
Introduction to R packages

Contents

1.

About R Packages

2.

R Download & Installation

3.

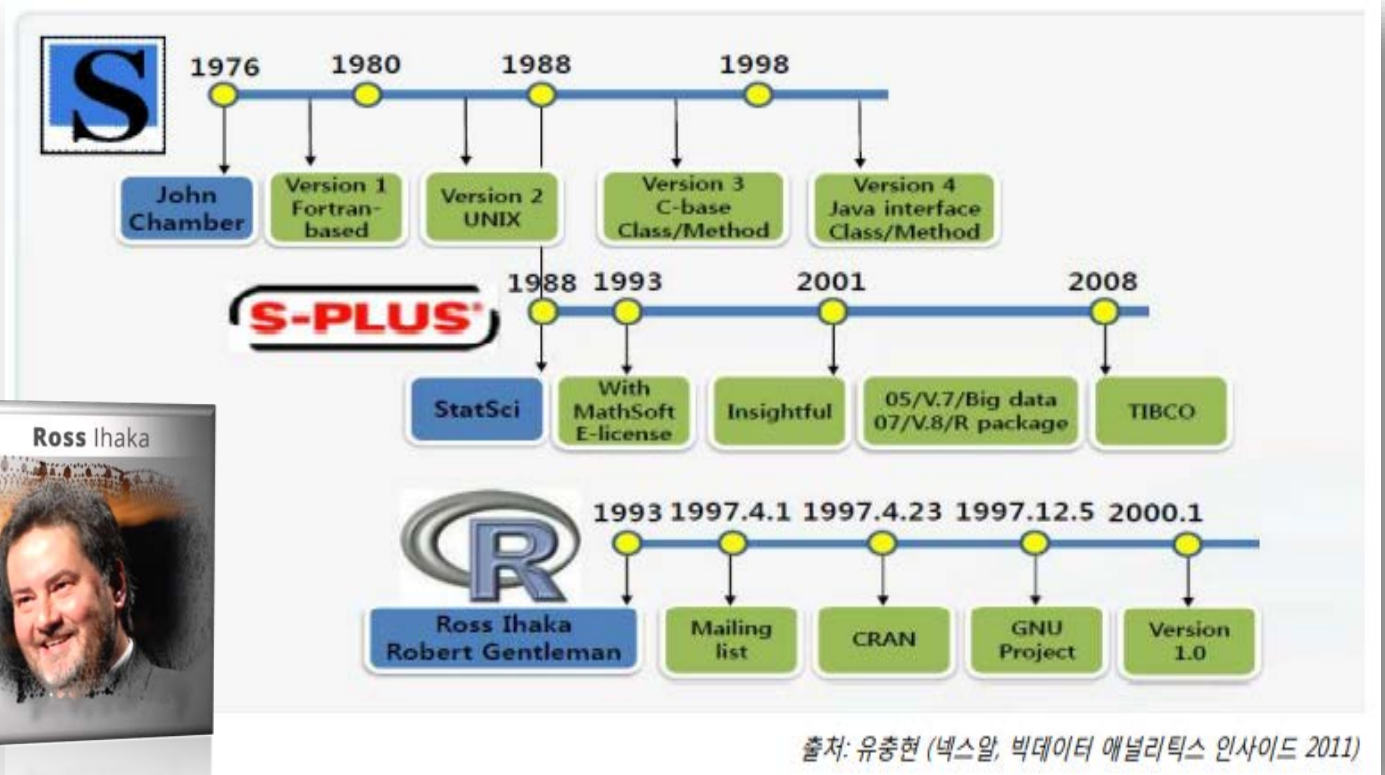
R-Programming

4.

R Graphics

1. About R packages

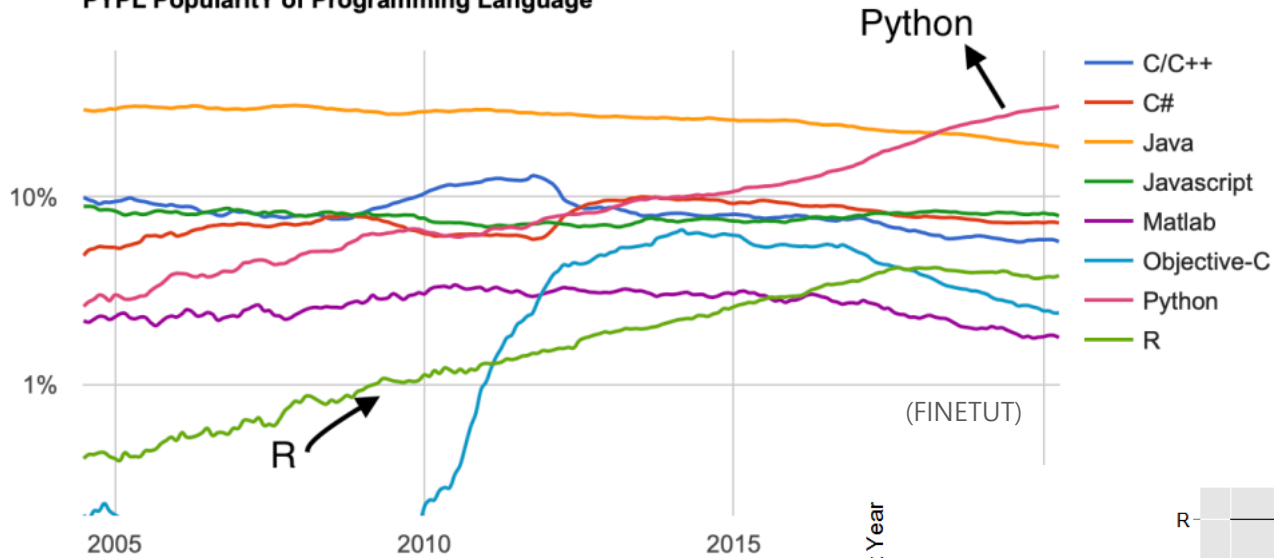
Founders of R



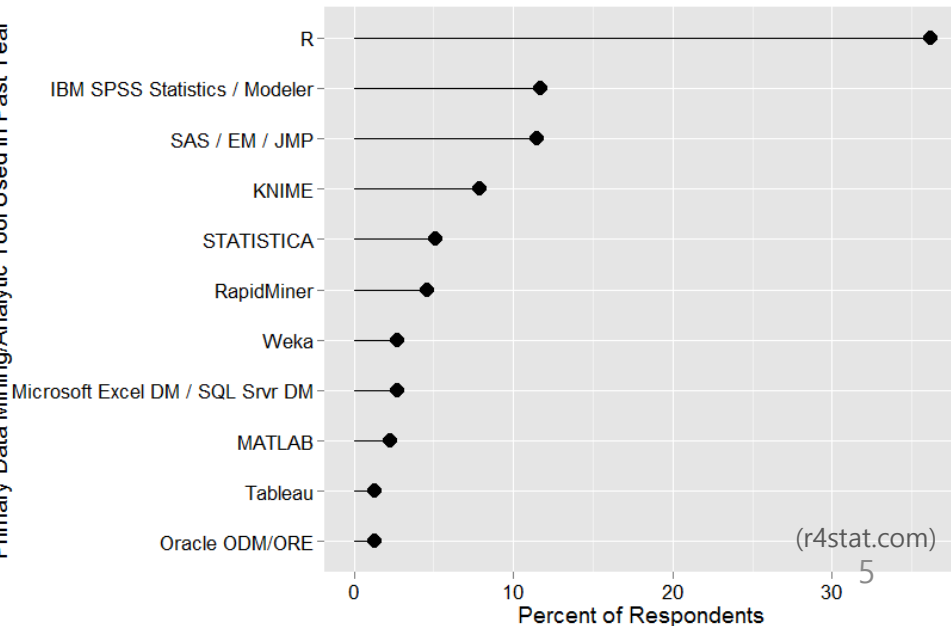
- **R 패키지**는 1990년대 중반 **Robert Gentleman** (23andMe)과 **Ross Ihaka** (Univ. of Auckland)에 의해 free software로 처음 개발되었다.
- 1976년 Bell Lab에서 개발된 **S Language**가 전신이다.

Popularity of R

PYPL Popularity of Programming Language



Primary Data Mining/Analytic Tool Used in Past Year



(r4stat.com)

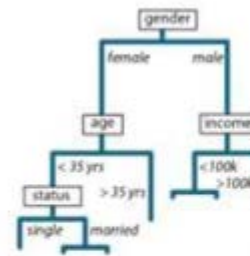
기업체에서의 R의 활용

- 빅데이터 (Big Data)를 활용하는 관련기업의 분석 플랫폼 엔진으로 사용 중
- 특히, Google과 Facebook은 R을 자사의 주된 분석 플랫폼으로 활용



How Google and Facebook are using R

by mike | February 19th, 2009



(March 26th Update: Video now available)

Last night, I moderated our Bay Area R Users Group kick-off event with a panel discussion entitled "The R and Science of Predictive Analytics", co-located with the Predictive Analytics World conference here in SF.

The panel comprised of four recognized R users from industry:

Bo Cowgill, Google

Itamar Rosenn, Facebook

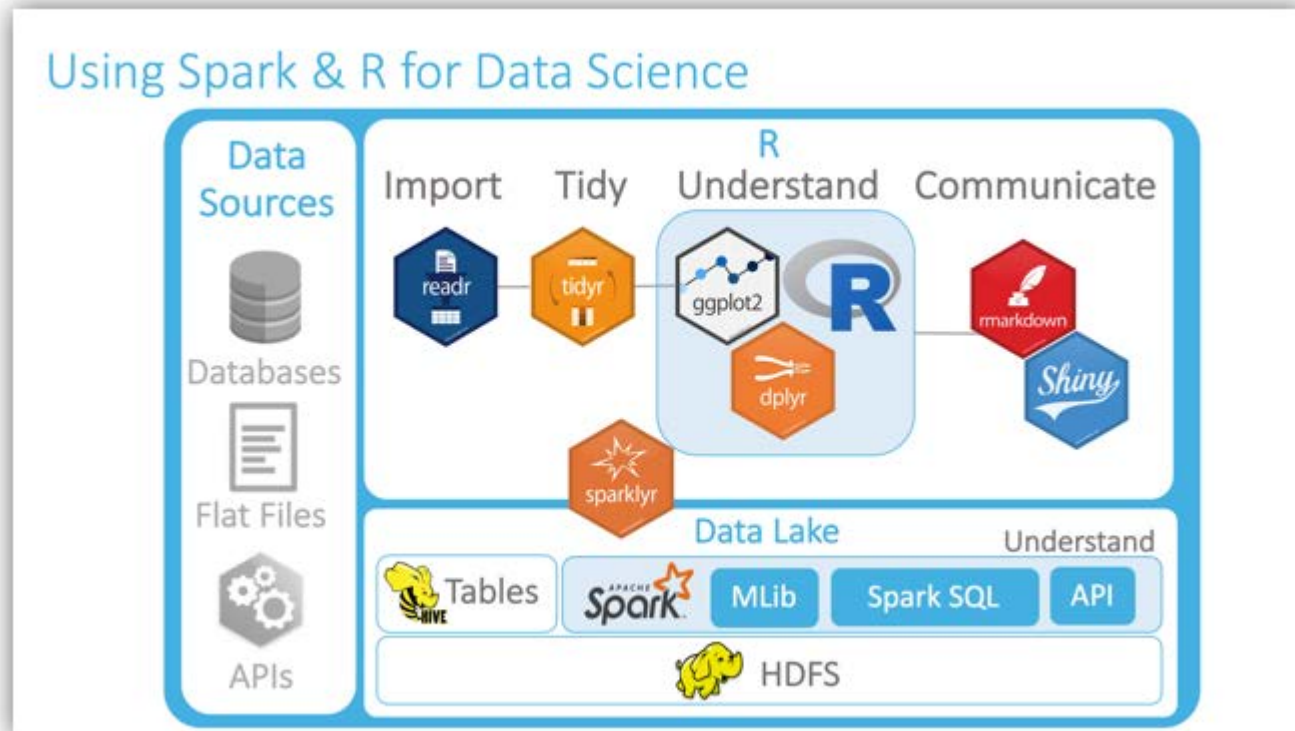
David Smith, Revolution Computing

Jim Porzak, The Generations Network (and Co-Chair of our R Users Group)

The panelists were asked to explain how they use R for predictive analytics within their firms, its strengths and weaknesses as a tool, and provide a case study. What follows is my summary with comments.

빅데이터 분석용 소프트웨어

- 빅데이터 분석을 위한 아키텍처 전반에 걸쳐 공통적인 분석 플랫폼으로 자리 잡음



R through Excel

- 빅데이터 뿐 아니라 통계 비전문가들을 위한 플랫폼도 다양하게 개발됨



About R

R is a language and environment for statistical computing and graphics.

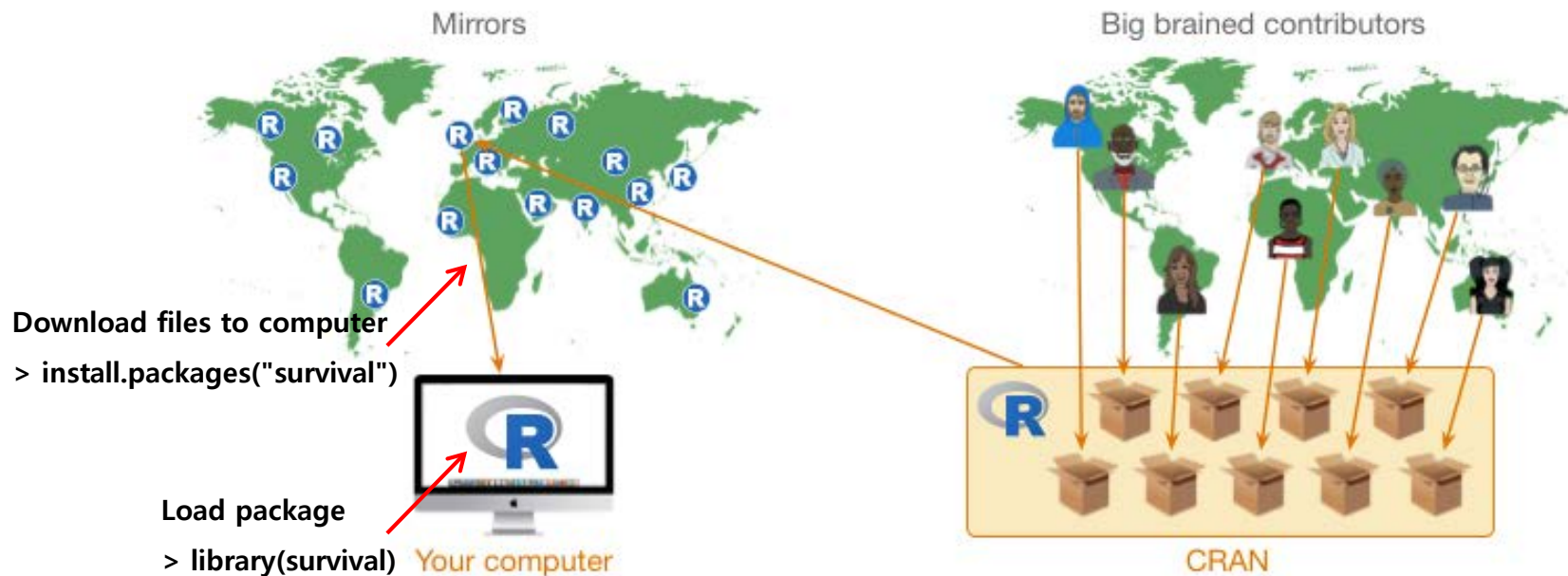
- R은 통계 계산을 위한 시스템으로 데이터 분석 및 그래프 작성을 위한 환경을 지닌다.
- R은 free open source 이다.
 - 언제 어디서든 다운로드 및 설치가 가능
 - Windows, Linux, Unix, Mac OS 등 다양한 운영체제에서 동작
 - 누구나 패키지를 만들어 다른 유저들과 공유 가능
 - Java, Python, Visual Studio 등 다양한 개발 언어 및 플랫폼과 연동
- 데이터 입출력, 데이터 처리, 데이터 분석, 그래프 작성 등을 위한 수많은 알고리즘 및 방법론을 제공한다.

Characteristics of R

- In-Memory Computing
 - 장점: 빠른 처리 속도
 - 단점: H/W 메모리 크기에 영향을 받음
- Object-oriented programming
 - 데이터와 함수가 object로 관리
- Packages
 - 최신의 알고리즘 및 방법론을 적용
 - 다양한 함수 및 데이터 내장
 - Help 창에서 Examples을 바로 사용 가능

R Packages

- R은 패키지 라는 구성요소를 가지고 있으며, 패키지 안에는 여러 함수 형태로 된 코드들이 담겨 있다. 패키지는 특정 기업/기관에서 주관하지 않고 사용하는 모든 user들로부터 개발되어지고 있다.



(Milton Rocks)

R CRAN

What is the CRAN mirror?

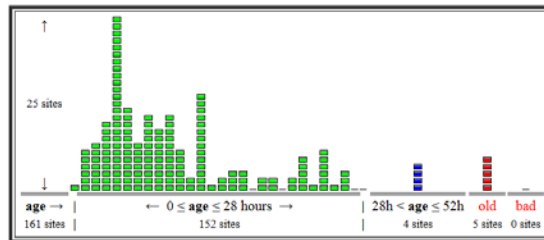
- The **C**omprehensive **R** **A**rchive **N**etwork
- 접근하기 쉽고 주기적으로 자주 업데이트되는 CRAN 선택
- https://cran.r-project.org/mirmon_report_release.html
- http는 안전하게 전송할 수 있고, https 빠르게 전송 가능하다.

Currently, the CRAN package repository features 12503 available packages.
 Currently, the CRAN package repository features 13697 available packages.
 Currently, the CRAN package repository features 14972 available packages.
 Currently, the CRAN package repository features 16214 available packages.

the status of [CRAN/bin/windows/contrib/r-release](https://cran.r-project.org/bin/windows/contrib/r-release) mirrors

date : Mon May 14 08:08:20 2018 (UTC)
 last check : Mon May 14 08:08:20 2018 (UTC)

age histogram



units: ■■■ represent one mirror site.

regions

[ar](#) [asia](#) [at](#) [au](#) [be](#) [bg](#) [br](#) [ca](#) [ch](#) [cl](#) [cn](#) [co](#) [cz](#) [de](#) [dk](#) [dz](#) [ec](#) [ee](#) [es](#) [fr](#) [gr](#) [hu](#) [id](#) [ie](#) [in](#) [ir](#) [is](#)

report

161 sites in 48 regions				
0 bad -- 5 older than 2.2 days -- 4 unreachable for more than 5 hours				
last probes : 157 were ok, 4 had no time				
mean mirror age is 21 hours, std_dev 3.3 days, median 8 hours				
CRAN/bin/windows/contrib/r-release site -- home	type	mirror age, daily stats	last probe, probe stats	last stat
Italy				
cran_mirror.garr.it @	http	4 hours	3 hours	ok
dssm.unipa.it @	http	16 hours	2 hours	ok
cran.stat.unipd.it @	http	10 hours	3 hours	ok
cran.stat.unipd.it @	https	10 hours	3 hours	ok
Japan				
cran.ism.ac.jp @	http	1 hour	renewed	ok
cran.ism.ac.jp @	https	3 hours	2 hours	ok
ftp.yz.yamagata-u.ac.jp @	https	6 hours	2 hours	ok
Korea, Republic of				
healthstat.snu.ac.kr @	http	25 hours	2 hours	ok
cran.biodisk.org @	http	19 hours	1 hour	ok
cran.biodisk.org @	https	19 hours	renewed	ok

R packages vs. other programs

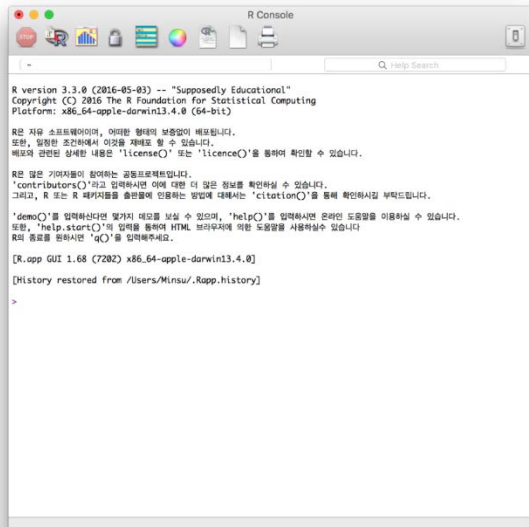
	SPSS	SAS	R
Current Version (Sep. 2020)	26.0	9.4	4.0.2
Creator	Norman H. Nie, Dale H. Bent, and Hadlai (Tex) Hull	Jim Goodnight and Jim Barr, North Carolina State University	Ross Ihaka and Robert Gentleman, University of Auckland, New Zealand and the R foundation
Year of Released	1968	Mass distributed since 1972	1995
User Interface	Point-and-click	Based code	Based code
Cost	Expensive (monthly)	Expensive (yearly)	Open source
Advantage	Simple to handle Similarity with Excel	Large scale data	Free

2. R Download & Installation

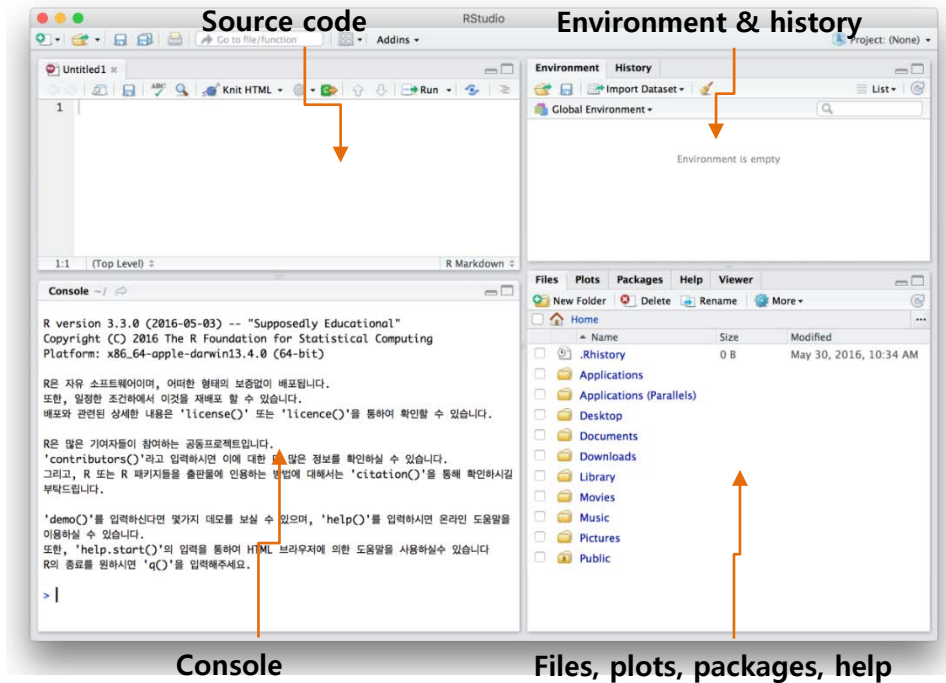
Download of R

- R과 R studio 다운로드는 아래 주소에서 가능하다.

- R base: <http://cran.r-project.org/>



- R studio: <http://www.rstudio.com/>



Installation of R (www.r-project.org)

The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX-like operating systems, Windows, and Mac OS.

Download CRAN

Download R for Windows

base

Download R 4.0.2 for Windows (84 megabytes, 32/64 bit)

Installation and other instructions
New features in this version

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, windows: both [graphical](#) and [command line versions](#) are available.

Information and Database Systems Laboratory, Pukyong National University
Yeungnam University
Bigdata Campus, Seoul Metropolitan Government
Graduate School of Public Health, Seoul National University, Seoul
The Genome Institute of UNIST (Ulsan National Institute of Science and Technology)
The Genome Institute of UNIST (Ulsan National Institute of Science and Technology)

Installation of R studio

(<https://www.rstudio.com/products/rstudio/download/>)



[Products](#) [Resources](#) [Pricing](#) [About Us](#) [Blogs](#) [Q](#)

Choose Your Version of RStudio

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace. [Learn More about RStudio features.](#)



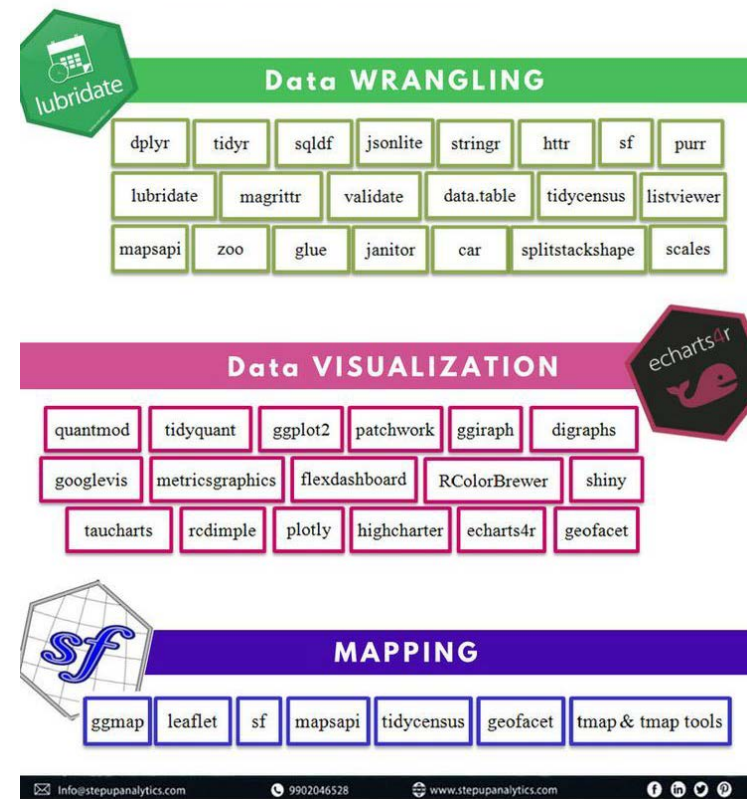
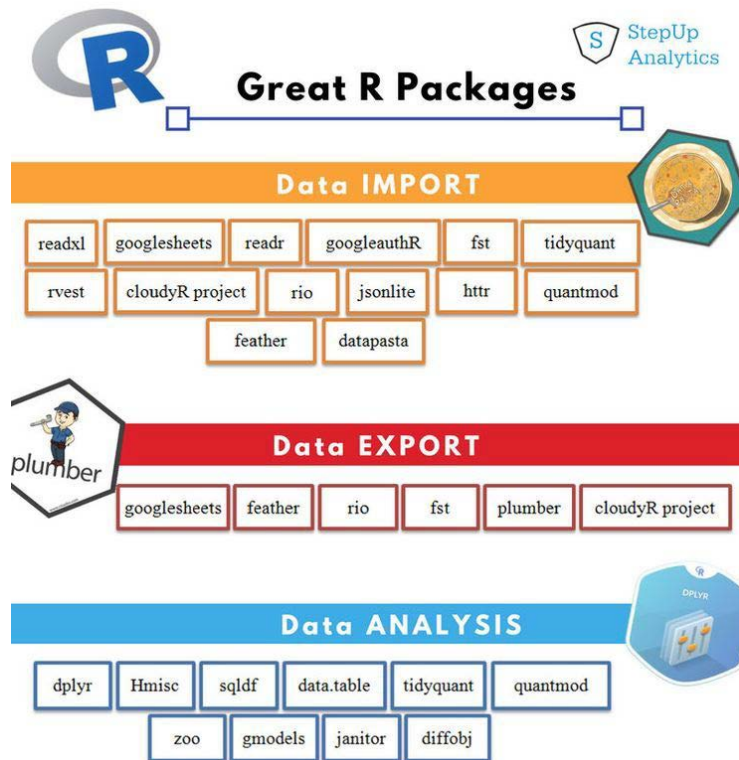
RStudio Desktop Open Source License	RStudio Desktop Commercial License	RStudio Server Open Source License	RStudio Server Pro Commercial License	RStudio Server Pro + RStudio Connect Commercial License
FREE	\$995 per year	FREE	\$9,995 per year	\$29,995 per year
DOWNLOAD Learn More	BUY Learn More	DOWNLOAD Learn More	DOWNLOAD Learn More	TALK Learn More
Integrated Tools for R				

Installers for Supported Platforms

Installers	Size	Date	MD5
RStudio 1.1.463 - Windows Vista/7/8/10	85.8 MB	2018-10-29	58b3d796d8cf96fb8580c62f46ab64d4
RStudio 1.1.463 - Mac OS X 10.6+ (64-bit)	74.5 MB	2018-10-29	a79032ba4d7daaa86a8da01948278d94
RStudio 1.1.463 - Ubuntu 12.04-15.10/Debian 8 (32-bit)	89.3 MB	2018-10-29	8a6755fa9fae2bafce289df3358aaf63

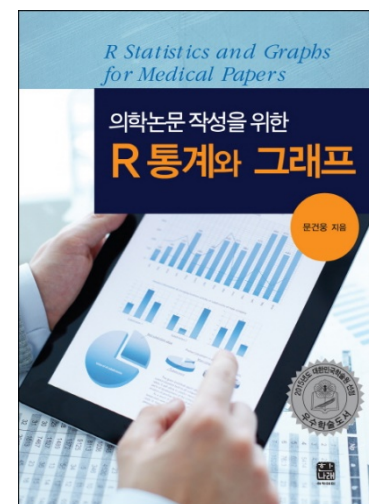
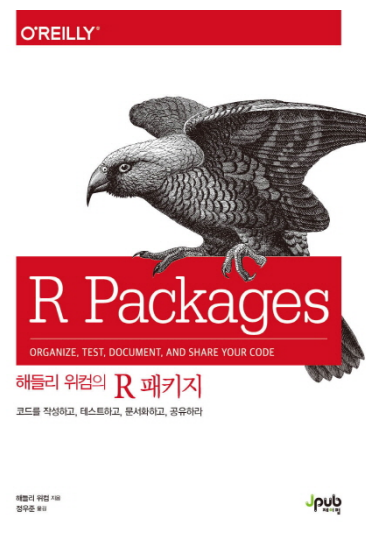
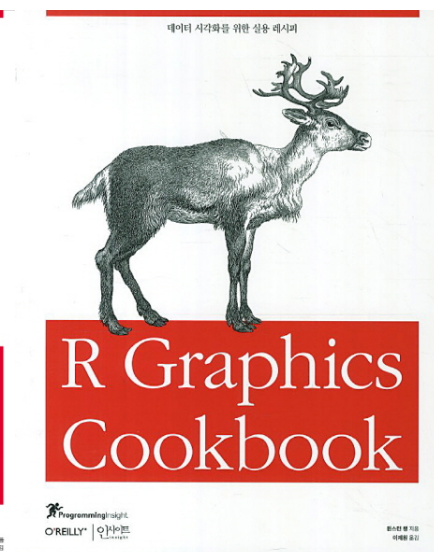
R Packages

- 목적에 따라 다음의 R 패키지들을 활용할 수 있습니다.



Books for R

- R Cookbook (Paul Teetor, 이제원 옮김)
- R Graphics Cookbook (Winston Chang, 이제원 옮김)
- 해들리 위컴의 R Packages (Hadley Wickham, 정우준 옮김)
- 의학논문작성을 위한 R통계와 그래프 (문건웅)



References for R

- <http://www.r-project.org>
- <http://r-project.kr> (한국 R 사용자 모임 – KRUG)
- <http://www.r-bloggers.com>
- <http://stackoverflow.com>
- <http://stats.stackexchange.com>
- <http://www.inside-r.org/>
- <http://www.r-statistics.com/>

Before beginning

- Window 64bit의 경우 "R x64 Version"을 사용하고 32bit의 경우 "R i386"을 사용함
- R Console 창에 직접 입력하는 것보다는 R Editor를 사용하는 것을 권장
- R Editor에서 코드 실행
 - R base: Ctrl+R or F5
 - R Studio: Ctrl+Enter (적용 후 cursor는 다음 line이동)
Alt+Enter (적용 후 cursor는 제자리)
- R Editor를 닫으면 기록된 code가 모두 삭제되므로 자주 저장을 해야 함
- R에서는 대소문자를 구분하기 때문에 주의해야 함 (예. Age, AGE, age는 모두 다른 변수)
- "#" 는 주석(annotation)을 작성하기 위한 시작 코드이며, 같은 line에서만 적용됨
- 다음 페이지부터 모든 코드 앞에는 ">" 표시가 있으며 해당 기호를 붙여 실행하면 오류 발생
- 슬라이드에 있는 " "와 R Editor에서 입력되는 " "가 다르기에 그대로 붙이면 오류 발생

3. R Programming

Help 창 활용

- 'library' 함수로 패키지 불러온 후 함수에 대한 help 창 호출

```
?Surv # library(survival) 먼저 수행  
help(Surv)  
help(package="survival")
```

- 설치되는 많은 패키지들 중에 keyword를 통해 함수 및 패키지 검색

```
??survival # or  
help.search("survival")
```

- 현재 사용 중인 workspace에서 특정 keyword를 포함하는 objects 호출

```
apropos("surv")  
apropos("^surv")
```

Working Directory

- Show & set the working directory

```
getwd() # Window에서는 '\' or '\ ' → '/' or '/'로 변경해야 함  
setwd("file path")
```

- 해당 폴더에 있는 파일 list 검색

```
dir() # lists files in the working directory
```

- 엑셀에서 새로운 폴더 생성하기

```
dir.create("file path/new folder")
```


Naming R Objects

- R 객체에 이름 지정하기
 - 문자 a~z, A~Z와 숫자 0~9, '.', '_'의 조합으로 구성
 - 대소문자는 구분하여 입력
 - 이름의 첫 자로 숫자와 '_'는 사용 불가
 - 'if', 'for'와 같이 R에서 사용되는 기본 명령어는 객체 이름으로 사용 불가

```
x_1 <- 1:7          # 소문자_숫자
x.1 <- c(1,2,3,4,5,6,7) # 소문자.숫자
X_1 <- c(1:7)        # 대문자_숫자
```

Types of R Objects

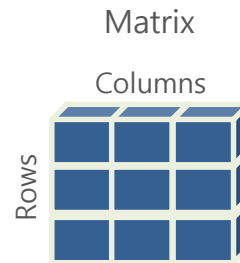
- 자료 내 변수 사용 시 R은 data type을 하나씩 정의하지 않는다.
- 각 object의 type은 자료를 load 하는 시점에서 결정되며, 별도로 type을 변경할 수 있다.

Common types are:

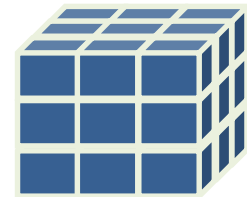
Vector



Matrix

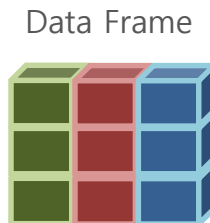


Array

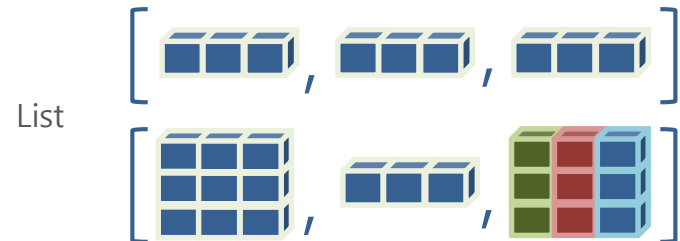


Array

Data Frame



List



Types of R Objects (cont'd)

- Scalar: numbers, character strings, logical values

```
a <- 5; b <- "hello"; c <- TRUE
```

- **Vector**: sets of one type of multiple scalar values

```
a <- c(1,2,3); b <- rep(1,5); c <- seq(1,9); d <- seq(1,9,3)
```

- Matrix: two-dimensional sets of one type of multiple scalar values

```
a <- matrix(c(1,2,3,4), ncol=2, nrow=2)
```

- Array: K-dimensional sets of one type of multiple scalar values

```
a <- array(1:30, dim=c(2,3,5))
```

- **List**: Any combination of the above

```
obj <- list(a="first", b="second")
```

- **Data frame**: A special list containing **variables of different types**

Basic Types of Data

- **numeric** : Numeric data (approximations of the real numbers, \mathbb{R}) - 수치형
- **integer** : Integer data (whole number, \mathbb{Z}) - 정수형
- **factor** : Categorical data (simple classifications) - 인자형
- **ordered** : Ordinal data (ordered classifications) – 순서형
- **character** : Character data (strings) – 문자형
- **logical** : TRUE or FALSE - 논리형

- 변수의 클래스 확인: `class(object)` # class 함수 사용

Vector

Vector



```
> 1:3
```

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

```
> c(1,2,3)
```

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

```
> seq(from=1, to=3, by=1)
```

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

```
> rep(1:3, times=2)
```

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

```
> rep(1:3, each=2)
```

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

```
> rep(1:3, length.out=8)
```

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

Vector (cont'd)

```
> x <- -1:3
```

```
> x
```

```
> x+3
```

```
> x + sqrt(4)
```

```
> x[1:2]
```

```
> x[-3]
```

Null and NA

- NULL is a object whose values are undefined yet
- NA is a logical constant of length 1 which contains a missing value.

```
c(1, NA, NULL)
```

```
[1] 1 NA
```

```
list(1, NA, NULL)
```

```
[[1]]
```

```
[1] 1
```

```
[[2]]
```

```
[1] NA
```

```
[[3]]
```

```
NULL
```

Vector (cont'd)

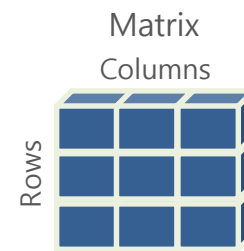
```
> x[x>2]
```

```
> subset(x, x>2)
```

```
> length(x)
```

```
> sign(x)
```

```
> abs(x)
```

Matrix

- 동일한 class의 자료 값으로 구성된 2차원 자료구조
 - 행(rows)과 열(columns)로 구성됨
 - 각 cell은 대괄호 안에 콤마로 구분하여 표현함 – matrix_object[행, 열] ex) X[1,3], X[2,2]...

```
m <- matrix(data=1:8, nrow=2)
```

```
m <- matrix(1:10, nrow=2, byrow=TRUE)
```

```
dim(m)          # 행렬의 dimension
```

```
t(m)            # 전치행렬, 행과 열을 바꾸기
```

```
a <- m %*% t(m);a  # 행렬 내적 (행렬곱)
```

```
m * m           # m의 각 원소들의 제곱 (원소곱)
```

```
apply(m, 1, mean) # 각 행의 평균
```

```
apply(m, 2, var)  # 각 열의 분산
```

Matrix (cont'd)

```
m <- matrix(1:10, nrow=2, byrow=TRUE)
```

```
m[1,3]      # 1행, 3열의 원소
```

```
m[,1]       # 1열 전체의 원소들
```

```
m[2,]       # 2행 전체의 원소들
```

```
rownames(m) <- c("x", "Y")          # 행 이름 정하기
```

```
colnames(m) <- paste0("X", 1:5, sep="") # 열 이름 정하기
```



Data.frame

- 변수와 관측값으로 구성된 2차원 자료구조
 - Matrix의 형태를 가지고 있으나 서로 다른 class를 가진 vector들로 구성됨
 - 또한 list에서 동일한 dimension을 가진 서로 다른 class로도 활용됨
 - Matrix와 동일하게 indexing 하여 각 cell에 접근함
 - 각 행은 observation으로 각 열은 variable로 활용됨
 - R의 패키지 내 대부분의 통계분석 함수들은 data.frame 형태로 자료를 받아들임

```
D1 <- data.frame(Age=c(31,27,39,24), Sex=factor(c(1,1,2,2)))
```

```
class(D1)
```

```
class(D1$Sex)
```

```
str(D1)
```

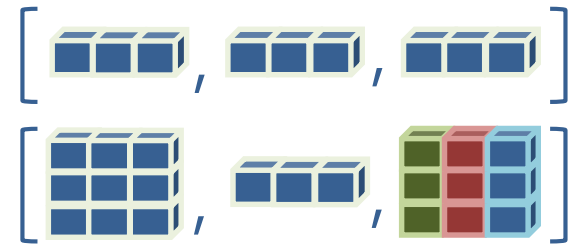
```
D1[,2]
```

```
D1$Sex
```

```
D1[[2]]
```

Lists

List



- 변수들의 묶음 형태로 표현되는 자료구조
 - 하나의 object는 여러 object들을 components 형태로 구성
 - 각 component는 서로 다른 class와 서로 다른 size로 구성해도 무방
 - 각 component에 있는 elements는 `[[index]]` 형태로 접근 가능
 - R의 패키지 내 함수들은 대부분 list 형태로 출력됨

```
L1 <- list(id=c(TRUE,FALSE), x=c(1,2,3), y=matrix(c("A","B","C","D"), nrow=2))
```

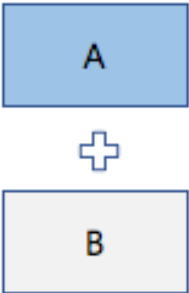

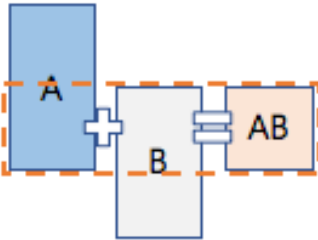
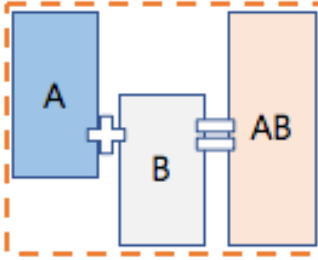
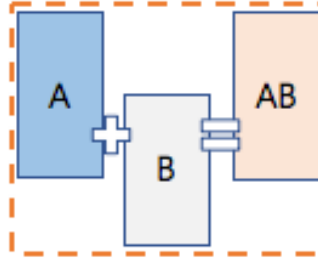
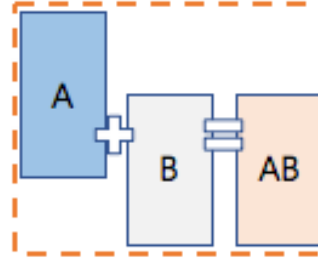
```
L1[c(1,3)]    # 첫 번째와 세 번째 component들만 출력
```

```
L1[3]         # 세 번째 component 만 출력
```

```
L1[[3]]
```

```
L1$y
```

Merging Data Sets

R-function	<u>rbind(A,B)</u>		<u>cbind(A,B)</u>	
Description				
R-function	merge(A,B, by=key column)	merge(A,B, <u>all=TRUE</u> , by=key column)	merge(A,B, <u>all.x=TRUE</u> , by=key column)	merge(A,B, <u>all.y=TRUE</u> , by=key column)
Description				

Merging Data Sets (cont'd)

data.frame vs. cbind

```
(a1 <- cbind(c(1,2,3), c("A", "B", "C")))
```

```
(a2 <- data.frame(list(V1=c(1,2,3), V2=c("A", "B", "C"))))
```

```
class(a1[,1]); class(a2[,1])
```

```
a <- c(1, 2, 3)
```

```
b <- c("a", "b", "c")
```

```
c <- as.factor(c("d", "e", "f"))
```

```
class(a); class(b); class(c)
```

```
cbind(a,b)
```

```
b[3] <- "s"; c[3] <- "s"
```

```
data.frame(a,b,c)
```

Merging Data Sets (cont'd)

```
library(survival)
```

```
data1 <- data.frame(id=as.factor(1:10), cancer[1:10, 1:5])
```

```
data2 <- data.frame(id=as.factor(6:15), cancer[6:15, c(1,2,3,6,7)])
```

```
merge(data1, data2, key="id")
```

```
merge(data1, data2, key="id", all=TRUE)
```

```
merge(data1, data2, key="id", all.x=TRUE)
```

```
merge(data1, data2, key="id", all.y=TRUE)
```

Exploring Data

```
library(survival)

head(cancer)      # head(cancer, 2)

tail(cancer)      # tail(cancer, 10)

# head(data, n=-10) = 뒤의 10명 제외하고 모두 출력


library(FSA); headtail(cancer)

dim(cancer)       # nrow(cancer) & ncol(cancer)

str(cancer)

summary(cancer)

names(cancer)
```


Exploring Data (cont'd)

```
library(survival)

names(cancer)

[1] "inst" "time" "status" "age"

[5] "sex" "ph.ecog" "ph.karno" "pat.karno"

[9] "meal.cal" "wt.loss"

cancer[1:5, ]

cancer[, 1:5]

cancer[, c("inst", "time", "status")]

cancer[, -(1:3)]

subset(cancer, status == 1)

subset(cancer, inst %in% c(3:7))
```

Exploring Data (cont'd)

```
inst
```

```
attach(cancer)
```

```
cancer$inst
```

```
inst
```

```
detach(cancer)
```

```
inst
```

```
rm(inst)          # 'inst' object 제거
```

```
rm(list=ls())      # Work space 내 모든 objects 제거
```

Exploring Data (cont'd)

- “ifelse” function: if-then-else

```
cancer$age.cate <- ifelse(cancer$age > 60, 1, 0)
```

```
cancer$meal.cal.omitNA <- ifelse(cancer[, "meal.cal"] < 0, NA, cancer[, "meal.cal"])
```

- “which” function: indicates of a logical object

```
which(cancer[, "meal.cal"] > 1000)
```

```
which.min(cancer[, "meal.cal"]); which.max(cancer[, "meal.cal"])
```

Exploring Data (cont'd)

```
summary(cancer)
```

```
describe(cancer) # library(Hmisc)
```

```
aggregate(ph.karno ~ sex, data=cancer, mean)
```

```
aggregate(ph.karno ~ sex, data=cancer, sd)
```

```
table(cancer$sex)
```

```
table(cancer$status, cancer$sex) ## = xtabs( ~ status + sex, data=cancer)
```

```
prop.table(table(cancer$sex))
```

```
prop.table(table(cancer$status, cancer$sex))
```

```
prop.table(table(cancer$status, cancer$sex), 1) ## by row
```

```
prop.table(table(cancer$status, cancer$sex), 2) ## by column
```

4. R Graphics

Main Functions







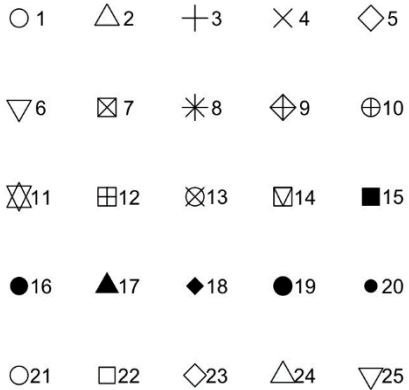
- **plot**: general x-y plotting
- **barplot**: bar plots
- **boxplot**: box-and-whisker plot
- **hist**: histograms
- **pie**: pie charts
- **dotchart**: cleveland dot plots
- **image, heatmap, contour, persp**: functions to generate image-like plots
- **qqnorm, qqline, qqplot**: distribution comparison plots
- **pairs, coplot**: display of multi-variant data

Sub Functions

- **points()** Add points to a figure
- **lines()** Add lines to a figure
- **text()** Insert text in the plot region
- **mtext()** Insert text in the figure and outer margins
- **title()** Add figure title or outer title
- **legend()** Insert legend
- **axis()** Customize axes
- **box()** Add horizontal and vertical lines or a single line

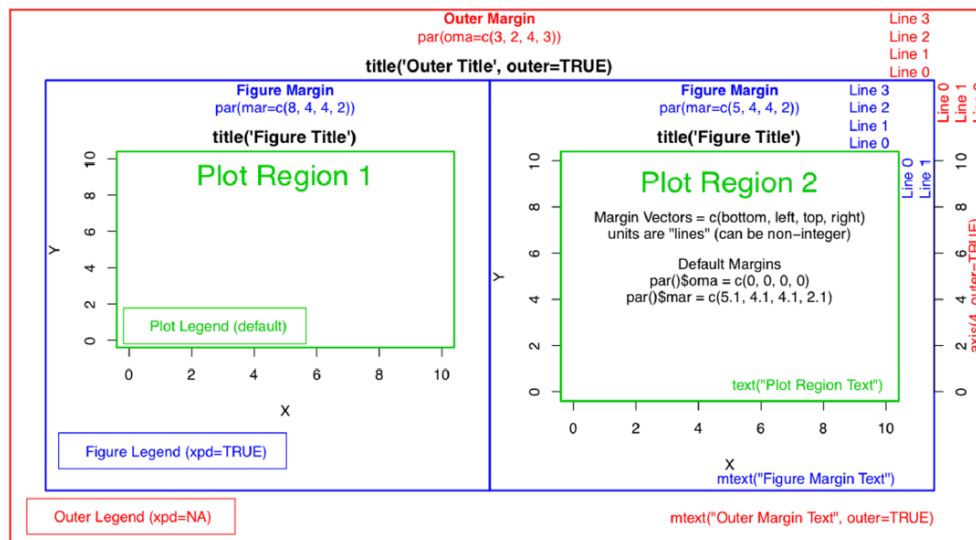
Plot Options

- xlim, ylim : 예) xlim=c(1, 100) # x축의 범위는 1부터 100까지만 표현
- main, xlab, ylab : 그래프 이름, 각 축의 이름
- cex : 크기 지정; cex.main : title의 글자 크기 지정; cex.lab : 축의 크기 지정
- col : 색 지정
- pch : 점 모양 지정
- type : 점 연결 방식
- lty: line의 종류
- lwd : line 굵기

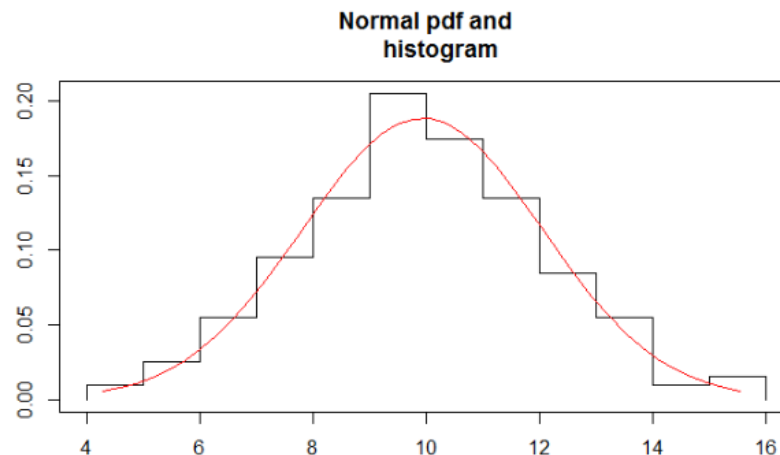
lty		lty + lwd	
0. 'blank'		lty = 6, lwd = 6	
1. 'solid'		lty = 5, lwd = 5	
2. 'dashed'		lty = 4, lwd = 4	
3. 'dotted'		lty = 3, lwd = 3	
4. 'dotdash'		lty = 2, lwd = 2	
5. 'longdash'		lty = 1, lwd = 1	
6. 'twodash'			
type		pch	
<ul style="list-style-type: none"> • "p" for points, • "l" for lines, • "b" for both, • "c" for the lines part alone of "b", • "o" for both 'overplotted', • "h" for 'histogram' like (or 'high-density') vertical lines, • "s" for stair steps, • "S" for other steps, see 'Details' below, • "n" for no plotting. 			

Multiple Plots

한 화면에 여러 plots을 그림
> par(mfrow=c(# of 행, # of 열))



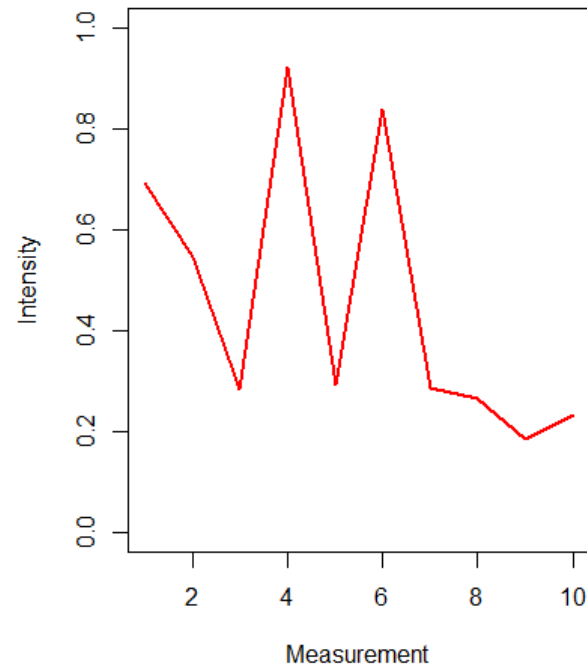
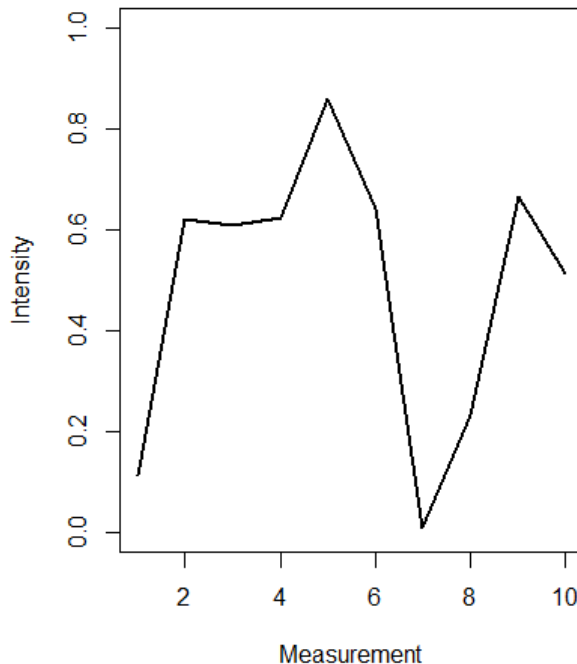
하나의 plot에 여러 그림을 추가
> par(new=TRUE)



Line Plot: multiple lines – par()

mar는 내부 / oma는 외부 Plot을 1-by-2로 지정
op <- par(mar=c(5,4,5,2), mfrow=c(1,2))
Margin size=c(bottom,left,top,right)
plot(y[,1], ylim=c(0,1), xlab="Measurement", ylab="Intensity", type="l", lwd=2, col=1)
plot(y[,2], ylim=c(0,1), xlab="Measurement", ylab="Intensity", type="l", lwd=2, col=2)
title("Multiple Lines", outer=TRUE, line=-3)
par(op)
par() 함수 끝내기 – 이렇게 정의하지 않으면 계속 1-by-2 형태로 plot이 입력됨

Multiple Lines



Line Plot: multiple lines – split.screen()

```
split.screen(c(1,1))
```

```
plot(y[,1], ylim=c(0,1), xlab="Measurement", ylab="Intensity", type="l", lwd=2, col=1)
```

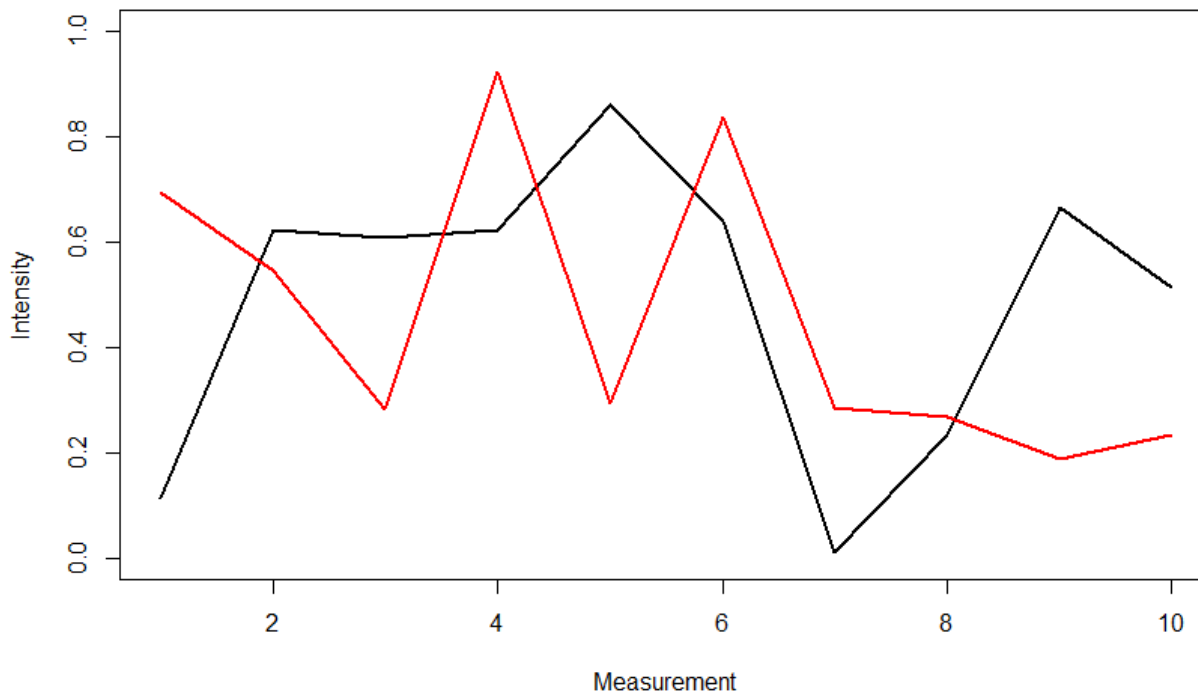
```
screen(1, new=FALSE)
```

type을 line으로 지정

새로운 screen 생성은 FALSE로 지정

```
plot(y[,2], ylim=c(0,1), type="l", lwd=2, col=2, xaxt="n", yaxt="n", ylab="", xlab="", main="", bty="n")
```

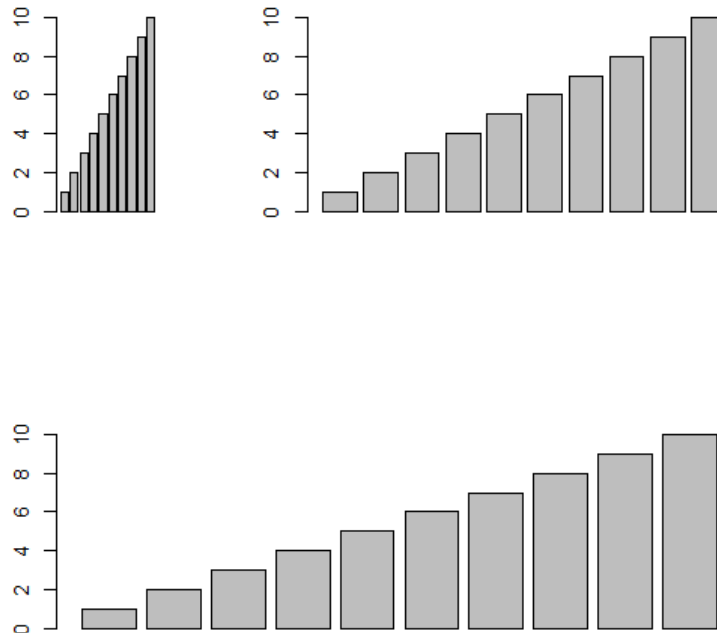
축, plot 제목 등은 모두 지우라는 의미 ("n")



Line Plot: multiple lines – layout()

```
nf <- layout(matrix(c(1,2,3,3), 2, 2, byrow=TRUE), c(3,7), c(5,5), respect=TRUE)  
for(i in 1:3){barplot(1:10)}
```

그림 들어갈 번호 지정, 2-by-2 frame 지정
1행 2행
각 행에 들어갈 plot size 지정

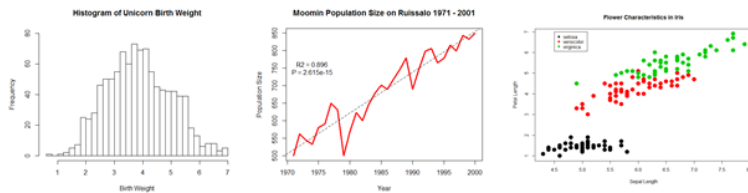


Plot vs. ggplot2

Using 'graphics' package

<https://rpubs.com/williamsurles/294096>

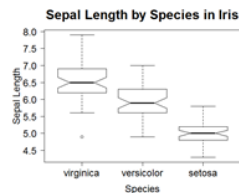
➤ graphics::plot - Use default package



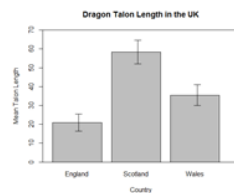
1. Basic Histogram

2. Line Graph with Regression

3. Scatterplot with Legend



4. Boxplot with reordered/formatted axes

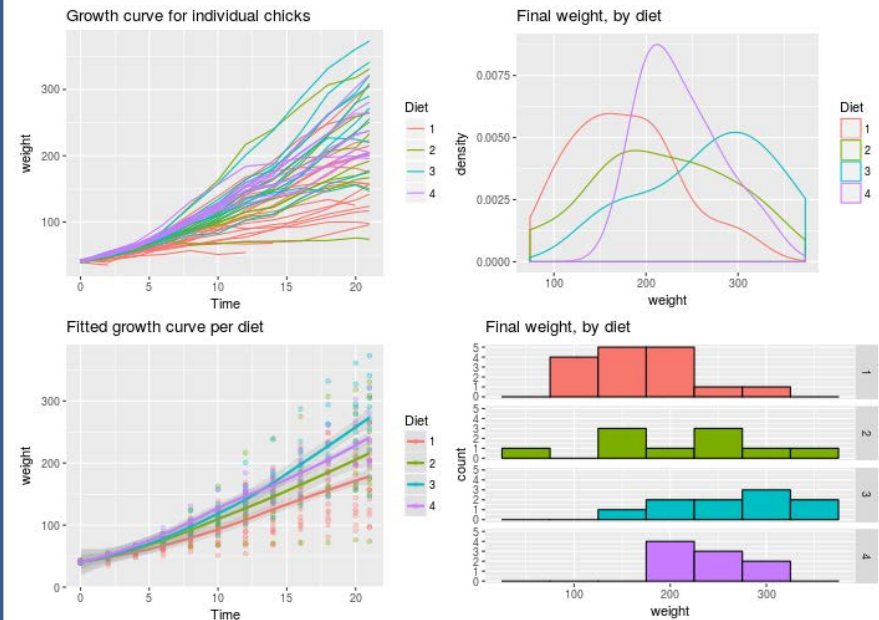


5. Boxplot with Error Bars

Using 'ggplot2' package

<https://rpubs.com/kimwoohyung/ggplot2>

➤ Use {ggplot2} package



ggplot2

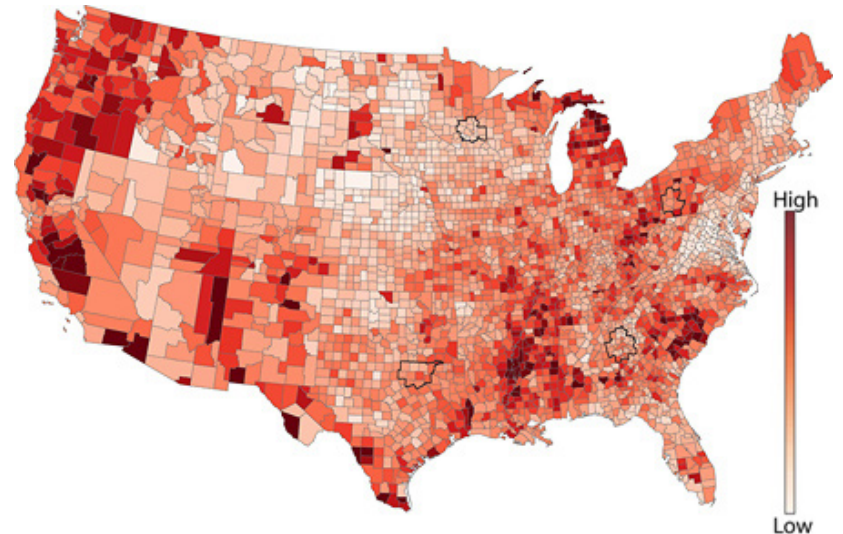
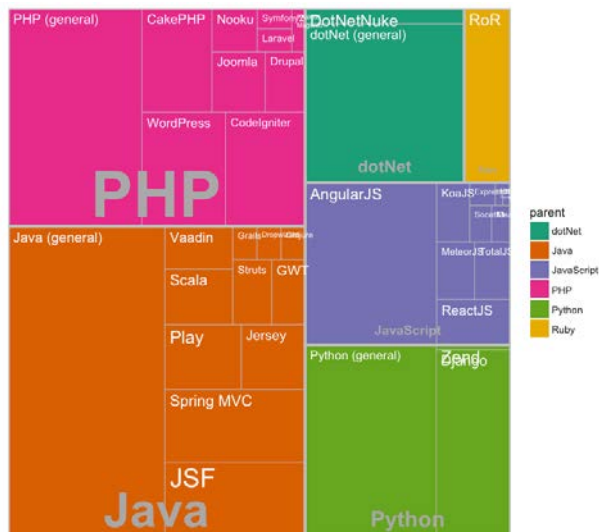
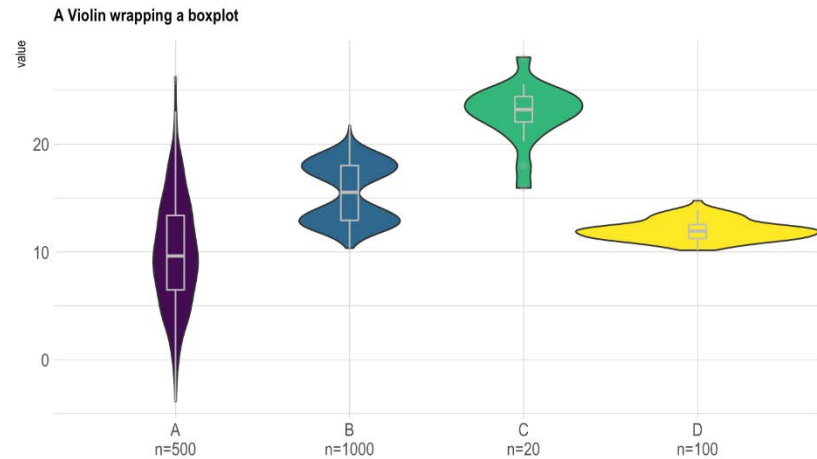
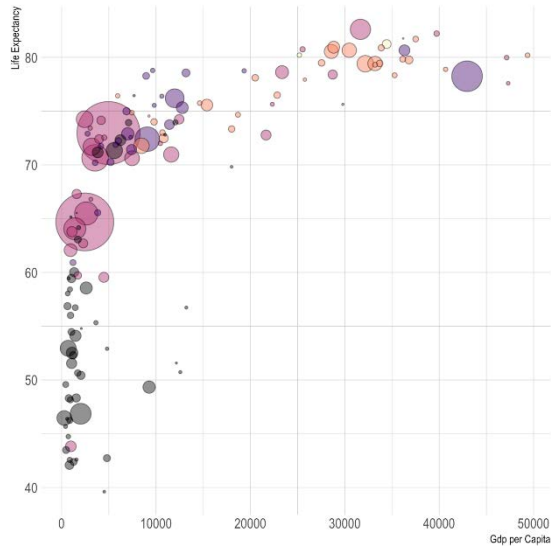
- **ggplot2 특징**

- High-level graphics system
- Flexible, colorful, and dynamic graphics package
- 그래픽 문법(grammar of graphics)에 기반한 그래프(plot)
 - ✓ 그래프를 데이터와 좌표 체계 등과 동일한 방식으로 작성하고자 함
- 기본 좌표에 점, 선, 축, legend 등의 요소들을 추가하는 형식
- 좌표에 geometric objects (e.g. points, lines, bars etc)를 추가하려면 'geoms_'에 해당 함수들을 추가한 후 '+' 값들로 연결시켜서 그래프 작성
 - ✓ 예) `ggplot(data, aes(...)) + geom_point() + geom_*() + theme() + stat_*`

ggplot2: 'theme()'

```
theme(  
  line,  
  rect,  
  text,  
  title,  
  aspect.ratio,  
  axis.title,  
  axis.title.x,  
  axis.title.x.top,  
  axis.title.x.bottom,  
  axis.title.y,  
  axis.title.y.left,  
  axis.title.y.right,  
  axis.text,  
  axis.text.x,  
  axis.text.x.top,  
  axis.text.x.bottom,  
  axis.text.y,  
  axis.text.y.left,  
  axis.text.y.right,  
  axis.ticks,  
  axis.ticks.x,  
  axis.ticks.x.top,  
  axis.ticks.x.bottom,  
  axis.ticks.y,  
  axis.ticks.y.left,  
  axis.ticks.y.right,  
  axis.ticks.length,  
  axis.ticks.length.x,  
  axis.ticks.length.x.top,  
  axis.ticks.length.x.bottom,  
  axis.ticks.length.y,  
  axis.ticks.length.y.left,  
  axis.ticks.length.y.right,  
  axis.line,  
  axis.line.x,  
  axis.line.x.top,  
  axis.line.x.bottom,  
  axis.line.y,  
  axis.line.y.left,  
  axis.line.y.right,  
  legend.background,  
  legend.margin,  
  legend.spacing,  
  legend.spacing.x,  
  legend.spacing.y,  
  legend.key,  
  legend.key.size,  
  legend.key.height,  
  legend.key.width,  
  legend.text,  
  legend.text.align,  
  legend.title,  
  legend.title.align,  
  legend.position,  
  legend.direction,  
  legend.justification,  
  legend.box,  
  legend.box.just,  
  legend.box.margin,  
  legend.box.background,  
  legend.box.spacing,  
  panel.background,  
  panel.border,  
  panel.spacing,  
  panel.spacing.x,  
  panel.spacing.y,  
  panel.grid,  
  panel.grid.major,  
  panel.grid.minor,  
  panel.grid.major.x,  
  panel.grid.major.y,  
  panel.grid.minor.x,  
  panel.grid.minor.y,  
  panel.ontop,  
  plot.background,  
  plot.title,  
  plot.title.position,  
  plot.subtitle,  
  plot.caption,  
  plot.caption.position,  
  plot.tag,  
  plot.tag.position,  
  plot.margin,  
  strip.background,  
  strip.background.x,  
  strip.background.y,  
  strip.placement,  
  strip.text,  
  strip.text.x,  
  strip.text.y,  
  strip.switch.pad.grid,  
  strip.switch.pad.wrap,  
  ...,  
  complete = FALSE,  
  validate = TRUE  
)
```

ggplot2: Examples



Save Plots

Save Plots

```
> jpeg("path/myplot.jpg")  
  
# png()  
  
# bmp()  
  
# pdf()  
  
> plot()  
  
Or  
  
> print(ggplot objects)  
  
> dev.off() # Stop graphics device
```

Thank you!
