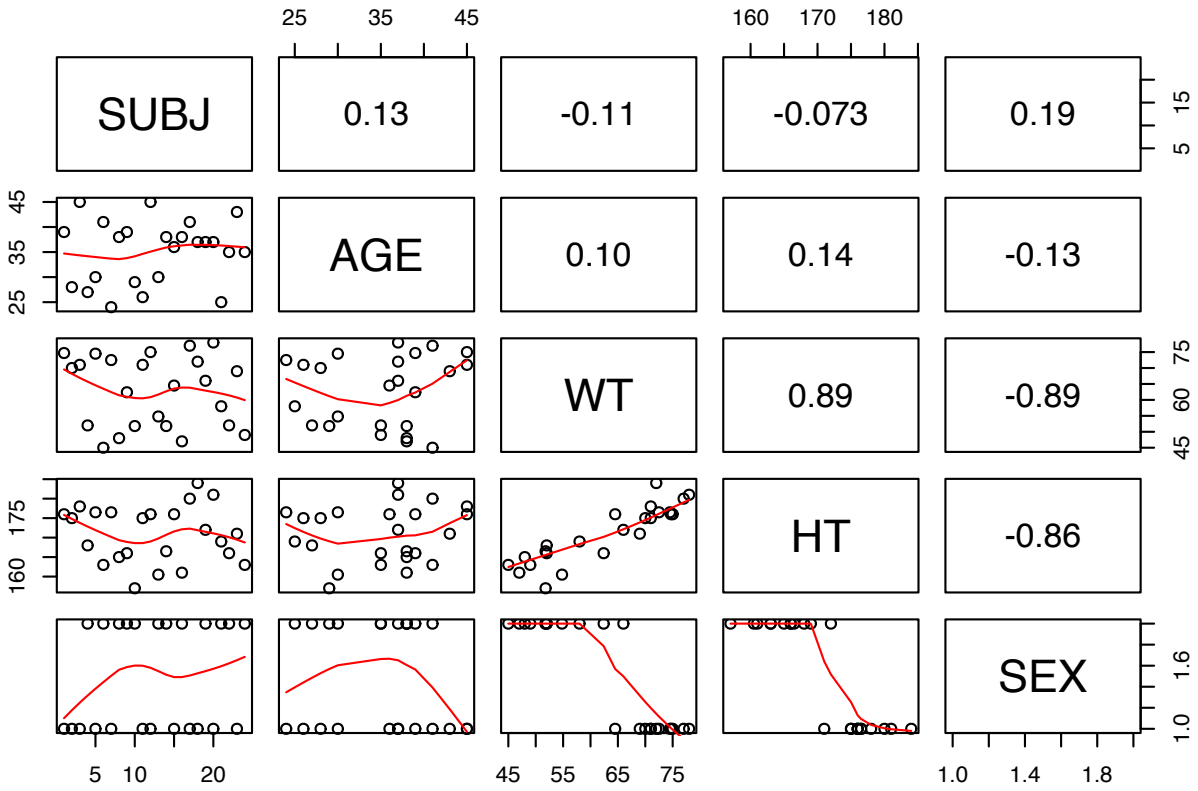


2.2.8.1 add a loess smoother, type

```
pairs(d.demog, panel = panel.smooth)
```




```
panel.cor <- function(x, y, digits=2, prefix="", cex.cor)
{
    usr <- par("usr"); on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    r = (cor(x, y))
    txt <- format(c(r, 0.123456789), digits=digits)[1]
    txt <- paste(prefix, txt, sep="")
    if(missing(cex.cor)) cex <- 1.5
    text(0.5, 0.5, txt, cex = 1.5)
}

pairs(d.demog, lower.panel=panel.smooth, upper.panel=panel.cor)
```

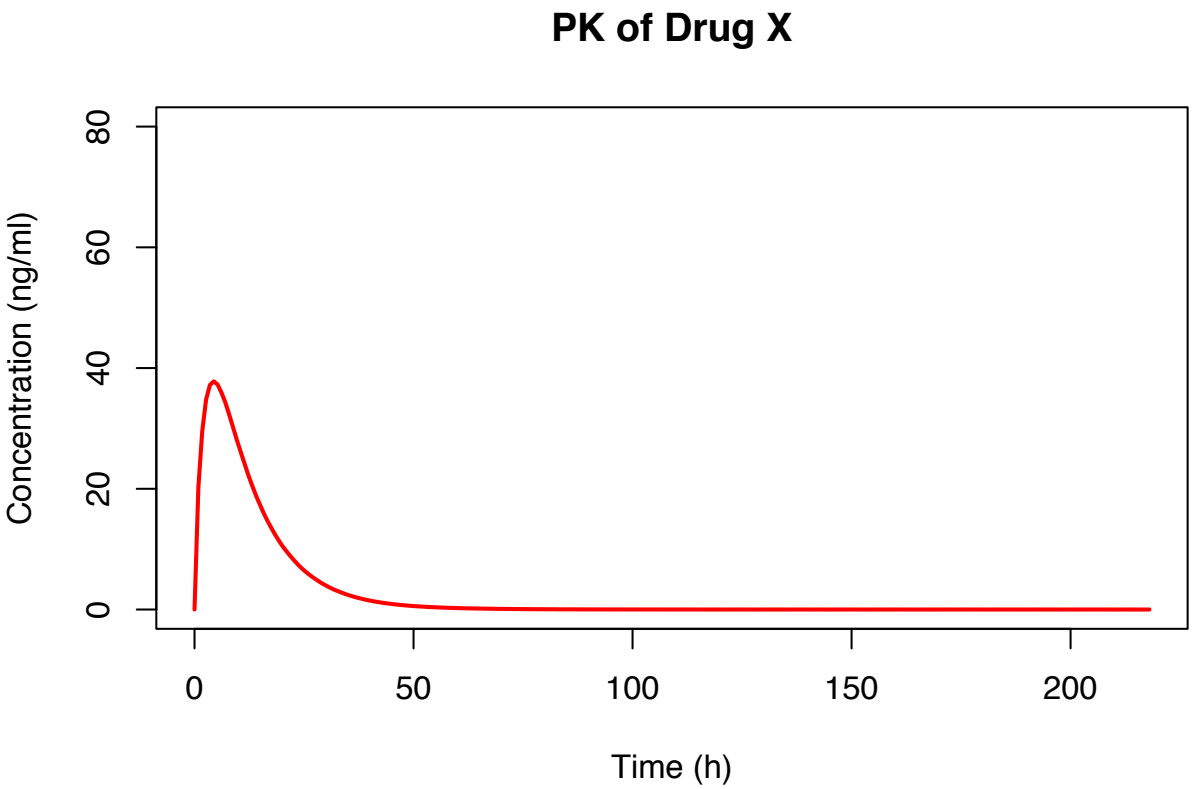


2.3 하위수준 그림 함수

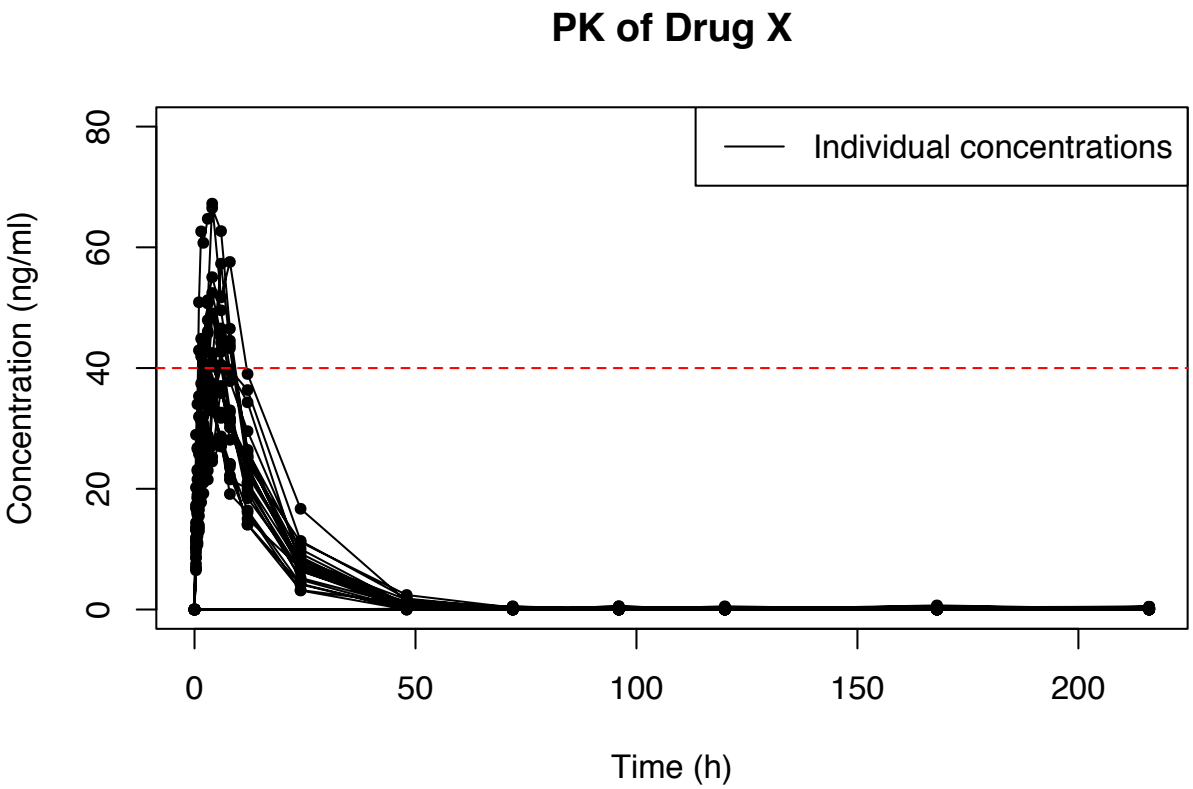
- points : 점추가
- lines : 선 추가
- abline : 기준선 추가
- mtext : 텍스트 추가
- legend : 설명 (legend) 추가
- polygon : polygon 추가

2.3.1 점, 선, 설명 추가 하기 {add}

```
plot(pct.95$TIME, pct.95$PCT50, main="PK of Drug X"
     , type="l", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80), lty=1, col="red", lwd=2)
```

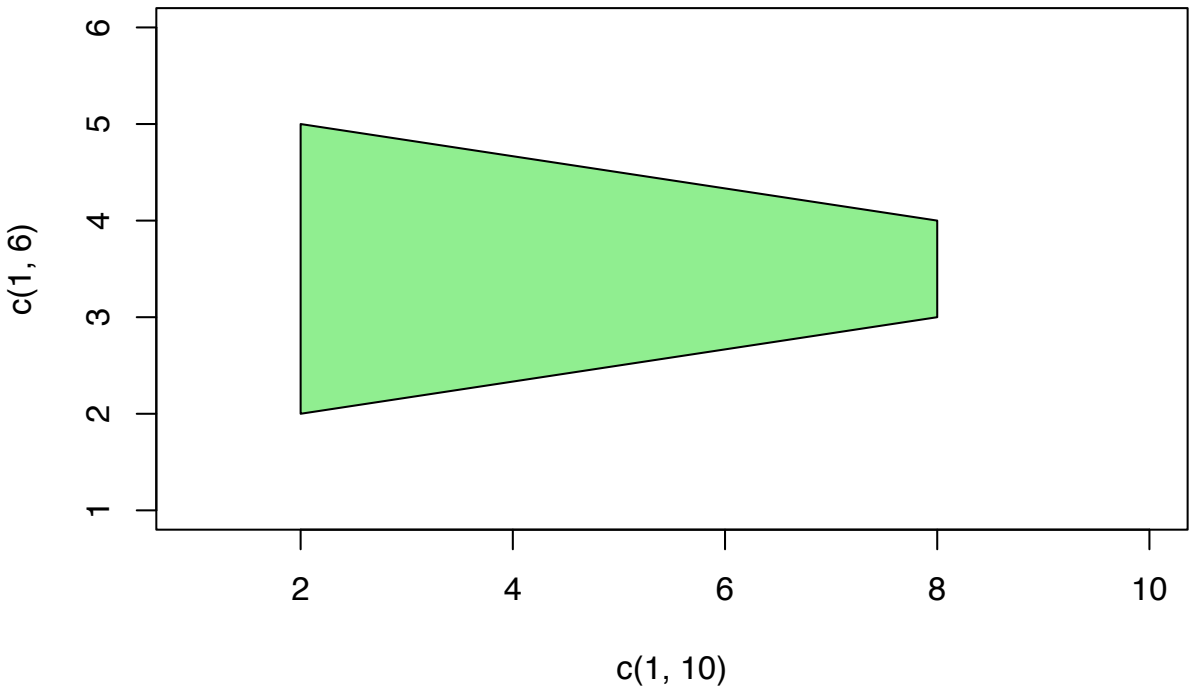



```r
plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  # abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")
```

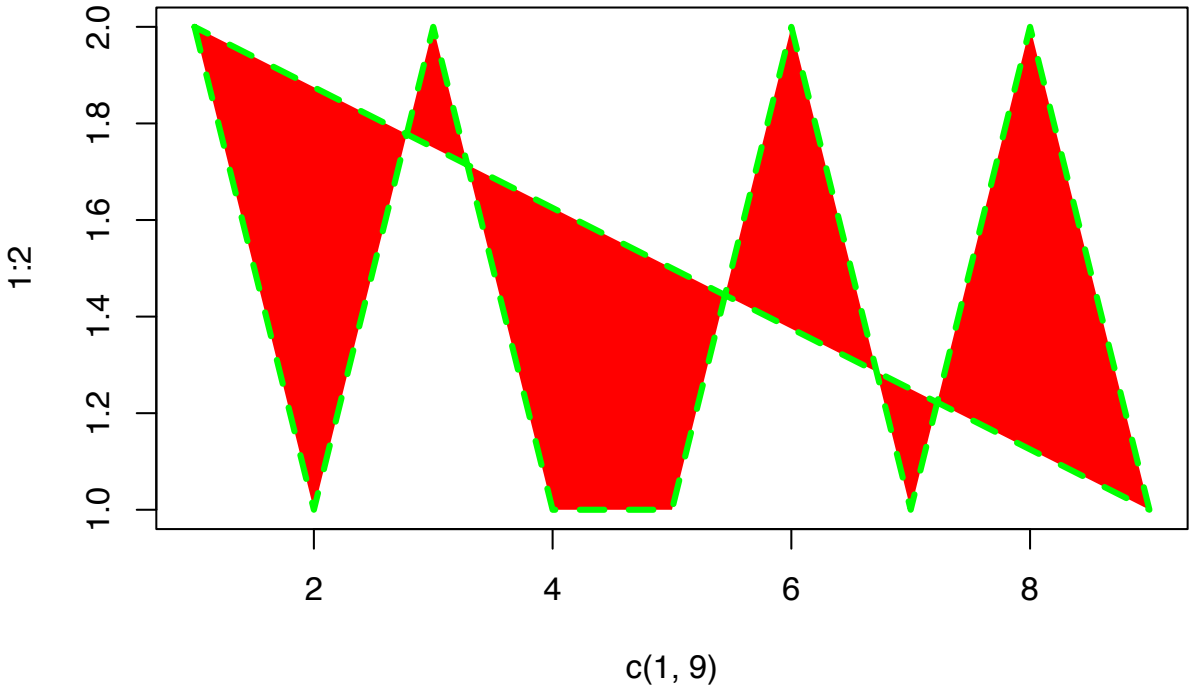


2.3.2 polygon 함수

```
plot(c(1, 10), c(1, 6), type = "n")
polygon(c(2,8,8,2), c(5,4,3,2), col="lightgreen")
```



```
plot(c(1, 9), 1:2, type = "n")
polygon(1:9, c(2,1,2,1,1,2,1,2,1),
        col = c("red", "blue"),
        border = c("green", "yellow"),
        lwd = 3, lty = c("dashed", "solid"))
```



2.4 그림 출력하기

2.4.1 pdf graphics devices

```
pdf("PK_of_Drug_X.pdf")

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  #abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")
```

```
dev.off()
```

```
## cairo_pdf
##         2
```

2.4.2 PNG graphics devices

```
png("PK_of_Drug_X.png")

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  #abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")

dev.off()
```

```
## cairo_pdf
##         2
```

3

Data Import / Export

2017-03-29 배균섭 교수님 강의

이번 시간에는 자료를 불러오고 조작을 가한 뒤 저장하는 방법에 대해 알아보겠습니다.

3.1 Read.csv

`setwd` 명령어를 통해서 자료가 있는 작업 공간을 설정할 수 있습니다. 설정 후에서는 `dir()`을 통해 파일의 이름을 확인 할 수 있습니다. `read.csv`를 통해서 자료를 R에서 사용할 수 있게 됩니다.

```
setwd("D:/Rt")
dir()
mydata <- read.csv("MyData2017.csv", as.is=TRUE)
```

3.2 Theoph 데이타

R에 기본적으로 들어있는 `Theoph` 약동학 자료에 대해 살펴보겠습니다.

```
head(Theoph, n = 11)
```

```
##    Subject   Wt Dose  Time  conc
## 1        1 79.6 4.02  0.00  0.74
```

```
## 2        1 79.6 4.02  0.25  2.84
## 3        1 79.6 4.02  0.57  6.57
## 4        1 79.6 4.02  1.12 10.50
## 5        1 79.6 4.02  2.02  9.66
## 6        1 79.6 4.02  3.82  8.58
## 7        1 79.6 4.02  5.10  8.36
## 8        1 79.6 4.02  7.03  7.47
## 9        1 79.6 4.02  9.05  6.89
## 10       1 79.6 4.02 12.12  5.94
## 11       1 79.6 4.02 24.37  3.28
```

```
tail(Theoph, n = 11)
```

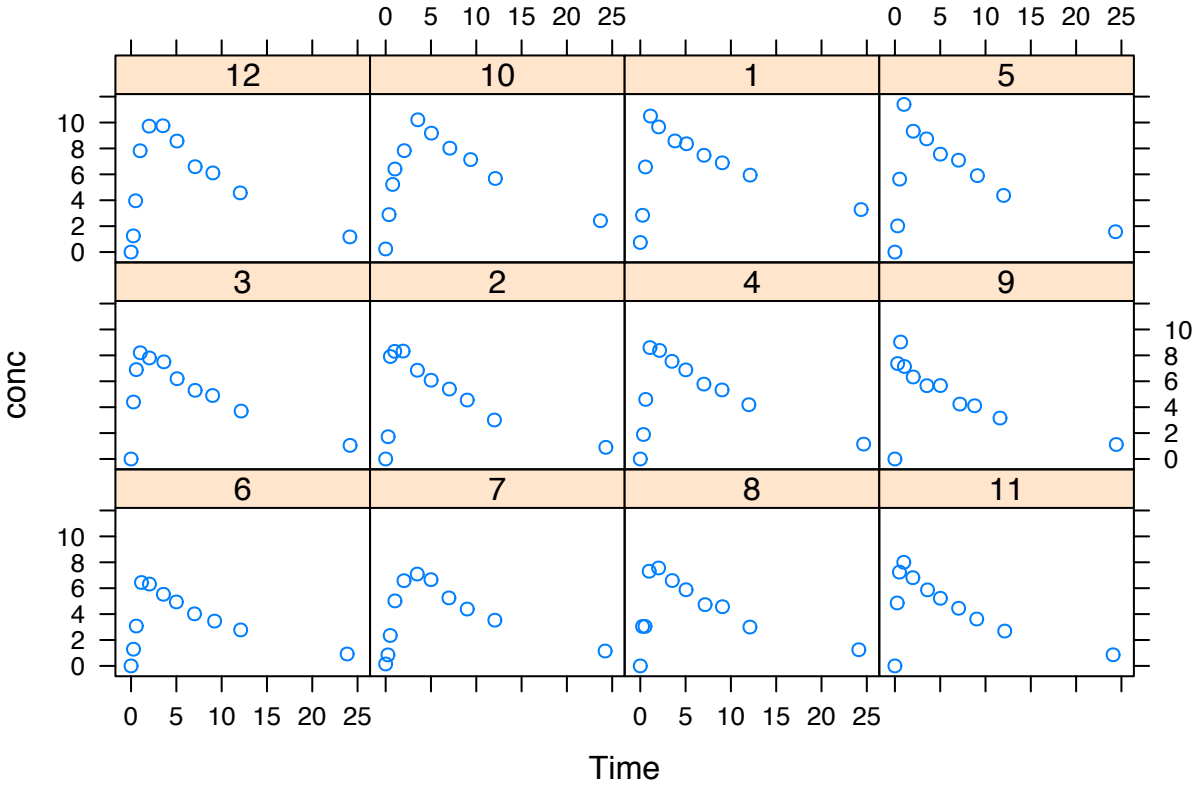
```
##     Subject   Wt Dose  Time conc
## 122      12 60.5  5.3  0.00 0.00
## 123      12 60.5  5.3  0.25 1.25
## 124      12 60.5  5.3  0.50 3.96
## 125      12 60.5  5.3  1.00 7.82
## 126      12 60.5  5.3  2.00 9.72
## 127      12 60.5  5.3  3.52 9.75
## 128      12 60.5  5.3  5.07 8.57
## 129      12 60.5  5.3  7.07 6.59
## 130      12 60.5  5.3  9.03 6.11
## 131      12 60.5  5.3 12.05 4.57
## 132      12 60.5  5.3 24.15 1.17
```

R console에서 ?Theoph를 타이핑 치면 좀 더 자세한 정보를 얻을 수 있습니다.

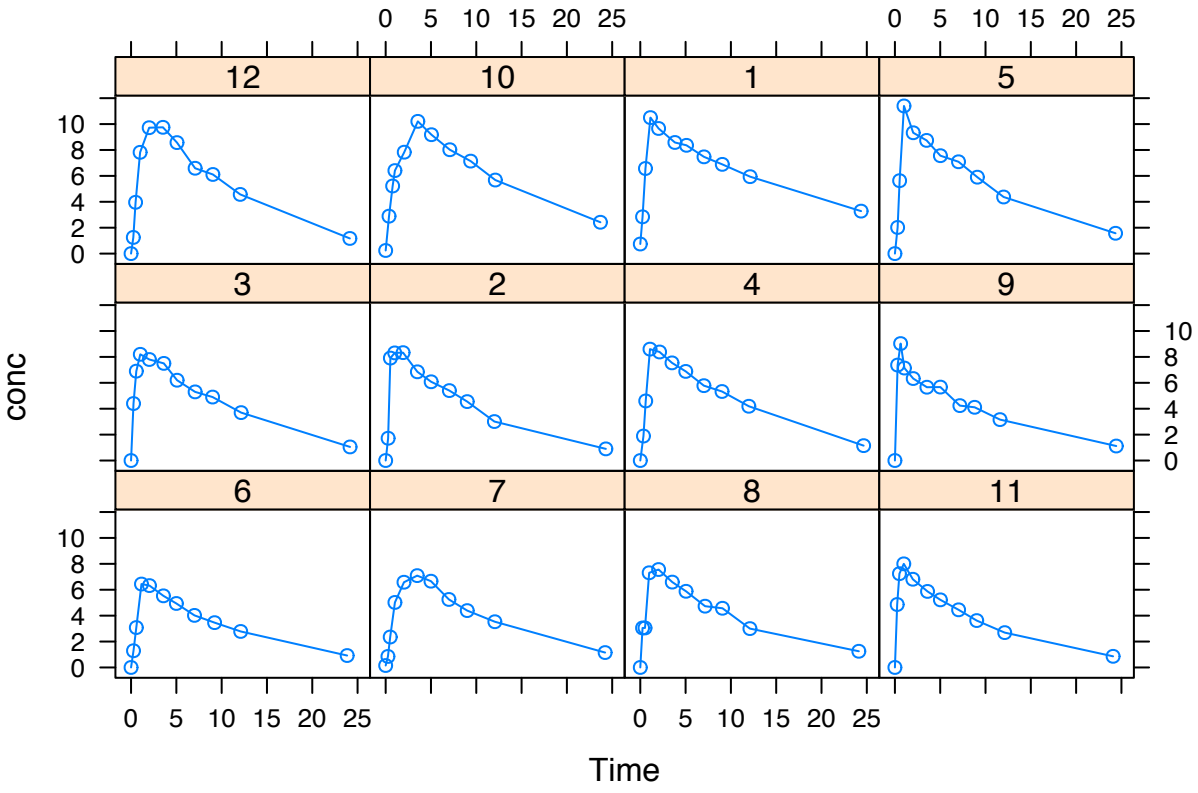
3.3 lattice

lattice 패키지를 불러온 뒤 그림을 그려보겠습니다. (Sarkar, 2017)

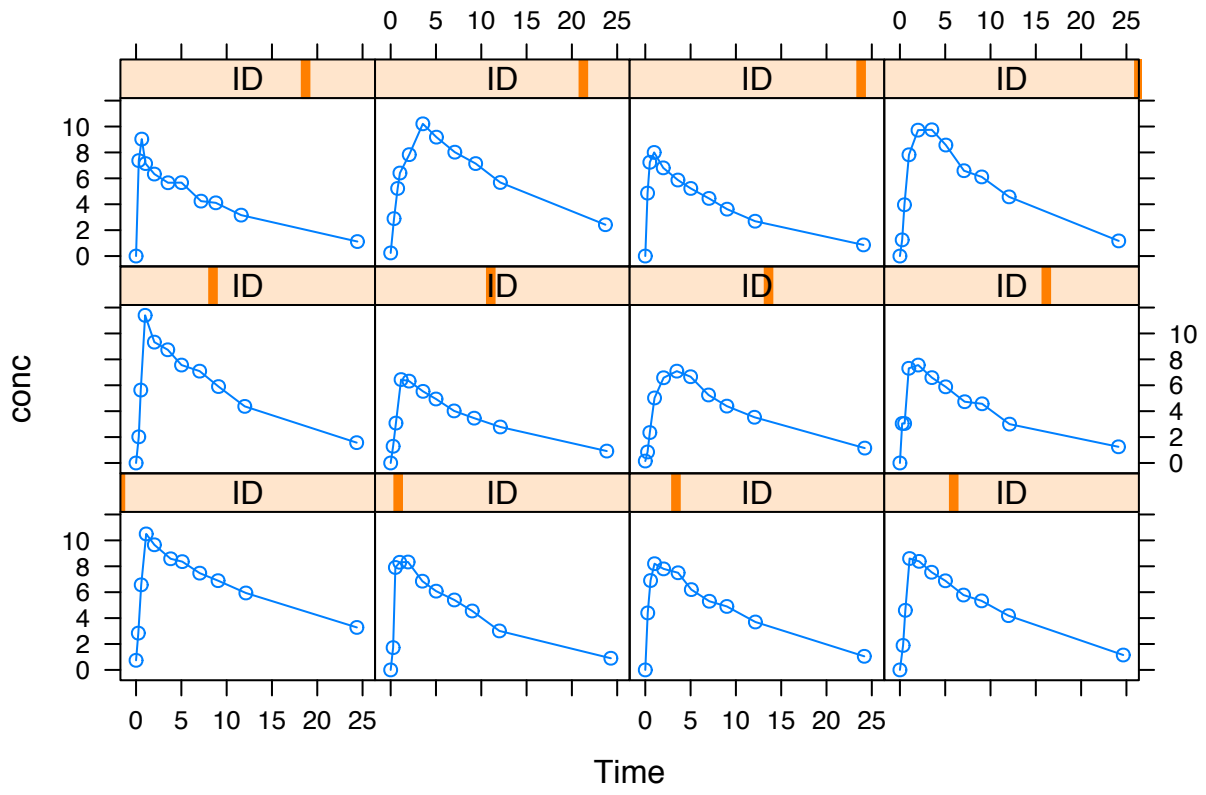
```
library(lattice) # trellis

xyplot(conc ~ Time | Subject, data=Theoph)
```



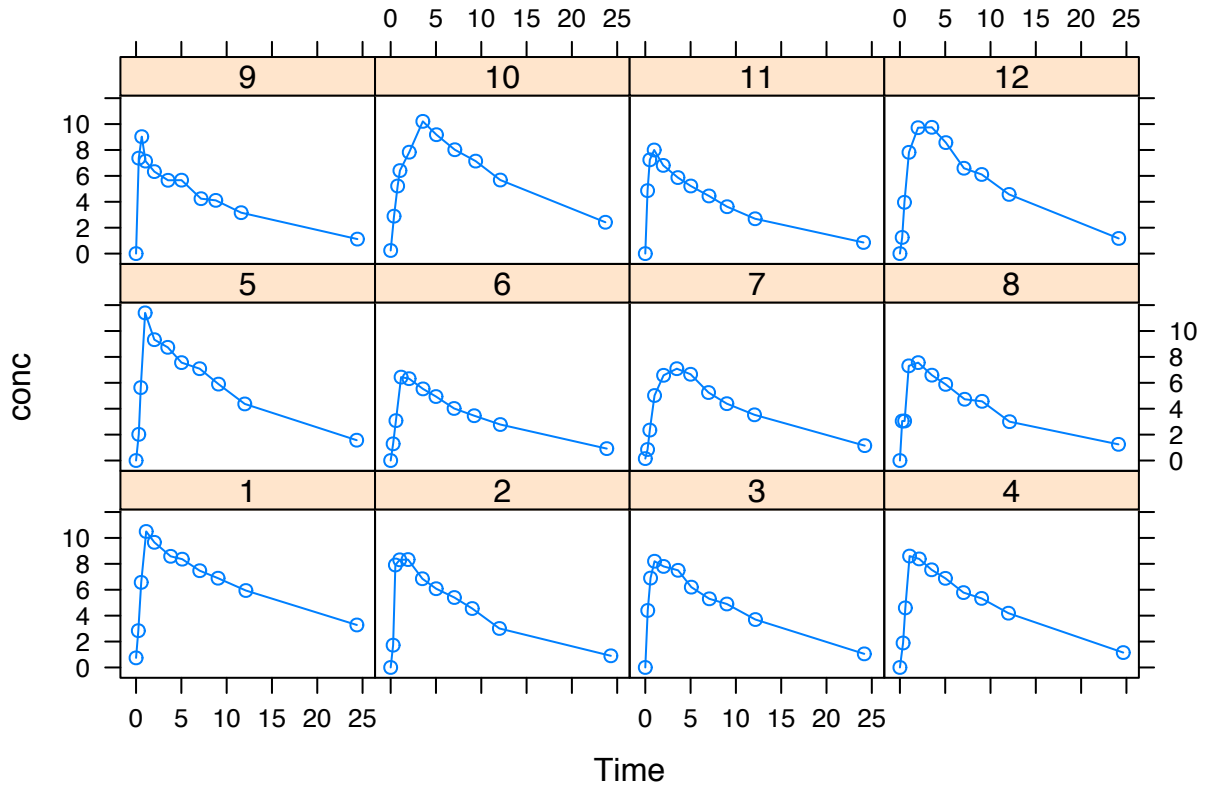

```
xyplot(conc ~ Time | Subject, data=Theoph, type="b")
```




```
Theoph[,"ID"] = as.numeric(as.character(Theoph[,"Subject"]))

xyplot(conc ~ Time | ID, data=Theoph, type="b")
```




```
xyplot(conc ~ Time | as.factor(ID), data=Theoph, type="b")
```



```
write.csv(Theoph, "Theoph.csv", row.names=FALSE, quote=FALSE, na="")
```

3.4 Subseting and write.csv

자료를 편집하고, subset을 만들고 각각을 파일로 저장하는 방법에 대해 알아보겠습니다.

```
IDs = sort(unique(Theoph[,"ID"])) ; IDs
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12
```

```
nID = length(IDs) ; nID
```

```
## [1] 12
```

```
demog = unique(Theoph[,c("ID","Wt")])
colnames(demog) = c("ID", "BWT")
write.csv(demog, "1-demog.csv", row.names=FALSE, quote=FALSE, na="")

DV = Theoph[,c("ID","Time", "conc")]
colnames(DV) = c("ID", "TIME", "DV")
write.csv(DV, "3-DV.csv", row.names=FALSE, quote=FALSE, na="")

adm = cbind(IDs, rep(0, nID), rep(320, nID))
colnames(adm) = c("ID", "TIME", "AMT")
write.csv(adm, "2-adm.csv", row.names=FALSE, quote=FALSE, na="")

demog = read.csv("1-demog.csv", as.is=TRUE)
adm = read.csv("2-adm.csv", as.is=TRUE)
dv = read.csv("3-dv.csv", as.is=TRUE)

AdmDv = merge(adm, dv, by=intersect(colnames(adm), colnames(dv)), all=TRUE)
AdmDv
```

```
##     ID  TIME AMT    DV
## 1    1  0.00 320  0.74
## 2    1  0.25  NA  2.84
## 3    1  0.57  NA  6.57
## 4    1  1.12  NA 10.50
## 5    1  2.02  NA  9.66
```

```
## 6    1  3.82  NA  8.58
## 7    1  5.10  NA  8.36
## 8    1  7.03  NA  7.47
## 9    1  9.05  NA  6.89
## 10   1 12.12  NA  5.94
## 11   1 24.37  NA  3.28
## 12   2  0.00 320  0.00
## 13   2  0.27  NA  1.72
## 14   2  0.52  NA  7.91
## 15   2  1.00  NA  8.31
## 16   2  1.92  NA  8.33
## 17   2  3.50  NA  6.85
## 18   2  5.02  NA  6.08
## 19   2  7.03  NA  5.40
## 20   2  9.00  NA  4.55
## 21   2 12.00  NA  3.01
## 22   2 24.30  NA  0.90
## 23   3  0.00 320  0.00
## 24   3  0.27  NA  4.40
## 25   3  0.58  NA  6.90
## 26   3  1.02  NA  8.20
## 27   3  2.02  NA  7.80
## 28   3  3.62  NA  7.50
## 29   3  5.08  NA  6.20
## 30   3  7.07  NA  5.30
## 31   3  9.00  NA  4.90
## 32   3 12.15  NA  3.70
## 33   3 24.17  NA  1.05
## 34   4  0.00 320  0.00
## 35   4  0.35  NA  1.89
## 36   4  0.60  NA  4.60
## 37   4  1.07  NA  8.60
## 38   4  2.13  NA  8.38
## 39   4  3.50  NA  7.54
## 40   4  5.02  NA  6.88
## 41   4  7.02  NA  5.78
## 42   4  9.02  NA  5.33
## 43   4 11.98  NA  4.19
## 44   4 24.65  NA  1.15
## 45   5  0.00 320  0.00
## 46   5  0.30  NA  2.02
## 47   5  0.52  NA  5.63
## 48   5  1.00  NA 11.40
## 49   5  2.02  NA  9.33
## 50   5  3.50  NA  8.74
```

```
## 51   5  5.02  NA  7.56
## 52   5  7.02  NA  7.09
## 53   5  9.10  NA  5.90
## 54   5 12.00  NA  4.37
## 55   5 24.35  NA  1.57
## 56   6  0.00 320  0.00
## 57   6  0.27  NA  1.29
## 58   6  0.58  NA  3.08
## 59   6  1.15  NA  6.44
## 60   6  2.03  NA  6.32
## 61   6  3.57  NA  5.53
## 62   6  5.00  NA  4.94
## 63   6  7.00  NA  4.02
## 64   6  9.22  NA  3.46
## 65   6 12.10  NA  2.78
## 66   6 23.85  NA  0.92
## 67   7  0.00 320  0.15
## 68   7  0.25  NA  0.85
## 69   7  0.50  NA  2.35
## 70   7  1.02  NA  5.02
## 71   7  2.02  NA  6.58
## 72   7  3.48  NA  7.09
## 73   7  5.00  NA  6.66
## 74   7  6.98  NA  5.25
## 75   7  9.00  NA  4.39
## 76   7 12.05  NA  3.53
## 77   7 24.22  NA  1.15
## 78   8  0.00 320  0.00
## 79   8  0.25  NA  3.05
## 80   8  0.52  NA  3.05
## 81   8  0.98  NA  7.31
## 82   8  2.02  NA  7.56
## 83   8  3.53  NA  6.59
## 84   8  5.05  NA  5.88
## 85   8  7.15  NA  4.73
## 86   8  9.07  NA  4.57
## 87   8 12.10  NA  3.00
## 88   8 24.12  NA  1.25
## 89   9  0.00 320  0.00
## 90   9  0.30  NA  7.37
## 91   9  0.63  NA  9.03
## 92   9  1.05  NA  7.14
## 93   9  2.02  NA  6.33
## 94   9  3.53  NA  5.66
## 95   9  5.02  NA  5.67
```

```
## 96   9  7.17  NA  4.24
## 97   9  8.80  NA  4.11
## 98   9 11.60  NA  3.16
## 99   9 24.43  NA  1.12
## 100 10  0.00 320  0.24
## 101 10  0.37  NA  2.89
## 102 10  0.77  NA  5.22
## 103 10  1.02  NA  6.41
## 104 10  2.05  NA  7.83
## 105 10  3.55  NA 10.21
## 106 10  5.05  NA  9.18
## 107 10  7.08  NA  8.02
## 108 10  9.38  NA  7.14
## 109 10 12.10  NA  5.68
## 110 10 23.70  NA  2.42
## 111 11  0.00 320  0.00
## 112 11  0.25  NA  4.86
## 113 11  0.50  NA  7.24
## 114 11  0.98  NA  8.00
## 115 11  1.98  NA  6.81
## 116 11  3.60  NA  5.87
## 117 11  5.02  NA  5.22
## 118 11  7.03  NA  4.45
## 119 11  9.03  NA  3.62
## 120 11 12.12  NA  2.69
## 121 11 24.08  NA  0.86
## 122 12  0.00 320  0.00
## 123 12  0.25  NA  1.25
## 124 12  0.50  NA  3.96
## 125 12  1.00  NA  7.82
## 126 12  2.00  NA  9.72
## 127 12  3.52  NA  9.75
## 128 12  5.07  NA  8.57
## 129 12  7.07  NA  6.59
## 130 12  9.03  NA  6.11
## 131 12 12.05  NA  4.57
## 132 12 24.15  NA  1.17
```

자료를 병합(merge)해 보겠습니다.

```
DataAll = merge(demog, AdmDv, by=c("ID"), all=TRUE)
DataAll
```

```
##     ID  BWT  TIME AMT   DV
## 1    1 79.6  0.00 320  0.74
```

```
## 2    1 79.6  0.25  NA  2.84
## 3    1 79.6  0.57  NA  6.57
## 4    1 79.6  1.12  NA 10.50
## 5    1 79.6  2.02  NA  9.66
## 6    1 79.6  3.82  NA  8.58
## 7    1 79.6  5.10  NA  8.36
## 8    1 79.6  7.03  NA  7.47
## 9    1 79.6  9.05  NA  6.89
## 10   1 79.6 12.12  NA  5.94
## 11   1 79.6 24.37  NA  3.28
## 12   2 72.4  0.00 320  0.00
## 13   2 72.4  0.27  NA  1.72
## 14   2 72.4  0.52  NA  7.91
## 15   2 72.4  1.00  NA  8.31
## 16   2 72.4  1.92  NA  8.33
## 17   2 72.4  3.50  NA  6.85
## 18   2 72.4  5.02  NA  6.08
## 19   2 72.4  7.03  NA  5.40
## 20   2 72.4  9.00  NA  4.55
## 21   2 72.4 12.00  NA  3.01
## 22   2 72.4 24.30  NA  0.90
## 23   3 70.5  0.00 320  0.00
## 24   3 70.5  0.27  NA  4.40
## 25   3 70.5  0.58  NA  6.90
## 26   3 70.5  1.02  NA  8.20
## 27   3 70.5  2.02  NA  7.80
## 28   3 70.5  3.62  NA  7.50
## 29   3 70.5  5.08  NA  6.20
## 30   3 70.5  7.07  NA  5.30
## 31   3 70.5  9.00  NA  4.90
## 32   3 70.5 12.15  NA  3.70
## 33   3 70.5 24.17  NA  1.05
## 34   4 72.7  0.00 320  0.00
## 35   4 72.7  0.35  NA  1.89
## 36   4 72.7  0.60  NA  4.60
## 37   4 72.7  1.07  NA  8.60
## 38   4 72.7  2.13  NA  8.38
## 39   4 72.7  3.50  NA  7.54
## 40   4 72.7  5.02  NA  6.88
## 41   4 72.7  7.02  NA  5.78
## 42   4 72.7  9.02  NA  5.33
## 43   4 72.7 11.98  NA  4.19
## 44   4 72.7 24.65  NA  1.15
## 45   5 54.6  0.00 320  0.00
## 46   5 54.6  0.30  NA  2.02
```

```
## 47   5 54.6  0.52  NA  5.63
## 48   5 54.6  1.00  NA 11.40
## 49   5 54.6  2.02  NA  9.33
## 50   5 54.6  3.50  NA  8.74
## 51   5 54.6  5.02  NA  7.56
## 52   5 54.6  7.02  NA  7.09
## 53   5 54.6  9.10  NA  5.90
## 54   5 54.6 12.00  NA  4.37
## 55   5 54.6 24.35  NA  1.57
## 56   6 80.0  0.00 320  0.00
## 57   6 80.0  0.27  NA  1.29
## 58   6 80.0  0.58  NA  3.08
## 59   6 80.0  1.15  NA  6.44
## 60   6 80.0  2.03  NA  6.32
## 61   6 80.0  3.57  NA  5.53
## 62   6 80.0  5.00  NA  4.94
## 63   6 80.0  7.00  NA  4.02
## 64   6 80.0  9.22  NA  3.46
## 65   6 80.0 12.10  NA  2.78
## 66   6 80.0 23.85  NA  0.92
## 67   7 64.6  0.00 320  0.15
## 68   7 64.6  0.25  NA  0.85
## 69   7 64.6  0.50  NA  2.35
## 70   7 64.6  1.02  NA  5.02
## 71   7 64.6  2.02  NA  6.58
## 72   7 64.6  3.48  NA  7.09
## 73   7 64.6  5.00  NA  6.66
## 74   7 64.6  6.98  NA  5.25
## 75   7 64.6  9.00  NA  4.39
## 76   7 64.6 12.05  NA  3.53
## 77   7 64.6 24.22  NA  1.15
## 78   8 70.5  0.00 320  0.00
## 79   8 70.5  0.25  NA  3.05
## 80   8 70.5  0.52  NA  3.05
## 81   8 70.5  0.98  NA  7.31
## 82   8 70.5  2.02  NA  7.56
## 83   8 70.5  3.53  NA  6.59
## 84   8 70.5  5.05  NA  5.88
## 85   8 70.5  7.15  NA  4.73
## 86   8 70.5  9.07  NA  4.57
## 87   8 70.5 12.10  NA  3.00
## 88   8 70.5 24.12  NA  1.25
## 89   9 86.4  0.00 320  0.00
## 90   9 86.4  0.30  NA  7.37
## 91   9 86.4  0.63  NA  9.03
```

```
## 92   9 86.4  1.05  NA  7.14
## 93   9 86.4  2.02  NA  6.33
## 94   9 86.4  3.53  NA  5.66
## 95   9 86.4  5.02  NA  5.67
## 96   9 86.4  7.17  NA  4.24
## 97   9 86.4  8.80  NA  4.11
## 98   9 86.4 11.60  NA  3.16
## 99   9 86.4 24.43  NA  1.12
## 100 10 58.2  0.00 320  0.24
## 101 10 58.2  0.37  NA  2.89
## 102 10 58.2  0.77  NA  5.22
## 103 10 58.2  1.02  NA  6.41
## 104 10 58.2  2.05  NA  7.83
## 105 10 58.2  3.55  NA 10.21
## 106 10 58.2  5.05  NA  9.18
## 107 10 58.2  7.08  NA  8.02
## 108 10 58.2  9.38  NA  7.14
## 109 10 58.2 12.10  NA  5.68
## 110 10 58.2 23.70  NA  2.42
## 111 11 65.0  0.00 320  0.00
## 112 11 65.0  0.25  NA  4.86
## 113 11 65.0  0.50  NA  7.24
## 114 11 65.0  0.98  NA  8.00
## 115 11 65.0  1.98  NA  6.81
## 116 11 65.0  3.60  NA  5.87
## 117 11 65.0  5.02  NA  5.22
## 118 11 65.0  7.03  NA  4.45
## 119 11 65.0  9.03  NA  3.62
## 120 11 65.0 12.12  NA  2.69
## 121 11 65.0 24.08  NA  0.86
## 122 12 60.5  0.00 320  0.00
## 123 12 60.5  0.25  NA  1.25
## 124 12 60.5  0.50  NA  3.96
## 125 12 60.5  1.00  NA  7.82
## 126 12 60.5  2.00  NA  9.72
## 127 12 60.5  3.52  NA  9.75
## 128 12 60.5  5.07  NA  8.57
## 129 12 60.5  7.07  NA  6.59
## 130 12 60.5  9.03  NA  6.11
## 131 12 60.5 12.05  NA  4.57
## 132 12 60.5 24.15  NA  1.17
```

4

Frequently Used Functions

2017-04-05 배균섭 교수님 강의

자주 쓰는 함수 및 명령어에 대해 알아보겠습니다.

4.1 Command

```
# 2017-04-05 R-intro.pdf Chapter 08

pois
```

```
## Error in eval(expr, envir, enclos): 객체 'pois'를 찾을 수 없습니다
```

```
# ?dbeta
dnorm(0)
```

```
## [1] 0.3989
```

```
pnorm(0)
```

```
## [1] 0.5
```

```
1 - pnorm(1.96)
```

```
## [1] 0.025
```

```r
# ?pnorm
pnorm(1.96, lower.tail=FALSE)
```

```
## [1] 0.025
```

```r
qnorm(0.5)
```

```
## [1] 0
```

```r
qnorm(0.975)
```

```
## [1] 1.96
```

```r
format(qnorm(0.975), digits=22)
```

```
## [1] "1.959963984540053605343"
```

```r
rnorm(5)
```

```
## [1] -1.0995  0.3497  0.4523  0.9362 -0.3453
```

```r
rnorm(5, 10, 1)
```

```
## [1] 10.01 10.78 10.29  9.76 10.42
```

```r
x = rnorm(100, 10, 1)
mean(x)
```

```
## [1] 10.13
```

```r
sd(x)
```

```
## [1] 0.9674
```

```r
2*pt(-2.43, df = 13)
```

```
## [1] 0.03033
```

```r
2*pt(-2.43, df = 1000)
```

```
## [1] 0.01527
```

```
qnorm(0.995)
```

```
## [1] 2.576
```

```
qf(0.01, 2, 7, lower.tail = FALSE)
```

```
## [1] 9.547
```

```
# ?fivenum
faithful
```

```
##     eruptions waiting
## 1       3.600      79
## 2       1.800      54
## 3       3.333      74
## 4       2.283      62
## 5       4.533      85
## 6       2.883      55
## 7       4.700      88
## 8       3.600      85
## 9       1.950      51
## 10      4.350      85
## 11      1.833      54
## 12      3.917      84
## 13      4.200      78
## 14      1.750      47
## 15      4.700      83
## 16      2.167      52
## 17      1.750      62
## 18      4.800      84
## 19      1.600      52
## 20      4.250      79
## 21      1.800      51
## 22      1.750      47
## 23      3.450      78
## 24      3.067      69
## 25      4.533      74
## 26      3.600      83
## 27      1.967      55
## 28      4.083      76
## 29      3.850      78
## 30      4.433      79
## 31      4.300      73
## 32      4.467      77
```

```
## 33      3.367      66
## 34      4.033      80
## 35      3.833      74
## 36      2.017      52
## 37      1.867      48
## 38      4.833      80
## 39      1.833      59
## 40      4.783      90
## 41      4.350      80
## 42      1.883      58
## 43      4.567      84
## 44      1.750      58
## 45      4.533      73
## 46      3.317      83
## 47      3.833      64
## 48      2.100      53
## 49      4.633      82
## 50      2.000      59
## 51      4.800      75
## 52      4.716      90
## 53      1.833      54
## 54      4.833      80
## 55      1.733      54
## 56      4.883      83
## 57      3.717      71
## 58      1.667      64
## 59      4.567      77
## 60      4.317      81
## 61      2.233      59
## 62      4.500      84
## 63      1.750      48
## 64      4.800      82
## 65      1.817      60
## 66      4.400      92
## 67      4.167      78
## 68      4.700      78
## 69      2.067      65
## 70      4.700      73
## 71      4.033      82
## 72      1.967      56
## 73      4.500      79
## 74      4.000      71
## 75      1.983      62
## 76      5.067      76
## 77      2.017      60
```

```
## 78      4.567      78
## 79      3.883      76
## 80      3.600      83
## 81      4.133      75
## 82      4.333      82
## 83      4.100      70
## 84      2.633      65
## 85      4.067      73
## 86      4.933      88
## 87      3.950      76
## 88      4.517      80
## 89      2.167      48
## 90      4.000      86
## 91      2.200      60
## 92      4.333      90
## 93      1.867      50
## 94      4.817      78
## 95      1.833      63
## 96      4.300      72
## 97      4.667      84
## 98      3.750      75
## 99      1.867      51
## 100     4.900      82
## 101     2.483      62
## 102     4.367      88
## 103     2.100      49
## 104     4.500      83
## 105     4.050      81
## 106     1.867      47
## 107     4.700      84
## 108     1.783      52
## 109     4.850      86
## 110     3.683      81
## 111     4.733      75
## 112     2.300      59
## 113     4.900      89
## 114     4.417      79
## 115     1.700      59
## 116     4.633      81
## 117     2.317      50
## 118     4.600      85
## 119     1.817      59
## 120     4.417      87
## 121     2.617      53
## 122     4.067      69
```

```
## 123     4.250      77
## 124     1.967      56
## 125     4.600      88
## 126     3.767      81
## 127     1.917      45
## 128     4.500      82
## 129     2.267      55
## 130     4.650      90
## 131     1.867      45
## 132     4.167      83
## 133     2.800      56
## 134     4.333      89
## 135     1.833      46
## 136     4.383      82
## 137     1.883      51
## 138     4.933      86
## 139     2.033      53
## 140     3.733      79
## 141     4.233      81
## 142     2.233      60
## 143     4.533      82
## 144     4.817      77
## 145     4.333      76
## 146     1.983      59
## 147     4.633      80
## 148     2.017      49
## 149     5.100      96
## 150     1.800      53
## 151     5.033      77
## 152     4.000      77
## 153     2.400      65
## 154     4.600      81
## 155     3.567      71
## 156     4.000      70
## 157     4.500      81
## 158     4.083      93
## 159     1.800      53
## 160     3.967      89
## 161     2.200      45
## 162     4.150      86
## 163     2.000      58
## 164     3.833      78
## 165     3.500      66
## 166     4.583      76
## 167     2.367      63
```

```
## 168     5.000      88
## 169     1.933      52
## 170     4.617      93
## 171     1.917      49
## 172     2.083      57
## 173     4.583      77
## 174     3.333      68
## 175     4.167      81
## 176     4.333      81
## 177     4.500      73
## 178     2.417      50
## 179     4.000      85
## 180     4.167      74
## 181     1.883      55
## 182     4.583      77
## 183     4.250      83
## 184     3.767      83
## 185     2.033      51
## 186     4.433      78
## 187     4.083      84
## 188     1.833      46
## 189     4.417      83
## 190     2.183      55
## 191     4.800      81
## 192     1.833      57
## 193     4.800      76
## 194     4.100      84
## 195     3.966      77
## 196     4.233      81
## 197     3.500      87
## 198     4.366      77
## 199     2.250      51
## 200     4.667      78
## 201     2.100      60
## 202     4.350      82
## 203     4.133      91
## 204     1.867      53
## 205     4.600      78
## 206     1.783      46
## 207     4.367      77
## 208     3.850      84
## 209     1.933      49
## 210     4.500      83
## 211     2.383      71
## 212     4.700      80
```

```
## 213     1.867      49
## 214     3.833      75
## 215     3.417      64
## 216     4.233      76
## 217     2.400      53
## 218     4.800      94
## 219     2.000      55
## 220     4.150      76
## 221     1.867      50
## 222     4.267      82
## 223     1.750      54
## 224     4.483      75
## 225     4.000      78
## 226     4.117      79
## 227     4.083      78
## 228     4.267      78
## 229     3.917      70
## 230     4.550      79
## 231     4.083      70
## 232     2.417      54
## 233     4.183      86
## 234     2.217      50
## 235     4.450      90
## 236     1.883      54
## 237     1.850      54
## 238     4.283      77
## 239     3.950      79
## 240     2.333      64
## 241     4.150      75
## 242     2.350      47
## 243     4.933      86
## 244     2.900      63
## 245     4.583      85
## 246     3.833      82
## 247     2.083      57
## 248     4.367      82
## 249     2.133      67
## 250     4.350      74
## 251     2.200      54
## 252     4.450      83
## 253     3.567      73
## 254     4.500      73
## 255     4.150      88
## 256     3.817      80
## 257     3.917      71
```

```
## 258     4.450      83
## 259     2.000      56
## 260     4.283      79
## 261     4.767      78
## 262     4.533      84
## 263     1.850      58
## 264     4.250      83
## 265     1.983      43
## 266     2.250      60
## 267     4.750      75
## 268     4.117      81
## 269     2.150      46
## 270     4.417      90
## 271     1.817      46
## 272     4.467      74
```

```
str(faithful)
```

```
## 'data.frame':    272 obs. of  2 variables:
##  $ eruptions: num  3.6 1.8 3.33 2.28 4.53 ...
##  $ waiting  : num  79 54 74 62 85 55 88 85 51 85 ...
```

```
eruptions
```

```
## Error in eval(expr, envir, enclos): 객체 'eruptions'를 찾을 수 없습니다
```

```
attach(faithful)
eruptions
```

```
##   [1] 3.600 1.800 3.333 2.283 4.533 2.883 4.700 3.600
##   [9] 1.950 4.350 1.833 3.917 4.200 1.750 4.700 2.167
##  [17] 1.750 4.800 1.600 4.250 1.800 1.750 3.450 3.067
##  [25] 4.533 3.600 1.967 4.083 3.850 4.433 4.300 4.467
##  [33] 3.367 4.033 3.833 2.017 1.867 4.833 1.833 4.783
##  [41] 4.350 1.883 4.567 1.750 4.533 3.317 3.833 2.100
##  [49] 4.633 2.000 4.800 4.716 1.833 4.833 1.733 4.883
##  [57] 3.717 1.667 4.567 4.317 2.233 4.500 1.750 4.800
##  [65] 1.817 4.400 4.167 4.700 2.067 4.700 4.033 1.967
##  [73] 4.500 4.000 1.983 5.067 2.017 4.567 3.883 3.600
##  [81] 4.133 4.333 4.100 2.633 4.067 4.933 3.950 4.517
##  [89] 2.167 4.000 2.200 4.333 1.867 4.817 1.833 4.300
##  [97] 4.667 3.750 1.867 4.900 2.483 4.367 2.100 4.500
## [105] 4.050 1.867 4.700 1.783 4.850 3.683 4.733 2.300
## [113] 4.900 4.417 1.700 4.633 2.317 4.600 1.817 4.417
```

```
## [121] 2.617 4.067 4.250 1.967 4.600 3.767 1.917 4.500
## [129] 2.267 4.650 1.867 4.167 2.800 4.333 1.833 4.383
## [137] 1.883 4.933 2.033 3.733 4.233 2.233 4.533 4.817
## [145] 4.333 1.983 4.633 2.017 5.100 1.800 5.033 4.000
## [153] 2.400 4.600 3.567 4.000 4.500 4.083 1.800 3.967
## [161] 2.200 4.150 2.000 3.833 3.500 4.583 2.367 5.000
## [169] 1.933 4.617 1.917 2.083 4.583 3.333 4.167 4.333
## [177] 4.500 2.417 4.000 4.167 1.883 4.583 4.250 3.767
## [185] 2.033 4.433 4.083 1.833 4.417 2.183 4.800 1.833
## [193] 4.800 4.100 3.966 4.233 3.500 4.366 2.250 4.667
## [201] 2.100 4.350 4.133 1.867 4.600 1.783 4.367 3.850
## [209] 1.933 4.500 2.383 4.700 1.867 3.833 3.417 4.233
## [217] 2.400 4.800 2.000 4.150 1.867 4.267 1.750 4.483
## [225] 4.000 4.117 4.083 4.267 3.917 4.550 4.083 2.417
## [233] 4.183 2.217 4.450 1.883 1.850 4.283 3.950 2.333
## [241] 4.150 2.350 4.933 2.900 4.583 3.833 2.083 4.367
## [249] 2.133 4.350 2.200 4.450 3.567 4.500 4.150 3.817
## [257] 3.917 4.450 2.000 4.283 4.767 4.533 1.850 4.250
## [265] 1.983 2.250 4.750 4.117 2.150 4.417 1.817 4.467
```

```
waiting
```

```
##   [1] 79 54 74 62 85 55 88 85 51 85 54 84 78 47 83 52
##  [17] 62 84 52 79 51 47 78 69 74 83 55 76 78 79 73 77
##  [33] 66 80 74 52 48 80 59 90 80 58 84 58 73 83 64 53
##  [49] 82 59 75 90 54 80 54 83 71 64 77 81 59 84 48 82
##  [65] 60 92 78 78 65 73 82 56 79 71 62 76 60 78 76 83
##  [81] 75 82 70 65 73 88 76 80 48 86 60 90 50 78 63 72
##  [97] 84 75 51 82 62 88 49 83 81 47 84 52 86 81 75 59
## [113] 89 79 59 81 50 85 59 87 53 69 77 56 88 81 45 82
## [129] 55 90 45 83 56 89 46 82 51 86 53 79 81 60 82 77
## [145] 76 59 80 49 96 53 77 77 65 81 71 70 81 93 53 89
## [161] 45 86 58 78 66 76 63 88 52 93 49 57 77 68 81 81
## [177] 73 50 85 74 55 77 83 83 51 78 84 46 83 55 81 57
## [193] 76 84 77 81 87 77 51 78 60 82 91 53 78 46 77 84
## [209] 49 83 71 80 49 75 64 76 53 94 55 76 50 82 54 75
## [225] 78 79 78 78 70 79 70 54 86 50 90 54 54 77 79 64
## [241] 75 47 86 63 85 82 57 82 67 74 54 83 73 73 88 80
## [257] 71 83 56 79 78 84 58 83 43 60 75 81 46 90 46 74
```

```
stem(waiting)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
```

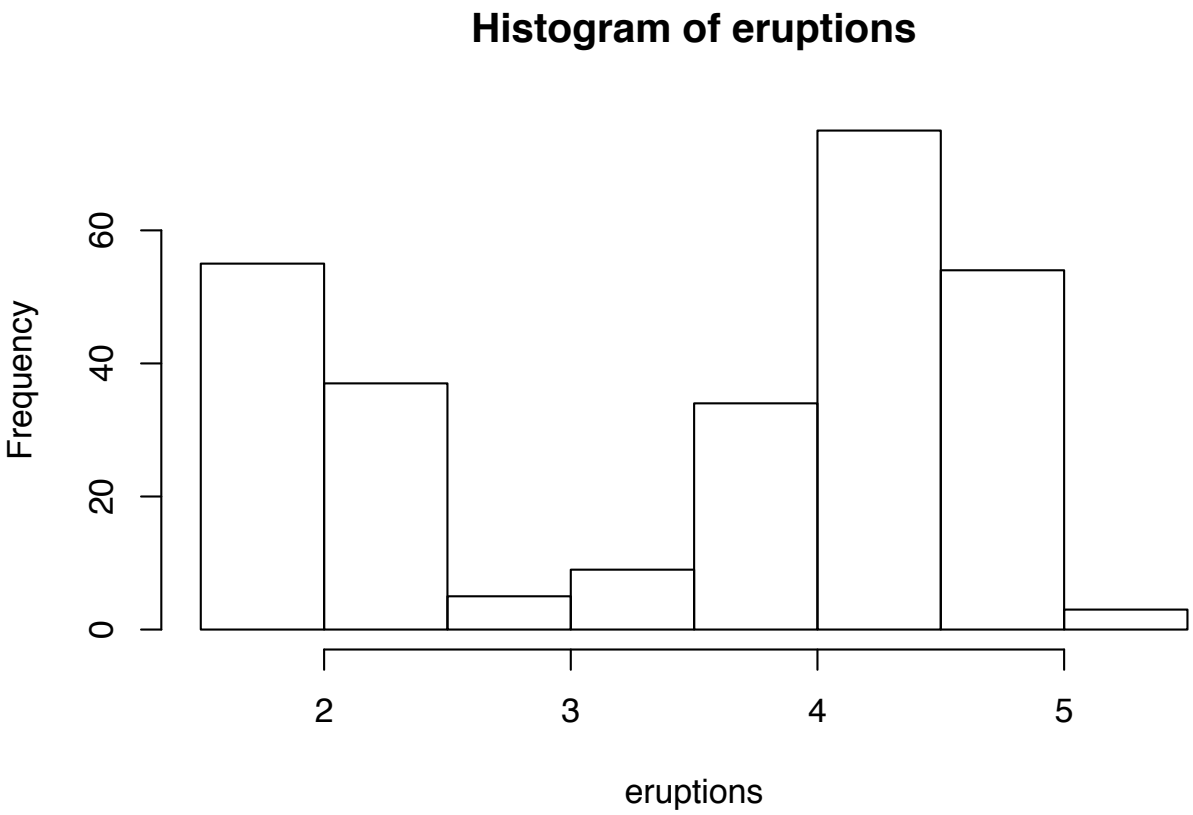
```
##   4 | 3
##   4 | 55566666777788899999
##   5 | 00000111111222223333333444444444
##   5 | 555555666677788889999999
##   6 | 00000022223334444
##   6 | 555667899
##   7 | 00001111123333333444444
##   7 | 55555555666666666777777777777888888888888888999999999
##   8 | 000000001111111111111222222222222333333333333334444444444
##   8 | 55555566666677888888999
##   9 | 00000012334
##   9 | 6
```

```
sort(eruptions)
```

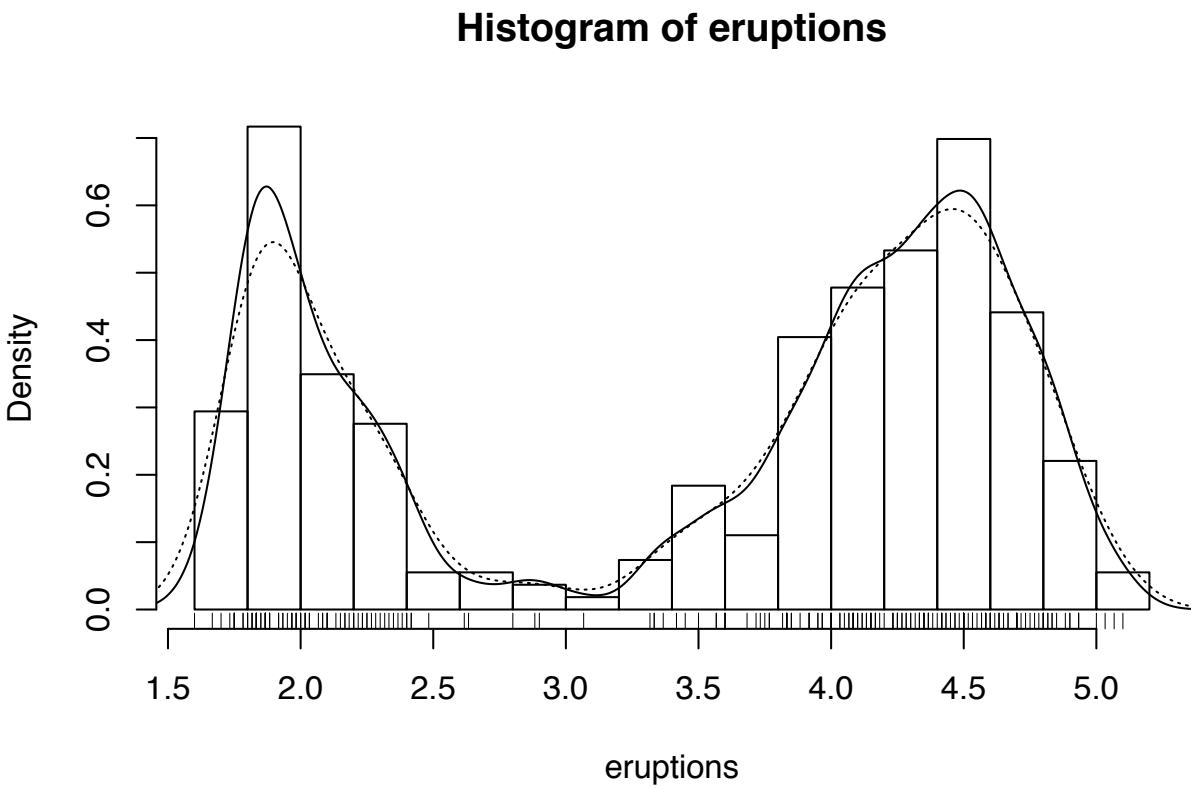
```
##   [1] 1.600 1.667 1.700 1.733 1.750 1.750 1.750 1.750
##   [9] 1.750 1.750 1.783 1.783 1.800 1.800 1.800 1.800
##  [17] 1.817 1.817 1.817 1.833 1.833 1.833 1.833 1.833
##  [25] 1.833 1.833 1.850 1.850 1.867 1.867 1.867 1.867
##  [33] 1.867 1.867 1.867 1.867 1.883 1.883 1.883 1.883
##  [41] 1.917 1.917 1.933 1.933 1.950 1.967 1.967 1.967
##  [49] 1.983 1.983 1.983 2.000 2.000 2.000 2.000 2.017
##  [57] 2.017 2.017 2.033 2.033 2.067 2.083 2.083 2.100
##  [65] 2.100 2.100 2.133 2.150 2.167 2.167 2.183 2.200
##  [73] 2.200 2.200 2.217 2.233 2.233 2.250 2.250 2.267
##  [81] 2.283 2.300 2.317 2.333 2.350 2.367 2.383 2.400
##  [89] 2.400 2.417 2.417 2.483 2.617 2.633 2.800 2.883
##  [97] 2.900 3.067 3.317 3.333 3.333 3.367 3.417 3.450
## [105] 3.500 3.500 3.567 3.567 3.600 3.600 3.600 3.600
## [113] 3.683 3.717 3.733 3.750 3.767 3.767 3.817 3.833
## [121] 3.833 3.833 3.833 3.833 3.850 3.850 3.883 3.917
## [129] 3.917 3.917 3.950 3.950 3.966 3.967 4.000 4.000
## [137] 4.000 4.000 4.000 4.000 4.033 4.033 4.050 4.067
## [145] 4.067 4.083 4.083 4.083 4.083 4.083 4.100 4.100
## [153] 4.117 4.117 4.133 4.133 4.150 4.150 4.150 4.150
## [161] 4.167 4.167 4.167 4.167 4.183 4.200 4.233 4.233
## [169] 4.233 4.250 4.250 4.250 4.250 4.267 4.267 4.283
## [177] 4.283 4.300 4.300 4.317 4.333 4.333 4.333 4.333
## [185] 4.333 4.350 4.350 4.350 4.350 4.366 4.367 4.367
## [193] 4.367 4.383 4.400 4.417 4.417 4.417 4.417 4.433
## [201] 4.433 4.450 4.450 4.450 4.467 4.467 4.483 4.500
## [209] 4.500 4.500 4.500 4.500 4.500 4.500 4.500 4.517
## [217] 4.533 4.533 4.533 4.533 4.533 4.550 4.567 4.567
## [225] 4.567 4.583 4.583 4.583 4.583 4.600 4.600 4.600
## [233] 4.600 4.617 4.633 4.633 4.633 4.650 4.667 4.667
```

```
## [241] 4.700 4.700 4.700 4.700 4.700 4.700 4.716 4.733
## [249] 4.750 4.767 4.783 4.800 4.800 4.800 4.800 4.800
## [257] 4.800 4.817 4.817 4.833 4.833 4.850 4.883 4.900
## [265] 4.900 4.933 4.933 4.933 5.000 5.033 5.067 5.100
```

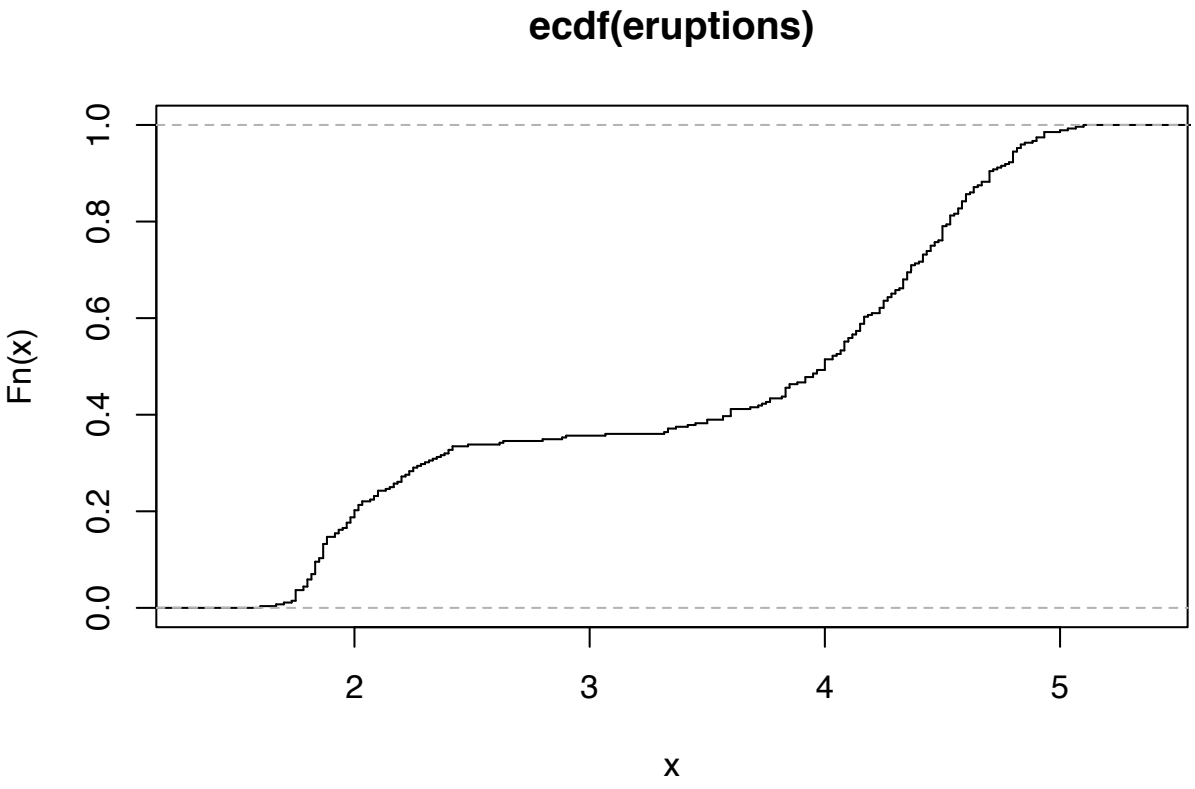
```
hist(eruptions)
```




```
hist(eruptions, seq(1.6, 5.2, 0.2), prob=TRUE)
lines(density(eruptions, bw=0.1))
rug(eruptions)
# ?hist
# ?density
lines(density(eruptions, bw="SJ"), lty=3)
```




```
plot(ecdf(eruptions), do.points=FALSE, verticals=TRUE)
```




```
# ?plot
ecdf(eruptions)
```

```
## Empirical CDF
```

```
## Call: ecdf(eruptions)
##  x[1:126] = 1.6, 1.7, 1.7,  ..., 5.1, 5.1
```

```
x = ecdf(eruptions)
x
```

```
## Empirical CDF
## Call: ecdf(eruptions)
##  x[1:126] = 1.6, 1.7, 1.7,  ..., 5.1, 5.1
```

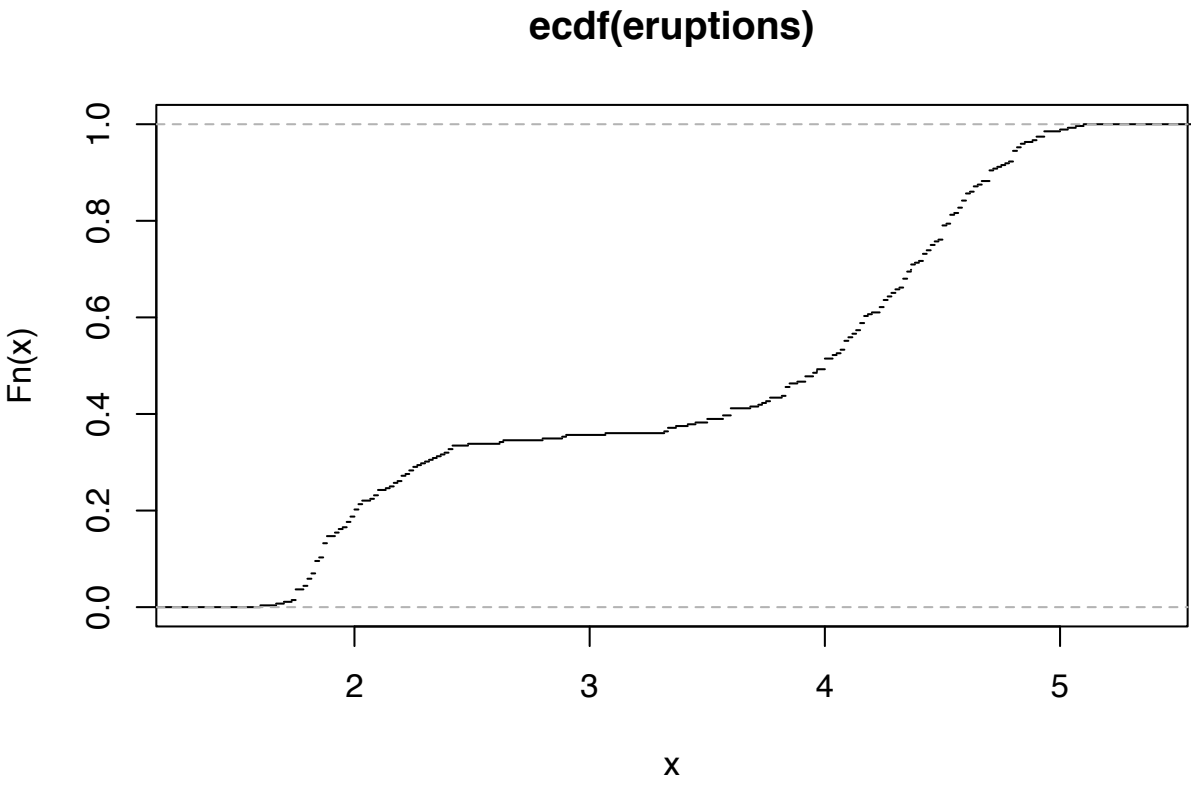
```
str(x)
```

```
## function (v)
##  - attr(*, "class")= chr [1:3] "ecdf" "stepfun" "function"
##  - attr(*, "call")= language ecdf(eruptions)
```

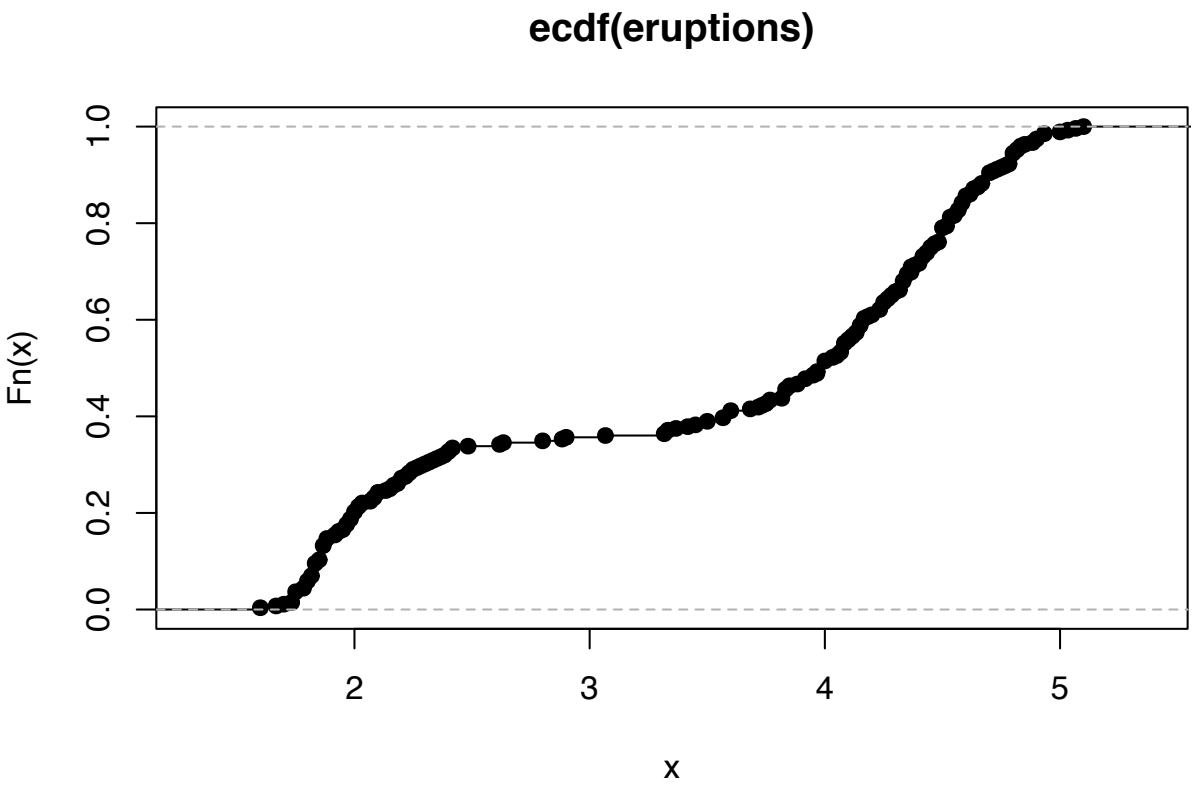
```
x()
```

```
## Error in .approxfun(x, y, v, method, yleft, yright, f): 기본값이 없는 인수 "v"가 누락되어 있
```

```
plot(ecdf(eruptions), do.points=FALSE)
```




```
plot(ecdf(eruptions))
```

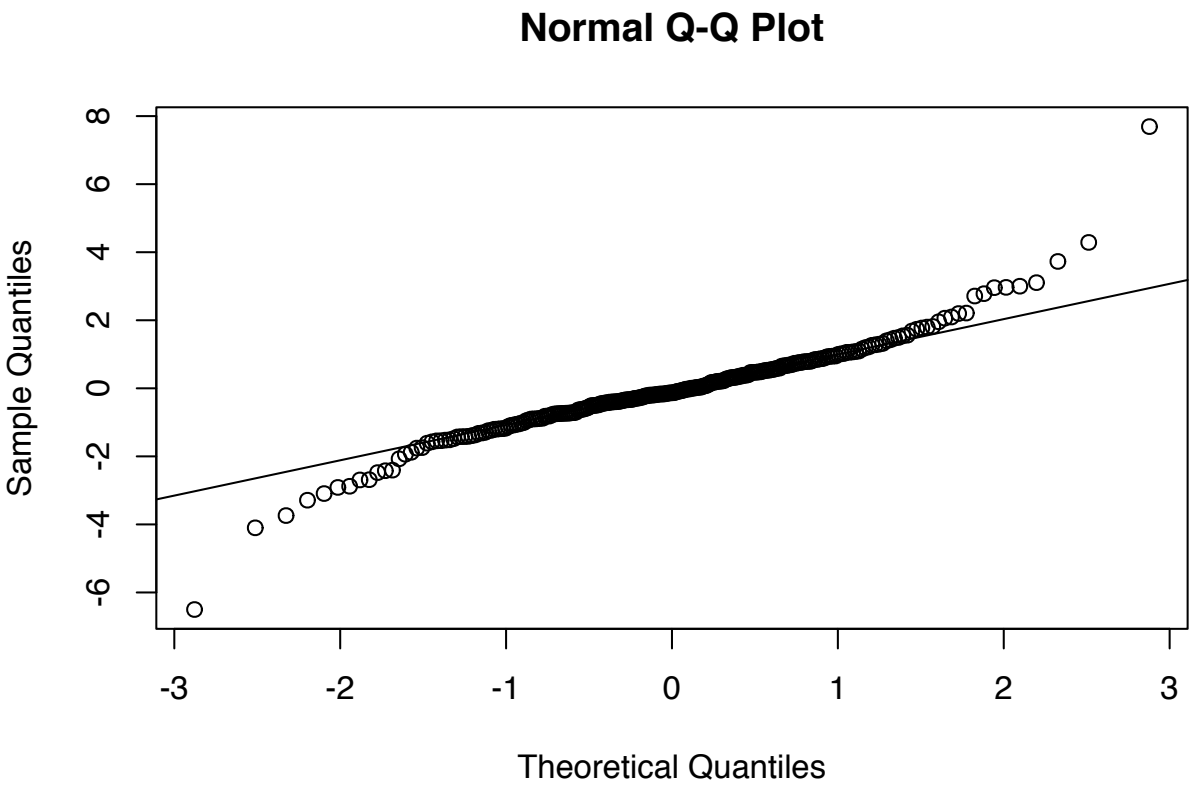



```
long <- eruptions[eruptions > 3]
x <- seq(3, 5.4, 0.01)
pnorm(x, mean=mean(long), sd=sqrt(var(long)))
```

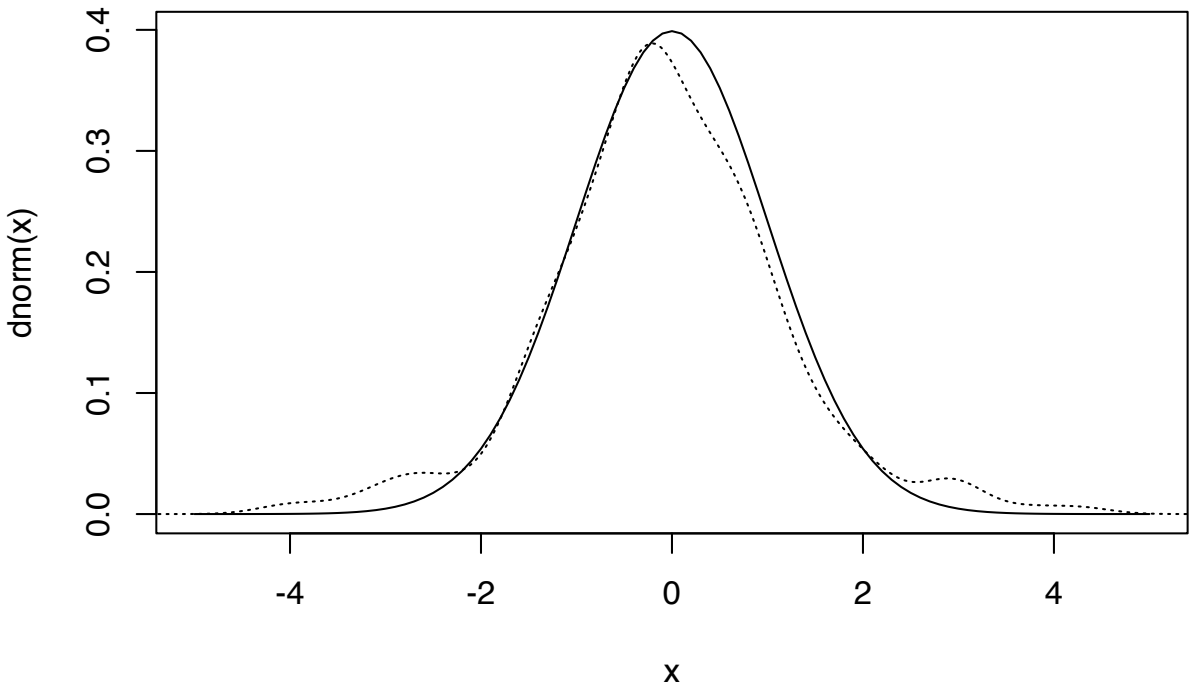
```
##   [1] 0.0008362 0.0009084 0.0009864 0.0010704 0.0011610
##   [6] 0.0012585 0.0013635 0.0014764 0.0015978 0.0017282
##  [11] 0.0018682 0.0020185 0.0021797 0.0023524 0.0025375
##  [16] 0.0027356 0.0029476 0.0031743 0.0034165 0.0036752
##  [21] 0.0039514 0.0042460 0.0045601 0.0048947 0.0052511
##  [26] 0.0056304 0.0060338 0.0064627 0.0069183 0.0074020
##  [31] 0.0079152 0.0084596 0.0090365 0.0096475 0.0102944
##  [36] 0.0109788 0.0117024 0.0124670 0.0132746 0.0141269
##  [41] 0.0150260 0.0159739 0.0169725 0.0180241 0.0191306
##  [46] 0.0202945 0.0215177 0.0228028 0.0241519 0.0255674
##  [51] 0.0270518 0.0286074 0.0302366 0.0319421 0.0337262
##  [56] 0.0355915 0.0375406 0.0395759 0.0417001 0.0439157
##  [61] 0.0462253 0.0486315 0.0511367 0.0537436 0.0564547
##  [66] 0.0592723 0.0621991 0.0652374 0.0683897 0.0716581
##  [71] 0.0750451 0.0785529 0.0821835 0.0859391 0.0898217
##  [76] 0.0938331 0.0979753 0.1022500 0.1066587 0.1112030
##  [81] 0.1158842 0.1207037 0.1256626 0.1307619 0.1360025
##  [86] 0.1413850 0.1469102 0.1525783 0.1583896 0.1643443
##  [91] 0.1704423 0.1766832 0.1830667 0.1895922 0.1962589
##  [96] 0.2030658 0.2100116 0.2170952 0.2243149 0.2316689
## [101] 0.2391554 0.2467722 0.2545170 0.2623872 0.2703803
```

```
## [106] 0.2784932 0.2867229 0.2950662 0.3035195 0.3120794
## [111] 0.3207419 0.3295032 0.3383590 0.3473052 0.3563373
## [116] 0.3654507 0.3746407 0.3839025 0.3932310 0.4026213
## [121] 0.4120681 0.4215661 0.4311100 0.4406943 0.4503134
## [126] 0.4599618 0.4696339 0.4793239 0.4890262 0.4987349
## [131] 0.5084444 0.5181489 0.5278427 0.5375199 0.5471751
## [136] 0.5568024 0.5663963 0.5759512 0.5854617 0.5949224
## [141] 0.6043279 0.6136730 0.6229527 0.6321619 0.6412957
## [146] 0.6503494 0.6593184 0.6681982 0.6769845 0.6856732
## [151] 0.6942601 0.7027416 0.7111139 0.7193735 0.7275172
## [156] 0.7355417 0.7434443 0.7512220 0.7588724 0.7663930
## [161] 0.7737818 0.7810366 0.7881558 0.7951377 0.8019809
## [166] 0.8086842 0.8152465 0.8216671 0.8279453 0.8340805
## [171] 0.8400726 0.8459213 0.8516267 0.8571890 0.8626087
## [176] 0.8678862 0.8730222 0.8780176 0.8828733 0.8875905
## [181] 0.8921703 0.8966142 0.9009236 0.9051002 0.9091456
## [186] 0.9130615 0.9168500 0.9205130 0.9240526 0.9274708
## [191] 0.9307700 0.9339522 0.9370200 0.9399756 0.9428215
## [196] 0.9455601 0.9481939 0.9507254 0.9531571 0.9554916
## [201] 0.9577315 0.9598792 0.9619375 0.9639088 0.9657957
## [206] 0.9676007 0.9693264 0.9709753 0.9725498 0.9740525
## [211] 0.9754857 0.9768519 0.9781534 0.9793926 0.9805717
## [216] 0.9816930 0.9827587 0.9837709 0.9847318 0.9856434
## [221] 0.9865077 0.9873267 0.9881024 0.9888365 0.9895310
## [226] 0.9901874 0.9908077 0.9913933 0.9919460 0.9924672
## [231] 0.9929585 0.9934212 0.9938569 0.9942668 0.9946523
## [236] 0.9950145 0.9953548 0.9956741 0.9959737 0.9962546
## [241] 0.9965177
```

```
# ?par
x <- rt(250, df = 5)
qqnorm(x); qqline(x)
```




```
curve(dnorm, -5, 5)
y = density(x)
lines(y, lty=3)
```




```
# ?ppoints
ppoints(250)
```

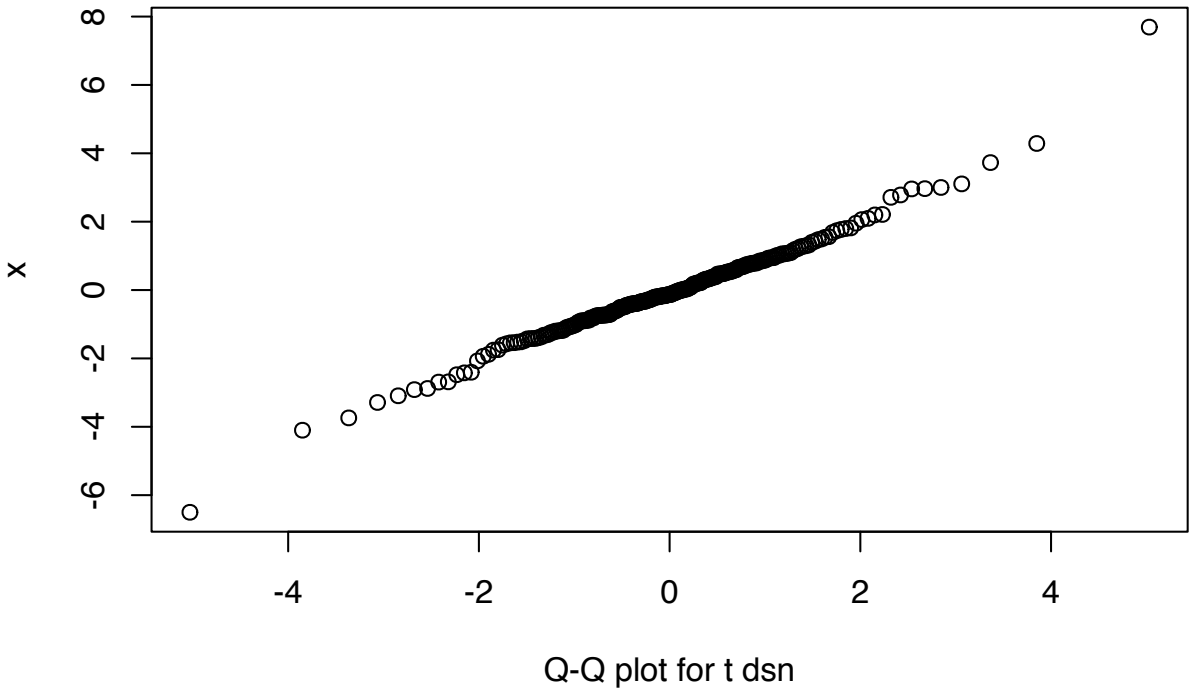
```
##   [1] 0.002 0.006 0.010 0.014 0.018 0.022 0.026 0.030
```

```
##   [9] 0.034 0.038 0.042 0.046 0.050 0.054 0.058 0.062
##  [17] 0.066 0.070 0.074 0.078 0.082 0.086 0.090 0.094
##  [25] 0.098 0.102 0.106 0.110 0.114 0.118 0.122 0.126
##  [33] 0.130 0.134 0.138 0.142 0.146 0.150 0.154 0.158
##  [41] 0.162 0.166 0.170 0.174 0.178 0.182 0.186 0.190
##  [49] 0.194 0.198 0.202 0.206 0.210 0.214 0.218 0.222
##  [57] 0.226 0.230 0.234 0.238 0.242 0.246 0.250 0.254
##  [65] 0.258 0.262 0.266 0.270 0.274 0.278 0.282 0.286
##  [73] 0.290 0.294 0.298 0.302 0.306 0.310 0.314 0.318
##  [81] 0.322 0.326 0.330 0.334 0.338 0.342 0.346 0.350
##  [89] 0.354 0.358 0.362 0.366 0.370 0.374 0.378 0.382
##  [97] 0.386 0.390 0.394 0.398 0.402 0.406 0.410 0.414
## [105] 0.418 0.422 0.426 0.430 0.434 0.438 0.442 0.446
## [113] 0.450 0.454 0.458 0.462 0.466 0.470 0.474 0.478
## [121] 0.482 0.486 0.490 0.494 0.498 0.502 0.506 0.510
## [129] 0.514 0.518 0.522 0.526 0.530 0.534 0.538 0.542
## [137] 0.546 0.550 0.554 0.558 0.562 0.566 0.570 0.574
## [145] 0.578 0.582 0.586 0.590 0.594 0.598 0.602 0.606
## [153] 0.610 0.614 0.618 0.622 0.626 0.630 0.634 0.638
## [161] 0.642 0.646 0.650 0.654 0.658 0.662 0.666 0.670
## [169] 0.674 0.678 0.682 0.686 0.690 0.694 0.698 0.702
## [177] 0.706 0.710 0.714 0.718 0.722 0.726 0.730 0.734
## [185] 0.738 0.742 0.746 0.750 0.754 0.758 0.762 0.766
## [193] 0.770 0.774 0.778 0.782 0.786 0.790 0.794 0.798
## [201] 0.802 0.806 0.810 0.814 0.818 0.822 0.826 0.830
## [209] 0.834 0.838 0.842 0.846 0.850 0.854 0.858 0.862
## [217] 0.866 0.870 0.874 0.878 0.882 0.886 0.890 0.894
## [225] 0.898 0.902 0.906 0.910 0.914 0.918 0.922 0.926
## [233] 0.930 0.934 0.938 0.942 0.946 0.950 0.954 0.958
## [241] 0.962 0.966 0.970 0.974 0.978 0.982 0.986 0.990
## [249] 0.994 0.998
```

```
ppoints(10)
```

```
##  [1] 0.06098 0.15854 0.25610 0.35366 0.45122 0.54878
##  [7] 0.64634 0.74390 0.84146 0.93902
```

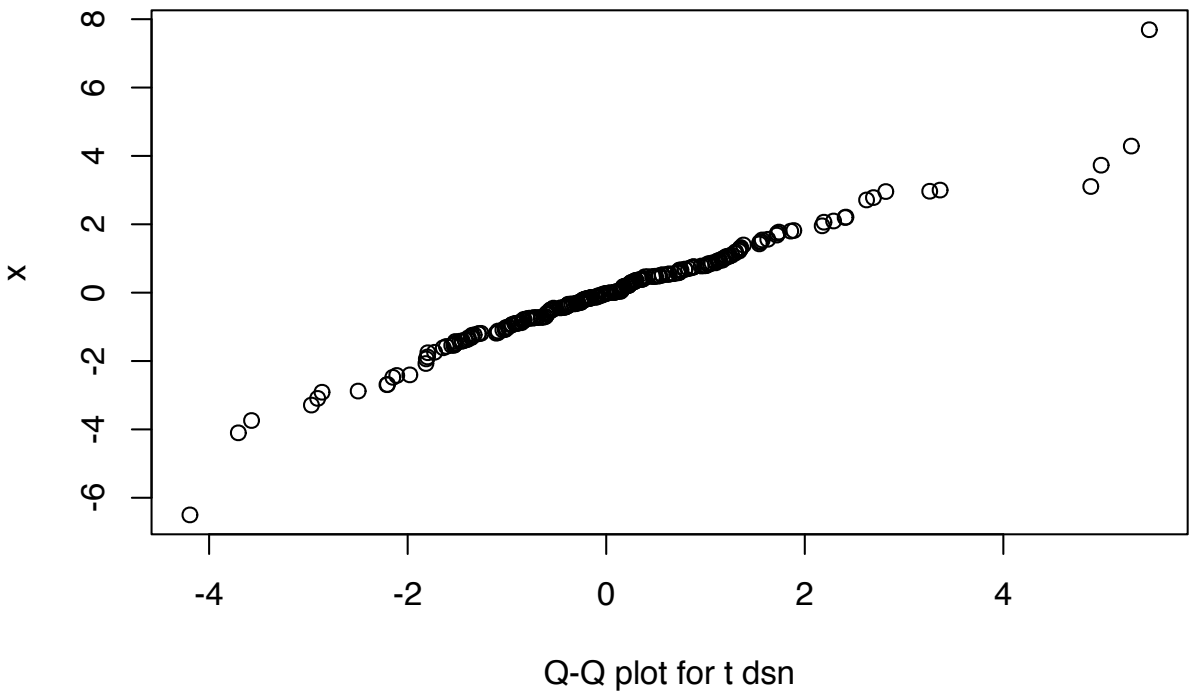
```
qqplot(qt(ppoints(250), df = 5), x, xlab = "Q-Q plot for t dsn")
```




```
windows()
```

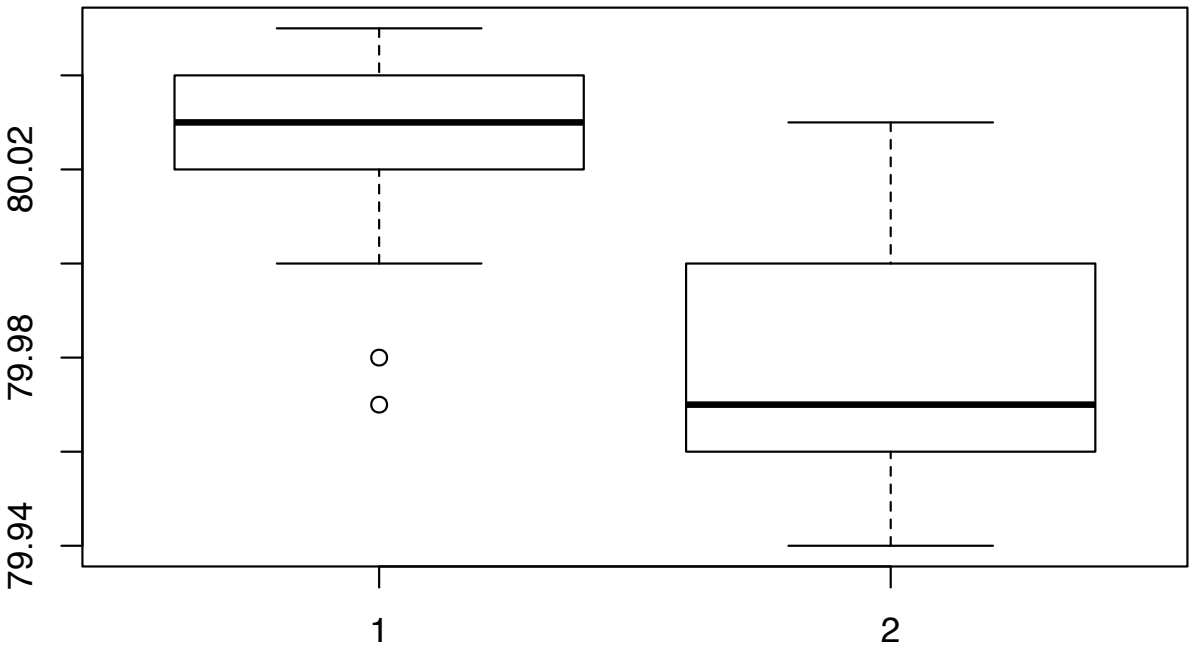
```
## Error in windows(): 함수 "windows"를 찾을 수 없습니다
```

```
qqplot(qt(runif(250), df = 5), x, xlab = "Q-Q plot for t dsn")
```




```
# ?shapiro.test
# ?ks.test
# ?t.test
```

```
A = c(79.98, 80.04, 80.02, 80.04, 80.03, 80.03, 80.04, 79.97, 80.05, 80.03, 80.02, 80.00, 80.02
B = c(80.02, 79.94, 79.98, 79.97, 79.97, 80.03, 79.95, 79.97)
boxplot(A, B)
```




```
t.test(A, B)
```

```
##
##  Welch Two Sample t-test
##
## data:  A and B
## t = 3.2, df = 12, p-value = 0.007
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.01386 0.07018
## sample estimates:
## mean of x mean of y
##     80.02     79.98
```

```
var.test(A, B)
```

```
##
##  F test to compare two variances
##
## data:  A and B
## F = 0.58, num df = 12, denom df = 7, p-value =
## 0.4
```

```
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.1251 2.1053
## sample estimates:
## ratio of variances
##             0.5837
```

```
t.test(A, B, var.equal=TRUE)
```

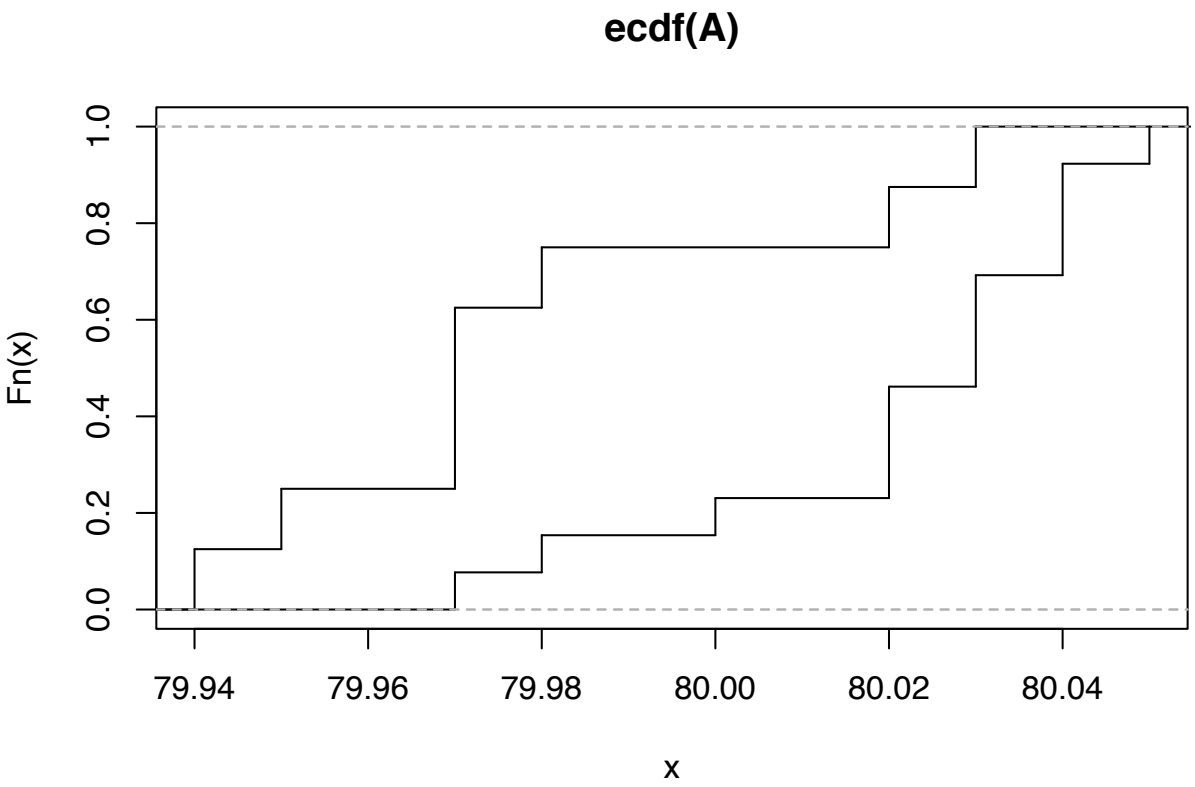
```
##
##  Two Sample t-test
##
## data:  A and B
## t = 3.5, df = 19, p-value = 0.003
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.01669 0.06735
## sample estimates:
## mean of x mean of y
##     80.02     79.98
```

```
wilcox.test(A, B)
```

```
## Warning in wilcox.test.default(A, B): cannot compute
## exact p-value with ties
```

```
##
##  Wilcoxon rank sum test with continuity
##  correction
##
## data:  A and B
## W = 89, p-value = 0.007
## alternative hypothesis: true location shift is not equal to 0
```

```
plot(ecdf(A), do.points=FALSE, verticals=TRUE, xlim=range(A, B))
plot(ecdf(B), do.points=FALSE, verticals=TRUE, add=TRUE)
```




```
ks.test(A, B)
```

```
## Warning in ks.test(A, B): cannot compute exact p-value
## with ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  A and B
## D = 0.6, p-value = 0.06
## alternative hypothesis: two-sided
```

```
# Chapter 9 Grouping, loops and conditional execution

# { } does grouping
# Usefulness of loops: for >> while >> repeat
for (i in 1:10) {
  print(2*i)
}
```

```
## [1] 2
## [1] 4
## [1] 6
## [1] 8
## [1] 10
## [1] 12
```

```
## [1] 14
## [1] 16
## [1] 18
## [1] 20
```

```
for (i in 1:10) print(2*i)
```

```
## [1] 2
## [1] 4
## [1] 6
## [1] 8
## [1] 10
## [1] 12
## [1] 14
## [1] 16
## [1] 18
## [1] 20
```

```
#while (    ) {
## Statements
#}

# # if ~ else ~
# if (   ) {
# # Statements 1
# } else {
# # Statements 2
# }
#
# if (    ) # Statement1
# else # Statement2
#
# if (   ) {
# # Statements 1
# } else if (   ) {
# # Statements 2
# } else if (   ) {
# # Statements 3
# } else {
# # Statements 4
# }


#
```

```
#

# Chapter 10 Writing your own functions

Square = function(x=0)
{
  return(x*x)
}

twosam = function(y1, y2)
{
  n1  = length(y1)
  n2  = length(y2)
  yb1 = mean(y1)
  yb2 = mean(y2)
  s1  = var(y1)
  s2  = var(y2)
  s   = ((n1 - 1)*s1 + (n2 - 1)*s2)/(n1 + n2 - 2)
  tst = (yb1 - yb2)/sqrt(s*(1/n1 + 1/n2))
  return (tst)
}

x = rnorm(10)
y = rt(10, 5)

twosam(x, y)
```

```
## [1] -1.432
```

```
T.test = function(y1, y2)
{
  n1  = length(y1)
  n2  = length(y2)
  yb1 = mean(y1)
  yb2 = mean(y2)
  s1  = var(y1)
  s2  = var(y2)
  s   = ((n1 - 1)*s1 + (n2 - 1)*s2)/(n1 + n2 - 2)

  tst = (yb1 - yb2)/sqrt(s*(1/n1 + 1/n2))
  DF = n1 + n2 - 2
  p.val = 2*(1 - pt(abs(tst), df=DF))

  Res = list(tst, DF, p.val, yb1, yb2)
```

```
  names(Res) = c("t", "df", "p-value", "mean of x", "mean of y")

  return (Res)
}

res = T.test(x, y)
t.test(x, y)
```

```
##
##  Welch Two Sample t-test
##
## data:  x and y
## t = -1.4, df = 18, p-value = 0.2
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.3379  0.2544
## sample estimates:
## mean of x mean of y
##   -0.6500   -0.1083
```

```
bslash = function(X, y)
{
  X = qr(X)
  return (qr.coef(X, y))
}

regcoeff = bslash(Xmat, yvar)
```

```
## Error in qr(X): 객체 'Xmat'를 찾을 수 없습니다
```

```
"%^%" = function(S, pow) with(eigen(S), vectors %*% (abs(values)^pow * t(vectors)))

M = matrix(c(2,1,1,2), nrow=2) ; M
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    1    2
```

```
M %^% 0.5
```

```
##       [,1]  [,2]
## [1,] 1.366 0.366
## [2,] 0.366 1.366
```

```
sqrtM = M%^%0.5 ; sqrtM
```

```
##       [,1]  [,2]
## [1,] 1.366 0.366
## [2,] 0.366 1.366
```

```
sqrtM %*% sqrtM
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    1    2
```

```
area = function(f, a, b, eps=1.0e-06, lim=10)
{
  fun1 = function(f, a, b, fa, fb, a0, eps, lim, fun)
  {
  ## function 'fun1' is only visible inside 'area'
    d = (a + b)/2
    h = (b - a)/4
    fd = f(d)
    a1 = h * (fa + fd)
    a2 = h * (fd + fb)
    if (abs(a0 - a1 - a2) < eps || lim == 0)
      return (a1 + a2)
    else {
      return (fun(f, a, d, fa, fd, a1, eps, lim - 1, fun) + fun(f, d, b, fd, fb, a2, eps, lim -
    }
  }
  fa = f(a)
  fb = f(b)
  a0 = ((fa + fb) * (b - a))/2
  fun1(f, a, b, fa, fb, a0, eps, lim, fun1)
}

area(dnorm, 0, 1)
```

```
## [1] 0.3413
```

```
integrate(dnorm, 0, 1)
```

```
## 0.3413 with absolute error < 3.8e-15
```

```
pnorm(1) - pnorm(0)
```

```
## [1] 0.3413
```

```
f = function(x)
{
  y = 2*x
  print(x)
  print(y)
  print(z)
}

f(1)
```

```
## [1] 1
## [1] 2
```

```
## Error in print(z): 객체 'z'를 찾을 수 없습니다
```

```
z = 3
f(1)
```

```
## [1] 1
## [1] 2
## [1] 3
```

```
cube = function(n) {
  sq = function() n*n
  n*sq()
}

cube(5)
```

```
## [1] 125
```

```
open.account = function(total)
{
  list(
    deposit = function(amount)
    {
      if(amount <= 0)
      stop("Deposits must be positive!\n")
      total <<- total + amount
      cat(amount, "deposited. Your balance is", total, "\n\n")
```

```
    },
    withdraw = function(amount)
    {
      if(amount > total)
      stop("You don't have that much money!\n")
      total <<- total - amount
      cat(amount, "withdrawn. Your balance is", total, "\n\n")
    },
    balance = function()
    {
      cat("Your balance is", total, "\n\n")
    }
  )
}

ross = open.account(100)
robert = open.account(200)

ross$balance()
```

```
## Your balance is 100
```

```
robert$balance()
```

```
## Your balance is 200
```

```
ross$deposit(50)
```

```
## 50 deposited. Your balance is 150
```

```
ross$balance()
```

```
## Your balance is 150
```

```
ross$withdraw(500)
```

```
## Error in ross$withdraw(500): You don't have that much money!
```

```
# More basic keywords and functions
1 %in% c(1,2,3,4)
```

```
## [1] TRUE
```

```
5 %in% c(1,2,3,4)
```

```
## [1] FALSE
```

```
is.finite(Inf)
```

```
## [1] FALSE
```

```
prod(1:3)
```

```
## [1] 6
```

```
cummax(1:10)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
cummax(10:1)
```

```
##  [1] 10 10 10 10 10 10 10 10 10 10
```

```
# ?xor
x = 11:20
x
```

```
##  [1] 11 12 13 14 15 16 17 18 19 20
```

```
which(x==3)
```

```
## integer(0)
```

```
which(x==13)
```

```
## [1] 3
```

```
length(x)
```

```
## [1] 10
```

```
y = "my string"
length(y)
```

```
## [1] 1
```

```
nchar(y)
```

```
## [1] 9
```

```
strsplit(y, " ")
```

```
## [[1]]
## [1] "my"     "string"
```

```
strsplit(y, " ")[[1]]
```

```
## [1] "my"     "string"
```

```
substr(y, 4, 5)
```

```
## [1] "st"
```

```
sample(1:10)
```

```
##  [1]  1  9  7  6  3  8 10  2  4  5
```

```
sample(1:10, 20)
```

```
## Error in sample.int(length(x), size, replace, prob): 'replace = FALSE' 일때는 모집단보다 큰 샘
```

```
sample(1:10, 20, replace=TRUE)
```

```
##  [1]  8  6  1  4  5  1  8  3 10 10  5  7  1  1  5  7  1
## [18]  2  8  7
```

```
sample(rep(1:10,2))
```

```
##  [1]  5  4  8  3  1  2  9 10 10  3  7  9  6  6  2  7  1
## [18]  5  8  4
```

```
## Error in na.locf(.): 함수 "na.locf"를 찾을 수 없습니다
```

```
## Error in split(Freqlist, Freqlist[, "Title"]): 객체 'Freqlist'를 찾을 수 없습니다
```

4.2 The basics

```
## Error in knitr::kable(Freqfinal[[1]], caption = names(Freqfinal)[1], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[2]], caption = names(Freqfinal)[2], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[3]], caption = names(Freqfinal)[3], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[4]], caption = names(Freqfinal)[4], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[5]], caption = names(Freqfinal)[5], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[6]], caption = names(Freqfinal)[6], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[7]], caption = names(Freqfinal)[7], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[8]], caption = names(Freqfinal)[8], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[9]], caption = names(Freqfinal)[9], booktabs = TRUE, : 객체
## Error in knitr::kable(Freqfinal[[10]], caption = names(Freqfinal)[10], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[11]], caption = names(Freqfinal)[11], : 객체 'Freqfinal'를
```

4.3 Common data structures

```
## Error in knitr::kable(Freqfinal[[12]], caption = names(Freqfinal)[12], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[13]], caption = names(Freqfinal)[13], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[14]], caption = names(Freqfinal)[14], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[15]], caption = names(Freqfinal)[15], : 객체 'Freqfinal'를
```

4.4 Statistics

```
## Error in knitr::kable(Freqfinal[[16]], caption = names(Freqfinal)[16], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[17]], caption = names(Freqfinal)[17], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[18]], caption = names(Freqfinal)[18], : 객체 'Freqfinal'를
## Error in knitr::kable(Freqfinal[[19]], caption = names(Freqfinal)[19], : 객체 'Freqfinal'를
```

```
## Error in knitr::kable(Freqfinal[[20]], caption = names(Freqfinal)[20], : 객체 'Freqfinal'를 찾
```

4.5 Working with R

```
## Error in knitr::kable(Freqfinal[[21]], caption = names(Freqfinal)[21], : 객체 'Freqfinal'를 찾
## Error in knitr::kable(Freqfinal[[22]], caption = names(Freqfinal)[22], : 객체 'Freqfinal'를 찾
## Error in knitr::kable(Freqfinal[[23]], caption = names(Freqfinal)[23], : 객체 'Freqfinal'를 찾
```

4.6 I/O

```
## Error in knitr::kable(Freqfinal[[24]], caption = names(Freqfinal)[24], : 객체 'Freqfinal'를 찾
## Error in knitr::kable(Freqfinal[[25]], caption = names(Freqfinal)[25], : 객체 'Freqfinal'를 찾
## Error in knitr::kable(Freqfinal[[26]], caption = names(Freqfinal)[26], : 객체 'Freqfinal'를 찾
```

5

stringr and lubridate

2017-05-24 조용순 전공의 강의

11주차 강의 자료입니다.

```
#"Stringr"
#R 표준 base 패키지에 포함된 함수군와 비슷한 기능을 하는 것으로 보이지만 더 합리적인 출력형식을 가지므
#패키지의 특징
#1) factor와 character를 같은 방식으로 처리
#2) 일관성 있는 함수 이름과 인수
#3) 다른 함수의 입력값으로 사용하기 편리한 출력값.
#  -입력값 NA가 포함되어 있을 때는 그 부분의 결과를 NA로 돌려줌
#4) 사용빈도가 떨어지는 문자열 조작 처리를 과감하게 제거하여 간략화시킴

#1. Installation
#install.packages("stringr")

library(stringr)

#2. Functions
#1) str_length(string): 문자열의 길이를 계산
#문자열의 길이를 계산해주는 함수
#base::nchar(x)와 같은 기능을 하는 함수
#단, NA 에 대해서는 2가 아닌 NA를 돌려줍니다.
str_length(c("i", "like", "programming", NA))
```

```
## [1]  1  4 11 NA
```

```
#> [1]  1  4 11 NA
nchar(c("i", "like", "programming", NA))
```

```
## [1]  1  4 11 NA
```

```
#> [1]  1  4 11  2

#2) str_sub(string, start=1, end=-1)
#문자열을 부분적으로 참조, 변경해주는 함수
#base::substr()와 같은 기능을 하는 함수
#음수를 사용하여 문자열의 끝으로 부터의 위치를 지정할 수 있습니다.
x <- "Michael Carreon"
str_sub(x,start=1,end=9)
```

```
## [1] "Michael C"
```

```
#> [1] "Michael C" * 띄어쓰기까지 포함하여 9번째 문자까지 반환해줍니다.
str_sub(x,1,9)
```

```
## [1] "Michael C"
```

```
#> [1] "Michael C" * start와 end는 쓰지 않아도 무방합니다.
str_sub(x,end=7)
```

```
## [1] "Michael"
```

```
#> [1] "Michael" * start 값을 지정해주지 않으면, default 값인 1로 지정됩니다. 즉, str_sub(x,1,7)과 같
str_sub(x,-7)
```

```
## [1] "Carreon"
```

```
#> [1] "Carreon" * 음수를 통하여 문자열 끝부터 7번째 오는 문자부터 반환해줍니다.
#Base R
substr(x,1,7)
```

```
## [1] "Michael"
```

```
#> [1] "Michael"

#3) str_c(..., sep='', collapse=NULL)
#문자열을 통합해주는 함수
#sep의 default가 스페이스 공백이 아니므로 base::paste0()와 비슷합니다.
str_c(letters[-26], " comes before ", letters[-1])
```

```
##  [1] "a comes before b" "b comes before c"
##  [3] "c comes before d" "d comes before e"
```

```
##  [5] "e comes before f" "f comes before g"
##  [7] "g comes before h" "h comes before i"
##  [9] "i comes before j" "j comes before k"
## [11] "k comes before l" "l comes before m"
## [13] "m comes before n" "n comes before o"
## [15] "o comes before p" "p comes before q"
## [17] "q comes before r" "r comes before s"
## [19] "s comes before t" "t comes before u"
## [21] "u comes before v" "v comes before w"
## [23] "w comes before x" "x comes before y"
## [25] "y comes before z"
```

```
#[1] "a comes before b" "b comes before c" "c comes before d" "d comes before e" "e comes befor
#[6] "f comes before g" "g comes before h" "h comes before i" "i comes before j" "j comes befor
#[11] "k comes before l" "l comes before m" "m comes before n" "n comes before o" "o comes befo
#[16] "p comes before q" "q comes before r" "r comes before s" "s comes before t" "t comes befo
#[21] "u comes before v" "v comes before w" "w comes before x" "x comes before y" "y comes befo
##Base R
paste0(letters[-26], " comes before ", letters[-1])
```

```
##  [1] "a comes before b" "b comes before c"
##  [3] "c comes before d" "d comes before e"
##  [5] "e comes before f" "f comes before g"
##  [7] "g comes before h" "h comes before i"
##  [9] "i comes before j" "j comes before k"
## [11] "k comes before l" "l comes before m"
## [13] "m comes before n" "n comes before o"
## [15] "o comes before p" "p comes before q"
## [17] "q comes before r" "r comes before s"
## [19] "s comes before t" "t comes before u"
## [21] "u comes before v" "v comes before w"
## [23] "w comes before x" "x comes before y"
## [25] "y comes before z"
```

```
str_c(letters, collapse = ", ")
```

```
## [1] "a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z"
```

```
#> [1] "a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z"
#sep와 collapse의 차이는 한 벡터 안에 존재하느냐 아니냐입니다.
str_c("안","녕","하","세","요",sep="_")
```

```
## [1] "안_녕_하_세_요"
```

```
#> [1] "안_녕_하_세_요"
str_c(c("안","녕","하","세","요"),collapse="_")
```

```
## [1] "안_녕_하_세_요"
```

```
#> [1] "안_녕_하_세_요"

#4) str_split(string, pattern, n=Inf)
#문자열을 분리해주는 함수--> 결과값은 list입니다.
#base::strsplit(x, split)와 대응하는 함수입니다.
#str_split_fixed()도 있고, 결과값은 matrix
fruits <- c("apples and oranges and pears and bananas", "pineapples and mangos and guavas")
str_split(fruits, " and ")
```

```
## [[1]]
## [1] "apples"  "oranges" "pears"   "bananas"
##
## [[2]]
## [1] "pineapples" "mangos"     "guavas"
```

```
#> [[1]]
#> [1] "apples"  "oranges" "pears"   "bananas"
#>
#> [[2]]
#> [1] "pineapples" "mangos"     "guavas"
#Base R
strsplit(fruits, "and")
```

```
## [[1]]
## [1] "apples "   " oranges " " pears "   " bananas"
##
## [[2]]
## [1] "pineapples " " mangos "    " guavas"
```

```
#> [[1]]
#> [1] "apples "   " oranges " " pears "   " bananas"
#>
#> [[2]]
#> [1] "pineapples " " mangos "    " guavas"
str_split(fruits, " and ", n = 3)
```

```
## [[1]]
## [1] "apples"            "oranges"
```

```
## [3] "pears and bananas"
##
## [[2]]
## [1] "pineapples" "mangos"     "guavas"
```

```
#> [[1]]
#> [1] "apples"            "oranges"           "pears and bananas"
#>
#> [[2]]
#> [1] "pineapples" "mangos"     "guavas"
str_split(fruits, " and ", n = 2)
```

```
## [[1]]
## [1] "apples"
## [2] "oranges and pears and bananas"
##
## [[2]]
## [1] "pineapples"        "mangos and guavas"
```

```
#> [[1]]
#> [1] "apples"                        "oranges and pears and bananas"
#>
#> [[2]]
#> [1] "pineapples"        "mangos and guavas"
str_split_fixed(fruits, " and ", 4)
```

```
##      [,1]         [,2]      [,3]     [,4]
## [1,] "apples"     "oranges" "pears"  "bananas"
## [2,] "pineapples" "mangos"  "guavas" ""
```

```
#>      [,1]         [,2]      [,3]     [,4]
#> [1,] "apples"     "oranges" "pears"  "bananas"
#> [2,] "pineapples" "mangos"  "guavas" "

#5)str_detect(string, pattern)
#매치하는 곳이 있는지 없는지를 logical 값(True or False)으로 반환해주는 함수
#base::grepl(pattern, x)과 대응
fruit <- c("apple", "banana", "pear", "pinapple")
str_detect(fruit, "a")
```

```
## [1] TRUE TRUE TRUE TRUE
```

```
#> [1] TRUE TRUE TRUE TRUE
str_detect(fruit, "^a")
```

```
## [1]  TRUE FALSE FALSE FALSE
```

```
#> [1]  TRUE FALSE FALSE FALSE
str_detect(fruit, "a$")
```

```
## [1] FALSE  TRUE FALSE FALSE
```

```
#> [1] FALSE  TRUE FALSE FALSE
str_detect(fruit, "b")
```

```
## [1] FALSE  TRUE FALSE FALSE
```

```
#> [1] FALSE  TRUE FALSE FALSE
str_detect(fruit, "[aeiou]")
```

```
## [1] TRUE TRUE TRUE TRUE
```

```
#> [1] TRUE TRUE TRUE TRUE

#6) str_count(string, pattern)
#매치하는 곳의 수를 반환해주는 함수
#그 글자가 몇 개 포함되어 있는지 알려줍니다.
str_count(fruit, "p")
```

```
## [1] 2 0 1 3
```

```
#> [1] 2 0 1 3
str_count(fruit, c("a", "b", "p", "p"))
```

```
## [1] 1 1 1 3
```

```
#> [1] 1 1 1 3

#7)str_locate(string, pattern)
#처음으로 매치되는 곳의 start, end 위치를 행렬로 반환해주는 함수
str_locate(fruit, "e")
```

```
##      start end
## [1,]     5   5
## [2,]    NA  NA
## [3,]     2   2
## [4,]     8   8
```

```
#>      start end
#> [1,]     5   5
#> [2,]    NA  NA
#> [3,]     2   2
#> [4,]     8   8
str_locate(fruit, "pl")
```

```
##      start end
## [1,]     3   4
## [2,]    NA  NA
## [3,]    NA  NA
## [4,]     6   7
```

```
#>      start end
#> [1,]     3   4
#> [2,]    NA  NA
#> [3,]    NA  NA
#> [4,]     6   7

#8)str_extract(string, pattern)
#매치된 부분 문자열을 추출하는 함수
#매치되지 않은 요소는 NA로 출력합니다
#base::grep(pattern, x, value=TRUE)와 비슷하나 이 함수는 매치된 요소만 원래의 형태로 돌려줍니다
shopping_list <- c("apples x4", "flour", "sugar", "milk x2")
str_extract(shopping_list, "\\d")
```

```
## [1] "4" NA  NA  "2"
```

```
#> [1] "4" NA  NA  "2"
grep("\\d", shopping_list, value = TRUE)
```

```
## [1] "apples x4" "milk x2"
```

```
#> [1] "apples x4" "milk x2"
str_extract(shopping_list, "[a-z]+")
```

```
## [1] "apples" "flour"  "sugar"  "milk"
```

```
#> [1] "apples" "flour"  "sugar"  "milk"
grep("[a-z]+", shopping_list, value = TRUE)
```

```
## [1] "apples x4" "flour"     "sugar"     "milk x2"
```

```
#> [1] "apples x4" "flour"     "sugar"     "milk x2"

#9)str_match(string, pattern)
#매치된 부분 문자열을 추출하고 참조를 행렬로 돌려주는 함수
#str_extract 함수의 결과를 1열에 , 각 괄호에 매치된 이후의 결과가 2열 이후에 들어갑니다.
strings <- c(" 219 733 8965", "329-293-8753 ", "banana", "595 794 7569", "387 287 6718", "apple
phone <- "([2-9][0-9]{2})[- .]([0-9]{3})[- .]([0-9]{4})"
str_extract(strings, phone)
```

```
##  [1] "219 733 8965" "329-293-8753" NA
##  [4] "595 794 7569" "387 287 6718" NA
##  [7] "233.398.9187" "482 952 3315" "239 923 8115"
## [10] "842 566 4692" "579-499-7527" NA
## [13] "543.355.3679"
```

```
#>  [1] "219 733 8965" "329-293-8753" NA             "595 794 7569"
#>  [5] "387 287 6718" NA             "233.398.9187" "482 952 3315"
#>  [9] "239 923 8115" "842 566 4692" "579-499-7527" NA
#> [13] "543.355.3679"
str_match(strings, phone)
```

```
##       [,1]           [,2]  [,3]  [,4]
##  [1,] "219 733 8965" "219" "733" "8965"
##  [2,] "329-293-8753" "329" "293" "8753"
##  [3,] NA             NA    NA    NA
##  [4,] "595 794 7569" "595" "794" "7569"
##  [5,] "387 287 6718" "387" "287" "6718"
##  [6,] NA             NA    NA    NA
##  [7,] "233.398.9187" "233" "398" "9187"
##  [8,] "482 952 3315" "482" "952" "3315"
##  [9,] "239 923 8115" "239" "923" "8115"
## [10,] "842 566 4692" "842" "566" "4692"
## [11,] "579-499-7527" "579" "499" "7527"
## [12,] NA             NA    NA    NA
## [13,] "543.355.3679" "543" "355" "3679"
```

```
#>       [,1]           [,2]  [,3]  [,4]
#>  [1,] "219 733 8965" "219" "733" "8965"
#>  [2,] "329-293-8753" "329" "293" "8753"
#>  [3,] NA             NA    NA    NA
#>  [4,] "595 794 7569" "595" "794" "7569"
#>  [5,] "387 287 6718" "387" "287" "6718"
#>  [6,] NA             NA    NA    NA
```

```
#>  [7,] "233.398.9187" "233" "398" "9187"
#>  [8,] "482 952 3315" "482" "952" "3315"
#>  [9,] "239 923 8115" "239" "923" "8115"
#> [10,] "842 566 4692" "842" "566" "4692"
#> [11,] "579-499-7527" "579" "499" "7527"
#> [12,] NA             NA    NA    NA
#> [13,] "543.355.3679" "543" "355" "3679"

#10)str_replace(string, pattern, replacement)
#매치되지 않은 부분은 그대로 두고 매치된 부분만 치환하는 함수
#base::sub(매치할 부분,치환할 문자,문자열)와 같은 기능을 합니다.
fruits <- c("one apple", "two pears", "three bananas")
str_replace(fruits, "[aeiou]", "-")
```

```
## [1] "-ne apple"     "tw- pears"     "thr-e bananas"
```

```
#> [1] "-ne apple"     "tw- pears"     "thr-e bananas"
str_replace_all(fruits, "[aeiou]", "-")
```

```
## [1] "-n- -ppl-"     "tw- p--rs"     "thr-- b-n-n-s"
```

```
#> [1] "-n- -ppl-"     "tw- p--rs"     "thr-- b-n-n-s"

#11)str_trim(string, side="both")
#공백문자를 제거하는 함수
str_trim("        fruits        ", side="both")
```

```
## [1] "fruits"
```

```
#>[1] "fruits"
Trim = function(x) gsub("^\\s+|\\s+$", "", x)
Trim("        fruits        ")
```

```
## [1] "fruits"
```

```
#>[1] "fruits"

#"lubridate"
#lubri:lubricate(기름을 치다, 기름을 바르다, 원활히 하다)+date
#Lubridate is an R package that makes it easier to work with dates and times
#1.Installation
install.packages("lubridate")
```

```
## Error in contrib.url(repos, "source"): trying to use CRAN without setting a mirror
```

```
library(lubridate)
```

```
## Loading required package: methods
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
#2.Functions
#1)Parsing dates and times(dates & times 객체 만들기)
##Date
#Base R
as.Date("2011-06-04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
as.Date("2011-6-4")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
as.Date("2011/06/04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
##
as.Date("20110604") # error
```

```
## Error in charToDate(x): character string is not in a standard unambiguous format
```

```
as.Date("06-04-2011") ### [1] "0006-04-20" (미국식 표현) #Problem
```

```
## [1] "0006-04-20"
```

```
#lubridate package
ymd("2011/06/04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
```

```
#심볼의 순서를 바꾸어도
mdy("06/04/2011")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
dmy("04/06/2011")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"

#lubridate에서의 날짜 양식의 관용
#heterogeneous format(불균일한 양식)에 대한 다양한 준비들이 되어있음
ymd("2011/06/04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("2011-06-04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("20110604")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("110604")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("11.06.04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("11,06,04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("11_06.04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("2011  06  04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("2011!?06??!04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"
ymd("2011 =06??04")
```

```
## [1] "2011-06-04"
```

```
## [1] "2011-06-04"

##Dates + Times 객체 만들기
#Base R
as.POSIXct("2011-06-04 13:30:50")
```

```
## [1] "2011-06-04 13:30:50 KST"
```

```
## [1] "2011-06-04 13:30:50 KST"
as.POSIXct("2011-06-04 13") # No
```

```
## [1] "2011-06-04 KST"
```

```
## [1] "2011-06-04 KST"
strptime("2011-06-04 13:30:50", "%Y-%m-%d %H:%M:%S")
```

```
## [1] "2011-06-04 13:30:50 KST"
```

```
## [1] "2011-06-04 13:30:50 KST"

#lubridate package
ymd_hms("2011-06-04 13:30:50")
```

```
## [1] "2011-06-04 13:30:50 UTC"
```

```
## [1] "2011-06-04 13:30:50 UTC"
#조금 더 융통성이 있게 사용할 수 있는 점
ymd_h("2011-06-04 13")
```

```
## [1] "2011-06-04 13:00:00 UTC"
```

```
## [1] "2011-06-04 13:00:00 UTC"

#2)Setting and Extracting information
#부분정보를 추출하기 위한 간편 함수들
#함수명칭도 상식적으로 이해하기 쉬운 것들
#:second(), minute(), hour(), day(), wday(), yday(), week(), month(), year()
ld1 <- ymd_hms("2011-06-04 13:30:50")

year(ld1)
```

```
## [1] 2011
```

```
## [1] 2011
month(ld1)
```

```
## [1] 6
```

```
## [1] 6
day(ld1)
```

```
## [1] 4
```

```
## [1] 4
wday(ld1)
```

```
## [1] 7
```

```
## [1] 7
yday(ld1)
```

```
## [1] 155
```

```
## [1] 155
hour(ld1)
```

```
## [1] 13
```

```
## [1] 13
minute(ld1)
```

```
## [1] 30
```

```
## [1] 30
second(ld1)
```

```
## [1] 50
```

```
## [1] 50

# month, wday 의 경우 label 인자를 가지고 있는데 이를 TRUE 로 설정할 경우
month(ld1, label = T)
```

```
## [1] Jun
## 12 Levels: Jan < Feb < Mar < Apr < May < ... < Dec
```

```
## [1] Jun
## 12 Levels: Jan < Feb < Mar < Apr < May < Jun < Jul < Aug < Sep < ... < Dec
wday(ld1, label = T)
```

```
## [1] Sat
## 7 Levels: Sun < Mon < Tues < Wed < Thurs < ... < Sat
```

```
## [1] Sat
## Levels: Sun < Mon < Tues < Wed < Thurs < Fri < Sat

#3)Update date-time
#"2011년 6월 4일 13:30:50" 로 저장되어있던 ld1 에 대해 시각(hour)을 10시로 바꾸려면
hour(ld1) <- 10
ld1
```

```
## [1] "2011-06-04 10:30:50 UTC"
```

```
## [1] "2011-06-04 10:30:50 UTC"

#update() 함수를 이용해 10시로 변경된 ld1 을 다시 13로
ld1 <- update(ld1, hour = 13)
ld1
```

```
## [1] "2011-06-04 13:30:50 UTC"
```

```
## [1] "2011-06-04 13:30:50 UTC"

#4) Arithmetic with date times
#lubridate 와 같은 패키지를 공부하는 목적 중 가장 중요한 특징
#산술연산에서 사용할 수 있는 패밀리:간편함수마지막에 "s" 가 붙음으로써 쓰임이 달라진 것
#days(), seconds(), minutes(), hours(), weeks(), years(), milliseconds(), microseconds(), nanos
ymd("2016-01-30") + days(2)
```

```
## [1] "2016-02-01"
```

```
## [1] "2016-02-01"
ymd("2016-01-30") - days(1:30)
```

```
##  [1] "2016-01-29" "2016-01-28" "2016-01-27"
##  [4] "2016-01-26" "2016-01-25" "2016-01-24"
##  [7] "2016-01-23" "2016-01-22" "2016-01-21"
## [10] "2016-01-20" "2016-01-19" "2016-01-18"
## [13] "2016-01-17" "2016-01-16" "2016-01-15"
## [16] "2016-01-14" "2016-01-13" "2016-01-12"
## [19] "2016-01-11" "2016-01-10" "2016-01-09"
## [22] "2016-01-08" "2016-01-07" "2016-01-06"
## [25] "2016-01-05" "2016-01-04" "2016-01-03"
## [28] "2016-01-02" "2016-01-01" "2015-12-31"
```

```
##  [1] "2016-01-29" "2016-01-28" "2016-01-27" "2016-01-26" "2016-01-25"
##  [6] "2016-01-24" "2016-01-23" "2016-01-22" "2016-01-21" "2016-01-20"
## [11] "2016-01-19" "2016-01-18" "2016-01-17" "2016-01-16" "2016-01-15"
## [16] "2016-01-14" "2016-01-13" "2016-01-12" "2016-01-11" "2016-01-10"
## [21] "2016-01-09" "2016-01-08" "2016-01-07" "2016-01-06" "2016-01-05"
## [26] "2016-01-04" "2016-01-03" "2016-01-02" "2016-01-01" "2015-12-31"
ymd("2013-01-31") + months(0:11)
```

```
##  [1] "2013-01-31" NA           "2013-03-31"
##  [4] NA           "2013-05-31" NA
##  [7] "2013-07-31" "2013-08-31" NA
## [10] "2013-10-31" NA           "2013-12-31"
```

```
##  [1] "2013-01-31" NA           "2013-03-31" NA           "2013-05-31"
##  [6] NA           "2013-07-31" "2013-08-31" NA           "2013-10-31"
## [11] NA           "2013-12-31"

#5) Application with lubridate and dplyr
#lubridate package 에 내장된 데이터셋 lakers 를 이용
```

```
#data(lakers)
sessionInfo()
```

```
## R version 3.4.0 (2017-04-21)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Sierra 10.12.5
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] ko_KR.UTF-8/ko_KR.UTF-8/ko_KR.UTF-8/C/ko_KR.UTF-8/ko_KR.UTF-8
##
## attached base packages:
## [1] methods   stats     graphics  grDevices utils
## [6] datasets  base
##
## other attached packages:
## [1] lubridate_1.6.0 stringr_1.2.0   lattice_0.20-35
## [4] dplyr_0.7.1     readxl_1.0.0    knitr_1.16
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.11     rstudioapi_0.6   bindr_0.1
##  [4] magrittr_1.5     R6_2.2.2         rlang_0.1.1
##  [7] tools_3.4.0      grid_3.4.0       htmltools_0.3.6
## [10] yaml_2.1.14      rprojroot_1.2    digest_0.6.12
## [13] assertthat_0.2.0 tibble_1.3.3     bookdown_0.4.1
## [16] bindrcpp_0.2     glue_1.1.1       evaluate_0.10.1
## [19] rmarkdown_1.6    stringi_1.1.5    compiler_3.4.0
## [22] cellranger_1.1.0 backports_1.1.0  pkgconfig_2.0.1
```

```
lakers <- lakers %>% tbl_df
lakers #--> date, time 변수가 서로 나뉘어 있다.
```

```
## # A tibble: 34,624 x 13
##        date opponent game_type  time period      etype
##       <int>    <chr>     <chr> <chr>  <int>      <chr>
##  1 20081028      POR      home 12:00      1  jump ball
##  2 20081028      POR      home 11:39      1       shot
##  3 20081028      POR      home 11:37      1    rebound
##  4 20081028      POR      home 11:25      1       shot
##  5 20081028      POR      home 11:23      1    rebound
##  6 20081028      POR      home 11:22      1       shot
```

```
##  7 20081028      POR      home 11:22      1       foul
##  8 20081028      POR      home 11:22      1 free throw
##  9 20081028      POR      home 11:00      1       foul
## 10 20081028      POR      home 10:53      1       shot
## # ... with 34,614 more rows, and 7 more variables:
## #   team <chr>, player <chr>, result <chr>,
## #   points <int>, type <chr>, x <int>, y <int>
```

```
## # A tibble: 34,624 <U+00D7> 13
##        date opponent game_type  time period      etype  team
##       <int>    <chr>     <chr> <chr>  <int>      <chr> <chr>
## 1  20081028      POR      home 12:00      1  jump ball   OFF
## 2  20081028      POR      home 11:39      1       shot   LAL
## 3  20081028      POR      home 11:37      1    rebound   LAL
## 4  20081028      POR      home 11:25      1       shot   LAL
## 5  20081028      POR      home 11:23      1    rebound   LAL
## 6  20081028      POR      home 11:22      1       shot   LAL
## 7  20081028      POR      home 11:22      1       foul   POR
## 8  20081028      POR      home 11:22      1 free throw   LAL
## 9  20081028      POR      home 11:00      1       foul   LAL
## 10 20081028      POR      home 10:53      1       shot   POR
## # ... with 34,614 more rows, and 6 more variables: player <chr>,
## #   result <chr>, points <int>, type <chr>, x <int>, y <int>

lakers <- lakers %>%
    mutate(date = paste(date, time) %>% ymd_hm) %>%
    dplyr::rename(time_index = date) %>%
    select(-time)

#date, time 두변수를 붙인 문자열에 대해 ymd_hm() 함수로 넘긴 후
#time_index 라는 변수에 담고,
#date, time 두 변수를 제외한 것이다.

lakers
```

```
## # A tibble: 34,624 x 12
##             time_index opponent game_type period
##                  <dttm>    <chr>     <chr>  <int>
##  1 2008-10-28 12:00:00      POR      home      1
##  2 2008-10-28 11:39:00      POR      home      1
##  3 2008-10-28 11:37:00      POR      home      1
##  4 2008-10-28 11:25:00      POR      home      1
##  5 2008-10-28 11:23:00      POR      home      1
##  6 2008-10-28 11:22:00      POR      home      1
##  7 2008-10-28 11:22:00      POR      home      1
```

```
##  8 2008-10-28 11:22:00      POR      home      1
##  9 2008-10-28 11:00:00      POR      home      1
## 10 2008-10-28 10:53:00      POR      home      1
## # ... with 34,614 more rows, and 8 more variables:
## #   etype <chr>, team <chr>, player <chr>,
## #   result <chr>, points <int>, type <chr>, x <int>,
## #   y <int>
```

```
## # A tibble: 34,624 <U+00D7> 12
##              time_index opponent game_type period      etype  team
##                   <dttm>    <chr>     <chr>  <int>      <chr> <chr>
## 1  2008-10-28 12:00:00      POR      home      1  jump ball   OFF
## 2  2008-10-28 11:39:00      POR      home      1       shot   LAL
## 3  2008-10-28 11:37:00      POR      home      1    rebound   LAL
## 4  2008-10-28 11:25:00      POR      home      1       shot   LAL
## 5  2008-10-28 11:23:00      POR      home      1    rebound   LAL
## 6  2008-10-28 11:22:00      POR      home      1       shot   LAL
## 7  2008-10-28 11:22:00      POR      home      1       foul   POR
## 8  2008-10-28 11:22:00      POR      home      1 free throw   LAL
## 9  2008-10-28 11:00:00      POR      home      1       foul   LAL
## 10 2008-10-28 10:53:00      POR      home      1       shot   POR
## # ... with 34,614 more rows, and 6 more variables: player <chr>,
## #   result <chr>, points <int>, type <chr>, x <int>, y <int>

#Using "group by" 월별 평균을 x, y 변수에 대해서 계산:month() 함수를 이용cf) 연별 평균을 계산하고 싶
lakers %>%
    group_by(month(time_index)) %>%
    summarize(mean_x = mean(x, na.rm = T), mean_y = mean(y, na.rm = T))
```

```
## # A tibble: 7 x 3
##   `month(time_index)` mean_x mean_y
##                 <dbl>  <dbl>  <dbl>
## 1                   1  25.49  13.89
## 2                   2  25.02  13.17
## 3                   3  25.52  13.21
## 4                   4  25.38  13.46
## 5                  10  24.92  13.12
## 6                  11  25.47  13.37
## 7                  12  25.06  13.48
```

```
## # A tibble: 7 <U+00D7> 3
##   `month(time_index)`    mean_x    mean_y
##                 <dbl>     <dbl>     <dbl>
## 1                   1 25.49382 13.89279
```

```
## 2                   2 25.01759 13.17499
## 3                   3 25.51587 13.20571
## 4                   4 25.38344 13.46396
## 5                  10 24.92188 13.12500
## 6                  11 25.47463 13.36926
## 7                  12 25.05895 13.48262
```

```
#Using "filter" "2008-10-28 12:00:00" 이전의 기간을 서브세팅
lakers %>%
    filter(time_index <= ymd_hms("2008-10-28 12:00:00"))
```

```
## # A tibble: 416 x 12
##             time_index opponent game_type period
##                  <dttm>    <chr>     <chr>  <int>
##  1 2008-10-28 12:00:00      POR      home      1
##  2 2008-10-28 11:39:00      POR      home      1
##  3 2008-10-28 11:37:00      POR      home      1
##  4 2008-10-28 11:25:00      POR      home      1
##  5 2008-10-28 11:23:00      POR      home      1
##  6 2008-10-28 11:22:00      POR      home      1
##  7 2008-10-28 11:22:00      POR      home      1
##  8 2008-10-28 11:22:00      POR      home      1
##  9 2008-10-28 11:00:00      POR      home      1
## 10 2008-10-28 10:53:00      POR      home      1
## # ... with 406 more rows, and 8 more variables:
## #   etype <chr>, team <chr>, player <chr>,
## #   result <chr>, points <int>, type <chr>, x <int>,
## #   y <int>
```

```
## # A tibble: 416 <U+00D7> 12
##             time_index opponent game_type period      etype  team
##                  <dttm>    <chr>     <chr>  <int>      <chr> <chr>
## 1  2008-10-28 12:00:00      POR      home      1  jump ball   OFF
## 2  2008-10-28 11:39:00      POR      home      1       shot   LAL
## 3  2008-10-28 11:37:00      POR      home      1    rebound   LAL
## 4  2008-10-28 11:25:00      POR      home      1       shot   LAL
## 5  2008-10-28 11:23:00      POR      home      1    rebound   LAL
## 6  2008-10-28 11:22:00      POR      home      1       shot   LAL
## 7  2008-10-28 11:22:00      POR      home      1       foul   POR
## 8  2008-10-28 11:22:00      POR      home      1 free throw   LAL
## 9  2008-10-28 11:00:00      POR      home      1       foul   LAL
## 10 2008-10-28 10:53:00      POR      home      1       shot   POR
## # ... with 406 more rows, and 6 more variables: player <chr>,
```

```
## #   result <chr>, points <int>, type <chr>, x <int>, y <int>
```

```
# "2008-10-28 12:00:00" ~ "2009-03-09 00:33:00" 의 기간에 대해서 서브세팅
lakers %>%
    filter(time_index >= ymd_hms("2008-10-28 12:00:00"), time_index <= ymd_hms("2009-03-09 00:3
```

```
## # A tibble: 25,554 x 12
##             time_index opponent game_type period
##                  <dttm>    <chr>     <chr>  <int>
##  1 2008-10-28 12:00:00      POR      home      1
##  2 2008-10-29 12:00:00      LAC      away      1
##  3 2008-10-29 11:36:00      LAC      away      1
##  4 2008-10-29 11:24:00      LAC      away      1
##  5 2008-10-29 11:24:00      LAC      away      1
##  6 2008-10-29 11:08:00      LAC      away      1
##  7 2008-10-29 10:58:00      LAC      away      1
##  8 2008-10-29 10:57:00      LAC      away      1
##  9 2008-10-29 10:41:00      LAC      away      1
## 10 2008-10-29 10:40:00      LAC      away      1
## # ... with 25,544 more rows, and 8 more variables:
## #   etype <chr>, team <chr>, player <chr>,
## #   result <chr>, points <int>, type <chr>, x <int>,
## #   y <int>
```

```
#interval() 함수와 %within% 연산자를 이용하면 조금 더 직관적인 서브세팅(interval() 함수대신 %--% 연
inter <- interval(ymd_hms("2008-10-28 12:00:00"), ymd_hms("2009-03-09 00:33:00"))
lakers %>%
    filter(time_index %within% inter)
```

```
## # A tibble: 25,554 x 12
##             time_index opponent game_type period
##                  <dttm>    <chr>     <chr>  <int>
##  1 2008-10-28 12:00:00      POR      home      1
##  2 2008-10-29 12:00:00      LAC      away      1
##  3 2008-10-29 11:36:00      LAC      away      1
##  4 2008-10-29 11:24:00      LAC      away      1
##  5 2008-10-29 11:24:00      LAC      away      1
##  6 2008-10-29 11:08:00      LAC      away      1
##  7 2008-10-29 10:58:00      LAC      away      1
##  8 2008-10-29 10:57:00      LAC      away      1
##  9 2008-10-29 10:41:00      LAC      away      1
## 10 2008-10-29 10:40:00      LAC      away      1
## # ... with 25,544 more rows, and 8 more variables:
## #   etype <chr>, team <chr>, player <chr>,
```

```
## #   result <chr>, points <int>, type <chr>, x <int>,
## #   y <int>
```

```
## # A tibble: 25,554 <U+00D7> 12
##             time_index opponent game_type period     etype  team
##                  <dttm>    <chr>     <chr>  <int>     <chr> <chr>
## 1  2008-10-28 12:00:00      POR      home      1 jump ball   OFF
## 2  2008-10-29 12:00:00      LAC      away      1 jump ball   OFF
## 3  2008-10-29 11:36:00      LAC      away      1      shot   LAL
## 4  2008-10-29 11:24:00      LAC      away      1      shot   LAC
## 5  2008-10-29 11:24:00      LAC      away      1   rebound   LAL
## 6  2008-10-29 11:08:00      LAC      away      1      shot   LAL
## 7  2008-10-29 10:58:00      LAC      away      1      shot   LAC
## 8  2008-10-29 10:57:00      LAC      away      1   rebound   LAL
## 9  2008-10-29 10:41:00      LAC      away      1      shot   LAL
## 10 2008-10-29 10:40:00      LAC      away      1   rebound   LAC
## # ... with 25,544 more rows, and 6 more variables: player <chr>,
## #   result <chr>, points <int>, type <chr>, x <int>, y <int>
```

```
# r if (knitr:::is_html_output()) '# Assignments'
```

A

Assignments

A.1 Assignment 1

첨부한 concUnitConv-test.R과 유사한 R script를 실행하였을 때, concUnitConv-test.Rout과 유사한 결과나 나오는 concUnitConv.R 파일을 작성하시오.

- 제출기한 : 2017-05-10 18:00
- 제출방법 : R scirpt와 output을 k@acr.kr[1], shan@acp.kr[2], sec@acp.kr[3] 로 제출

A.1.1 concUnitConv-test.R

```
source("D:/G/Desk/R/concUnitConv.R")

concUnitConv() # Wrong input
concUnitConv("kg/L", "g/L") # Wrong input
concUnitConv("g/kL", "g/L") # Wrong input

concUnitConv("mg/L", "ug/mL")
Theoph$conc * concUnitConv("mg/L", "ug/L")
Theoph$conc * concUnitConv("mg/L", "mg/mL")
Theoph$conc * concUnitConv("mg/L", "mmol/L") # Wrong input
Theoph$conc * concUnitConv("mg/L", "mmol/L", MW=-100) # Wrong input
Theoph$conc * concUnitConv("mg/L", "mM", MW=180.164) # Wrong input
Theoph$conc * concUnitConv("mg/L", "mmol/L", MW=180.164)
Theoph$mM = Theoph$conc * concUnitConv("mg/L", "mmol/L", MW=180.164)
Theoph$mM * concUnitConv("mmol/L", "ug/L", MW=180.164)
Theoph$mM * concUnitConv("mmol/L", "ug/mL", MW=180.164)
```

[1] mailto:k@acr.kr
[2] mailto:shan@acp.kr
[3] mailto:sec@acp.kr

A.1.2 concUnitConv-test.Rout

```
> source("D:/G/Desk/R/concUnitConv.R")
>
> concUnitConv() # Wrong input
Error in concUnitConv() : Source concentration unit is not valid.
> concUnitConv("kg/L", "g/L") # Wrong input
Error in concUnitConv("kg/L", "g/L") : Source amount is not supported.
> concUnitConv("g/kL", "g/L") # Wrong input
Error in concUnitConv("g/kL", "g/L") : Volume unit is not supported.
>
> concUnitConv("mg/L", "ug/mL")

1
> Theoph$conc * concUnitConv("mg/L", "ug/L")
  [1]   740  2840  6570 10500  9660  8580  8360  7470  6890  5940  3280     0  1720  7910  8310
 [28]  7500  6200  5300  4900  3700  1050     0  1890  4600  8600  8380  7540  6880  5780  5330
 [55]  1570     0  1290  3080  6440  6320  5530  4940  4020  3460  2780   920   150   850  2350
 [82]  7560  6590  5880  4730  4570  3000  1250     0  7370  9030  7140  6330  5660  5670  4240
[109]  5680  2420     0  4860  7240  8000  6810  5870  5220  4450  3620  2690   860     0  1250
> Theoph$conc * concUnitConv("mg/L", "mg/mL")
  [1] 0.00074 0.00284 0.00657 0.01050 0.00966 0.00858 0.00836 0.00747 0.00689 0.00594 0.00328 0
 [21] 0.00301 0.00090 0.00000 0.00440 0.00690 0.00820 0.00780 0.00750 0.00620 0.00530 0.00490 0
 [41] 0.00578 0.00533 0.00419 0.00115 0.00000 0.00202 0.00563 0.01140 0.00933 0.00874 0.00756 0
 [61] 0.00553 0.00494 0.00402 0.00346 0.00278 0.00092 0.00015 0.00085 0.00235 0.00502 0.00658 0
 [81] 0.00731 0.00756 0.00659 0.00588 0.00473 0.00457 0.00300 0.00125 0.00000 0.00737 0.00903 0
[101] 0.00289 0.00522 0.00641 0.00783 0.01021 0.00918 0.00802 0.00714 0.00568 0.00242 0.00000 0
[121] 0.00086 0.00000 0.00125 0.00396 0.00782 0.00972 0.00975 0.00857 0.00659 0.00611 0.00457 0
> Theoph$conc * concUnitConv("mg/L", "mmol/L") # Wrong input
Error in concUnitConv("mg/L", "mmol/L") :
  Positive molecular weight should be given.
> Theoph$conc * concUnitConv("mg/L", "mmol/L", MW=-100) # Wrong input
Error in concUnitConv("mg/L", "mmol/L", MW = -100) :
  Positive molecular weight should be given.
> Theoph$conc * concUnitConv("mg/L", "mM", MW=180.164) # Wrong input
Error in concUnitConv("mg/L", "mM", MW = 180.164) :
  Target concentration unit is not valid.
> Theoph$conc * concUnitConv("mg/L", "mmol/L", MW=180.164)
  [1] 0.0041073688 0.0157634156 0.0364667747 0.0582802336 0.0536178149 0.0476232766 0.046402166
 [13] 0.0095468573 0.0439044426 0.0461246420 0.0462356520 0.0380209143 0.0337470305 0.029972691
 [25] 0.0382984392 0.0455140872 0.0432938878 0.0416287383 0.0344130903 0.0294176417 0.027197442
 [37] 0.0477342865 0.0465131769 0.0418507582 0.0381874292 0.0320818810 0.0295841567 0.023256588
 [49] 0.0517861504 0.0485113563 0.0419617682 0.0393530339 0.0327479408 0.0242556782 0.008714282
```

```
 [61] 0.0306942563 0.0274194623 0.0223130037 0.0192047246 0.0154303856 0.0051064586 0.00083257
 [73] 0.0369663196 0.0291401168 0.0243666881 0.0195932595 0.0063830732 0.0000000000 0.01692902
 [85] 0.0262538576 0.0253657778 0.0166514953 0.0069381230 0.0000000000 0.0409071735 0.05012100
 [97] 0.0228125486 0.0175395751 0.0062165582 0.0013321196 0.0160409405 0.0289736018 0.03557869
[109] 0.0315268311 0.0134322062 0.0000000000 0.0269754224 0.0401856087 0.0444039875 0.03779889
[121] 0.0047734287 0.0000000000 0.0069381230 0.0219799738 0.0434048978 0.0539508448 0.05411735
> Theoph$mM = Theoph$conc * concUnitConv("mg/L", "mmol/L", MW=180.164)
> Theoph$mM * concUnitConv("mmol/L", "ug/L", MW=180.164)
  [1]   740  2840  6570 10500  9660  8580  8360  7470  6890  5940  3280     0  1720  7910  8310
 [28]  7500  6200  5300  4900  3700  1050     0  1890  4600  8600  8380  7540  6880  5780  5330
 [55]  1570     0  1290  3080  6440  6320  5530  4940  4020  3460  2780   920   150   850  2350
 [82]  7560  6590  5880  4730  4570  3000  1250     0  7370  9030  7140  6330  5660  5670  4240
[109]  5680  2420     0  4860  7240  8000  6810  5870  5220  4450  3620  2690   860     0  1250
> Theoph$mM * concUnitConv("mmol/L", "ug/mL", MW=180.164)
  [1]  0.74  2.84  6.57 10.50  9.66  8.58  8.36  7.47  6.89  5.94  3.28  0.00  1.72  7.91  8.31
 [28]  7.50  6.20  5.30  4.90  3.70  1.05  0.00  1.89  4.60  8.60  8.38  7.54  6.88  5.78  5.33
 [55]  1.57  0.00  1.29  3.08  6.44  6.32  5.53  4.94  4.02  3.46  2.78  0.92  0.15  0.85  2.35
 [82]  7.56  6.59  5.88  4.73  4.57  3.00  1.25  0.00  7.37  9.03  7.14  6.33  5.66  5.67  4.24
[109]  5.68  2.42  0.00  4.86  7.24  8.00  6.81  5.87  5.22  4.45  3.62  2.69  0.86  0.00  1.25
>
>
```

A.2 Assignment 2

자신이 직접 R package를 만드는 것이 과제 2입니다.

- 제출기한 : 2017-06-21 18:00
- 제출방법 : R package를 shan@acp.kr[4] ksbae@acp.kr[5] 메일로 보내십시오. CRAN에 올리면 더 좋습니다.

[4] mailto:shan@acp.kr

[5] mailto:ksbae@acp.kr

B

As-is R Files

교수님께서 주신 원본 R 파일 입니다.

B.1 Lecture 3

```
#####################################################
##---------------------------------------------##
##                   Graphics                  ##
##---------------------------------------------##
#####################################################

# 상위수준 그림 함수는 그림을 생성한다.
# 하위수준 그림 함수는 기존의 그림에 그림을 추가한다.

## 상위수준 그림 함수의 주요 인자 (arguments) ###

# main : 제목
# xlab/ylab : x축 및 y축 레이블
# xlim/ylim : x축 및 y축 범위
# col : 색깔
# lty : 선 모양
# pch : 점 모양
# cex : 그림 성분의 크기
# lwd : 선 굵기
# type : 그림 타입


#####################################################
##########      상위수준 그림 함수    ############
#####################################################
```

```
WD <- "D:\\AMC\\Education\\UU\\2017\\R\\Graphics\\"

setwd(WD)

dta <- read.csv("PK.csv")
head(dta)
str(dta)

################ scatter plot ###################

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0])

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], log="y")

plot(dta$TIME[dta$MDV==0], log(dta$DV[dta$MDV==0]))

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0]
    , xlab="Time (hr)", ylab="Concentration (ng/mL)"
    , type="o", pch=2, col=1, main="PK time-course of Drug X"
    , xlim =c(-2,218), ylim=c(0,80))

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], axes=F,
    , xlab="Time (hr)", ylab="Concentration (ng/mL)"
    , type="o", pch=2, col=1, main="PK time-course of Drug X"
    , xlim =c(-2,218), ylim=c(0,80))
axis(1, at=seq(0, 218, 24))
axis(2)
box()



################## Histogram ####################

d.demog <- read.csv("DEMOG.csv")

# histogram
hist(d.demog$HT)

hist(d.demog$HT, breaks=10)
hist(d.demog$HT, nclass=10)

# with density line
hist (d.demog$HT, probability=TRUE, breaks=10)
lines(density(d.demog$HT))
```

```
hist (d.demog$HT, probability=TRUE, breaks=9, xaxt="n"
      , main="Histogram for Height", xlab="Height (cm)", ylab="Probability (%)")
axis(1, at=seq(min(d.demog$HT), max(d.demog$HT), 3))
lines(density(d.demog$HT))


hist (d.demog$HT, probability=TRUE, breaks=9, xaxt="n"
      , main="Histogram for Height", xlab="Height (cm)", ylab="Probability (%)"
      , col = "lightblue", border = "pink")
axis(1, at=seq(min(d.demog$HT), max(d.demog$HT), 3))
lines(density(d.demog$HT))


############### Box-Whisker Plot #################

# Box-and-Whisker Plot

boxplot(d.demog$WT)

boxplot(d.demog$WT ~ d.demog$SEX)

boxplot(split(d.demog$WT, d.demog$SEX))

boxplot(WT ~ SEX, data=d.demog)

boxplot(d.demog$WT ~ d.demog$SEX
        , names=c("Male","Female"), ylab="AGE, year", ylim=c(min(d.demog$WT)-2, max(d.demog$WT)
            , col="pink")

boxplot(d.demog$WT ~ d.demog$SEX
        , names=c("Male","Female"), ylab="AGE, year", ylim=c(min(d.demog$WT)-2, max(d.demog$WT)
            , col=c("lightblue", "salmon"), width=c(0.6, 1))

#varwidth: if varwidth is TRUE, the boxes are drawn with widths proportional
#to the square-roots of the number of observations in the groups.

boxplot(d.demog$WT ~ d.demog$SEX
        , names=c("Male","Female"), ylab="AGE, year", ylim=c(min(d.demog$WT)-2, max(d.demog$WT)
            , col=c("lightblue", "salmon")
            , varwidth=TRUE)
```

```
################## Bar Plot ####################

barplot(d.demog$HT)

VADeaths

barplot(VADeaths, border = "dark blue")

barplot(VADeaths, col = rainbow(20))

barplot(VADeaths, col = heat.colors(8))

barplot(VADeaths, col = gray.colors(4))

barplot(VADeaths, col = gray.colors(4), log="x")
barplot(VADeaths, col = gray.colors(4), log="y")
barplot(VADeaths, col = gray.colors(4), log="xy")



################## pie chart ####################

drug.X.market <- c(0.12, 0.29, 0.32, 0.22, 0.11, 0.28)
names(drug.X.market) <- c("South Korea","China","USA","Japan","Austria","EU")
pie(drug.X.market)


################ matplot 함수 ##################

# matrix와 column 사이의 그림

pct.95 <- read.csv("pct95.csv")
matplot(pct.95[,1], pct.95[,2:ncol(pct.95)], pch=1)

matplot(pct.95[,1], pct.95[,2:ncol(pct.95)], pch=1, col=c(1,2,1), type="l", lty=1, lwd=c(1,2,1)



###### Scatter plot matrices (pairs plots) ######

pairs(d.demog)

#add a loess smoother, type:
```

```r
pairs(d.demog, panel = panel.smooth)

  panel.cor <- function(x, y, digits=2, prefix="", cex.cor)
   {
       usr <- par("usr"); on.exit(par(usr))
       par(usr = c(0, 1, 0, 1))
       r = (cor(x, y))
       txt <- format(c(r, 0.123456789), digits=digits)[1]
       txt <- paste(prefix, txt, sep="")
       if(missing(cex.cor)) cex <- 1.5
       text(0.5, 0.5, txt, cex = 1.5)
   }

pairs(d.demog, lower.panel=panel.smooth, upper.panel=panel.cor)



################################################
##              하위수준 그림 함수            ##
################################################

# points : 점추가
# lines : 선 추가
# abline : 기준선 추가
# mtext : 텍스트 추가
# legend : 설명(legend) 추가
# polygon : polygon 추가


############ 점, 선, 설명 추가 하기 #############

plot(pct.95$TIME, pct.95$PCT50, main="PK of Drug X"
     , type="l", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80), lty=1, col="red", lwd=2)


plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  #abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")
```

```
################## polygon 함수 ###################

plot(c(1, 10), c(1, 6), type = "n")
polygon(c(2,8,8,2), c(5,4,3,2), col="lightgreen")


plot(c(1, 9), 1:2, type = "n")
polygon(1:9, c(2,1,2,1,1,2,1,2,1),
        col = c("red", "blue"),
        border = c("green", "yellow"),
        lwd = 3, lty = c("dashed", "solid"))


################## 그림 출력하기 ##################

#--pdf graphics devices
pdf("PK_of_Drug_X.pdf")

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  #abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")

dev.off()


#--PNG graphics devices
png("PK_of_Drug_X.png")

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  #abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")
```

```
dev.off()



#--Windows graphics devices
win.metafile("PK_of_Drug_X.wmf")

plot(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], main="PK of Drug X"
     , type="n", xlab="Time (h)", ylab="Concentration (ng/ml)"
     , ylim=range(0,80))
points(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], pch = 16, cex=0.8)
lines(dta$TIME[dta$MDV==0], dta$DV[dta$MDV==0], col="black", lwd=1)
abline(40, 0, col="red", lty=2)                                  #abline(a,b): y=a+b*x
legend("topright", legend=c("Individual concentrations")
       , lty=1, col="black")

dev.off()
```

B.2 Lecture 4

```
# 2017-03-29

setwd("D:/Rt")
dir()

mydata = read.csv("MyData2017.csv", as.is=TRUE)

Theoph
library(lattice) # trellis

xyplot(conc ~ Time | Subject, data=Theoph)

xyplot(conc ~ Time | Subject, data=Theoph, type="b")

Theoph[,"ID"] = as.numeric(as.character(Theoph[,"Subject"]))

xyplot(conc ~ Time | ID, data=Theoph, type="b")
```

```
xyplot(conc ~ Time | as.factor(ID), data=Theoph, type="b")

write.csv(Theoph, "Theoph.csv", row.names=FALSE, quote=FALSE, na="")



IDs = sort(unique(Theoph[,"ID"])) ; IDs
nID = length(IDs) ; nID

demog = unique(Theoph[,c("ID","Wt")])
colnames(demog) = c("ID", "BWT")
write.csv(demog, "1-demog.csv", row.names=FALSE, quote=FALSE, na="")

DV = Theoph[,c("ID","Time", "conc")]
colnames(DV) = c("ID", "TIME", "DV")
write.csv(DV, "3-DV.csv", row.names=FALSE, quote=FALSE, na="")

adm = cbind(IDs, rep(0, nID), rep(320, nID))
colnames(adm) = c("ID", "TIME", "AMT")
write.csv(adm, "2-adm.csv", row.names=FALSE, quote=FALSE, na="")



demog = read.csv("1-demog.csv", as.is=TRUE)
adm = read.csv("2-adm.csv", as.is=TRUE)
dv = read.csv("3-dv.csv", as.is=TRUE)

AdmDv = merge(adm, dv, by=intersect(colnames(adm), colnames(dv)), all=TRUE)

DataAll = merge(demog, AdmDv, by=c("ID"), all=TRUE)
```

B.3 Lecture 5

```
# 2017-04-05 R-intro.pdf Chapter 08

pois
# ?dbeta
dnorm(0)
pnorm(0)
1 - pnorm(1.96)
```

```
# ?pnorm
pnorm(1.96, lower.tail=FALSE)
qnorm(0.5)
qnorm(0.975)
format(qnorm(0.975), digits=22)
rnorm(5)
rnorm(5, 10, 1)
x = rnorm(100, 10, 1)
mean(x)
sd(x)

2*pt(-2.43, df = 13)

2*pt(-2.43, df = 1000)

qnorm(0.995)
qf(0.01, 2, 7, lower.tail = FALSE)

# ?fivenum
faithful
str(faithful)
eruptions
attach(faithful)
eruptions
waiting

stem(waiting)
sort(eruptions)

hist(eruptions)
hist(eruptions, seq(1.6, 5.2, 0.2), prob=TRUE)
lines(density(eruptions, bw=0.1))
rug(eruptions)
# ?hist
# ?density
lines(density(eruptions, bw="SJ"), lty=3)
plot(ecdf(eruptions), do.points=FALSE, verticals=TRUE)
# ?plot
ecdf(eruptions)
x = ecdf(eruptions)
x
str(x)
x()
plot(ecdf(eruptions), do.points=FALSE)
```

```
plot(ecdf(eruptions))
long <- eruptions[eruptions > 3]
x <- seq(3, 5.4, 0.01)
pnorm(x, mean=mean(long), sd=sqrt(var(long)))

# ?par
x <- rt(250, df = 5)
qqnorm(x); qqline(x)

curve(dnorm, -5, 5)
y = density(x)
lines(y, lty=3)
# ?ppoints
ppoints(250)
ppoints(10)

qqplot(qt(ppoints(250), df = 5), x, xlab = "Q-Q plot for t dsn")
windows()
qqplot(qt(runif(250), df = 5), x, xlab = "Q-Q plot for t dsn")
# ?shapiro.test
# ?ks.test
# ?t.test


A = c(79.98, 80.04, 80.02, 80.04, 80.03, 80.03, 80.04, 79.97, 80.05, 80.03, 80.02, 80.00, 80.02
B = c(80.02, 79.94, 79.98, 79.97, 79.97, 80.03, 79.95, 79.97)
boxplot(A, B)
t.test(A, B)

var.test(A, B)
t.test(A, B, var.equal=TRUE)
wilcox.test(A, B)
plot(ecdf(A), do.points=FALSE, verticals=TRUE, xlim=range(A, B))
plot(ecdf(B), do.points=FALSE, verticals=TRUE, add=TRUE)
ks.test(A, B)

# Chapter 9 Grouping, loops and conditional execution

# { } does grouping
# Usefulness of loops: for >> while >> repeat
for (i in 1:10) {
  print(2*i)
}
```

```r
for (i in 1:10) print(2*i)

#while (    ) {
## Statements
#}

# # if ~ else ~
# if (   ) {
# # Statements 1
# } else {
# # Statements 2
# }
#
# if (    ) # Statement1
# else # Statement2
#
# if (   ) {
# # Statements 1
# } else if (   ) {
# # Statements 2
# } else if (   ) {
# # Statements 3
# } else {
# # Statements 4
# }



#
#

# Chapter 10 Writing your own functions

Square = function(x=0)
{
  return(x*x)
}

twosam = function(y1, y2)
{
  n1  = length(y1)
  n2  = length(y2)
  yb1 = mean(y1)
  yb2 = mean(y2)
  s1  = var(y1)
```

```
  s2  = var(y2)
  s   = ((n1 - 1)*s1 + (n2 - 1)*s2)/(n1 + n2 - 2)
  tst = (yb1 - yb2)/sqrt(s*(1/n1 + 1/n2))
  return (tst)
}

x = rnorm(10)
y = rt(10, 5)

twosam(x, y)

T.test = function(y1, y2)
{
  n1  = length(y1)
  n2  = length(y2)
  yb1 = mean(y1)
  yb2 = mean(y2)
  s1  = var(y1)
  s2  = var(y2)
  s   = ((n1 - 1)*s1 + (n2 - 1)*s2)/(n1 + n2 - 2)

  tst = (yb1 - yb2)/sqrt(s*(1/n1 + 1/n2))
  DF = n1 + n2 - 2
  p.val = 2*(1 - pt(abs(tst), df=DF))

  Res = list(tst, DF, p.val, yb1, yb2)
  names(Res) = c("t", "df", "p-value", "mean of x", "mean of y")

  return (Res)
}

res = T.test(x, y)
t.test(x, y)


bslash = function(X, y)
{
  X = qr(X)
  return (qr.coef(X, y))
}

regcoeff = bslash(Xmat, yvar)
```

```
"%^%" = function(S, pow) with(eigen(S), vectors %*% (abs(values)^pow * t(vectors)))

M = matrix(c(2,1,1,2), nrow=2) ; M
M %^% 0.5
sqrtM = M%^%0.5 ; sqrtM
sqrtM %*% sqrtM




area = function(f, a, b, eps=1.0e-06, lim=10)
{
  fun1 = function(f, a, b, fa, fb, a0, eps, lim, fun)
  {
  ## function 'fun1'is only visible inside 'area'
    d = (a + b)/2
    h = (b - a)/4
    fd = f(d)
    a1 = h * (fa + fd)
    a2 = h * (fd + fb)
    if (abs(a0 - a1 - a2) < eps || lim == 0)
      return (a1 + a2)
    else {
      return (fun(f, a, d, fa, fd, a1, eps, lim - 1, fun) + fun(f, d, b, fd, fb, a2, eps, lim -
    }
  }
  fa = f(a)
  fb = f(b)
  a0 = ((fa + fb) * (b - a))/2
  fun1(f, a, b, fa, fb, a0, eps, lim, fun1)
}

area(dnorm, 0, 1)
integrate(dnorm, 0, 1)
pnorm(1) - pnorm(0)


f = function(x)
{
  y = 2*x
  print(x)
  print(y)
  print(z)
```

```
}

f(1)
z = 3
f(1)


cube = function(n) {
  sq = function() n*n
  n*sq()
}

cube(5)

open.account = function(total)
{
  list(
    deposit = function(amount)
    {
      if(amount <= 0)
      stop("Deposits must be positive!\n")
      total <<- total + amount
      cat(amount, "deposited. Your balance is", total, "\n\n")
    },
    withdraw = function(amount)
    {
      if(amount > total)
      stop("You don't have that much money!\n")
      total <<- total - amount
      cat(amount, "withdrawn. Your balance is", total, "\n\n")
    },
    balance = function()
    {
      cat("Your balance is", total, "\n\n")
    }
  )
}

ross = open.account(100)
robert = open.account(200)

ross$balance()
robert$balance()
ross$deposit(50)
```

```
ross$balance()
ross$withdraw(500)



# More basic keywords and functions
1 %in% c(1,2,3,4)
5 %in% c(1,2,3,4)
is.finite(Inf)
prod(1:3)
cummax(1:10)
cummax(10:1)
# ?xor
x = 11:20
x
which(x==3)
which(x==13)

length(x)
y = "my string"
length(y)
nchar(y)
strsplit(y, " ")
strsplit(y, " ")[[1]]
substr(y, 4, 5)



sample(1:10)
sample(1:10, 20)
sample(1:10, 20, replace=TRUE)
sample(rep(1:10,2))
```

C

R Tips

- Handbook of statistical distributions with applications[1] (Krishnamoorthy, 2006)
- Materials
  - https://cran.r-project.org/manuals.html
  - https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf
- Changing defualt R console size and etc : 배균섭 교수님께서 알려주신 tip을 참고하여 video clip을 만들었습니다. 매일같이 마주하게 되는 R console이 너무 작게 느껴지셨다면 다음의 동영상을 참고하셔서 초기 세팅 (Rconsole 파일)을 바꿔서 해결할 수 있습니다. https://youtu.be/uSunEN8W5Mo

C.1 Using Coursera

> PAGK에 보낸 이메일을 그대로 옮겼습니다.

배균섭 교수님의 추천을 받아 다음과 같은 강의와 책을 공유하고자 합니다. Coursera.com에 유익한 R 강좌가 열렸습니다. "Mastering Software Development in R Specialization"이란 제목의 강좌인데 4개 Course를 무료로 들을 수 있게 되어있습니다. https://www.coursera.org/specializations/r 이것이 본래의 Link인데 여기서 각각의 Course를 찾거나 혹은 아래의 링크를 각각 클릭하여 하단에 나오는 "Audit"을 클릭하면 무료로 들을 수 있습니다.

Audit 버튼이 안보이신다고 하신 분들이 몇분 계셔서 첨언합니다. Coursera 회원가입하시고 로그인 한 뒤, Enroll Now를 누르시면 Audit 혹은 청강하기 라디오버튼을 보실 수 있습니다. 앱에서도 마찬가지입니다. 이외에도 코세라에는 많은 유익한 강의가 있는 것 같습니다. 다만 코스(Course)의 묶음인 "Specialization"에서는 유료등록(Enroll) 밖에

[1] http://www.stat.rice.edu/~dobelman/textfiles/DistributionsHandbook.pdf

없으므로 Certificate가 필요하지 않다면, 각각의 코스를 구글검색 혹은 코세라 내에서 검색해서 “Audit(청강)” 하시면 무료로 강의를 들을 수 있습니다.

- https://www.coursera.org/learn/r-programming-environment
- https://www.coursera.org/learn/advanced-r
- https://www.coursera.org/learn/r-packages
- https://www.coursera.org/learn/r-data-visualization

https://bookdown.org/rdpeng/RProgDA/ 이 링크는 무료로 공개된 강의 책자입니다. 강의를 듣지 않고 책으로 보고 싶으신 분은 참고하시면 됩니다.

D

Acknowledgement

이 웹북을 만드는데 도움을 주신 분들은 다음과 같습니다.

1. Dr. Jekyll
2. Hyde

Bibliography

Krishnamoorthy, K. (2006). *Handbook of statistical distributions with applications*. Chapman & Hall/CRC, Boca Raton.

Sarkar, D. (2017). *lattice: Trellis Graphics for R*. R package version 0.20-35.

Xie, Y. (2017). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.4.1.

Index