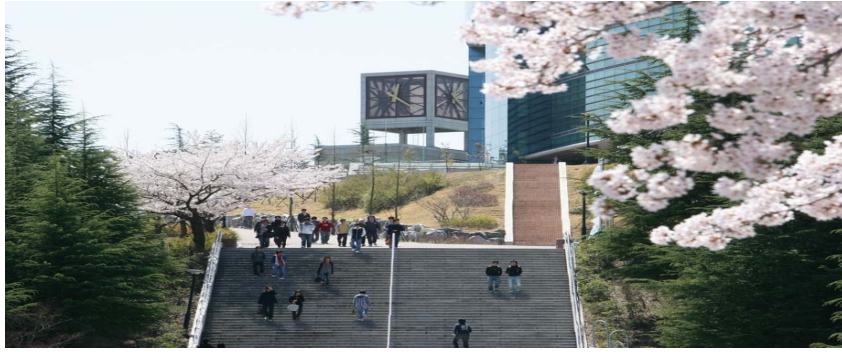


# 1. R의 기초와 기본스크립트 (RStudio설치)



이혜선

POSTECH 산업경영공학과

email: hyelee@postech.ac.kr

web: <http://www.postech.ac.kr/~hyelee/>

	단위별 학습내용 (Week1)
wk1-1	R 소개 및 설치
wk1-2	RStudio 레이아웃과 실행
wk1-3	기본스크립트와 함수
wk1-4	R 추가패키지 사용법

## Wk1-1 : R소개 및 설치

## R 프로그램 소개

### • 오픈소스 통계분석 프로그램

→ 어디서나, 누구나 사용할수 있는 프로그램

### • 빅데이터 분석 도구 – 다양한 통계기법과 시각화 도구 제공

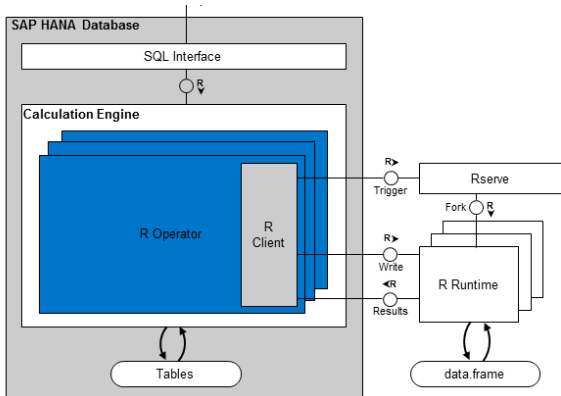
통계기법, 데이터마이닝, 기계학습, 텍스트마이닝, 인공지능 분석툴

Ggplot을 이용한 시각적분석, 공간분석맵

### • 지금까지 15,000여개 이상의 패키지 업로드 제공

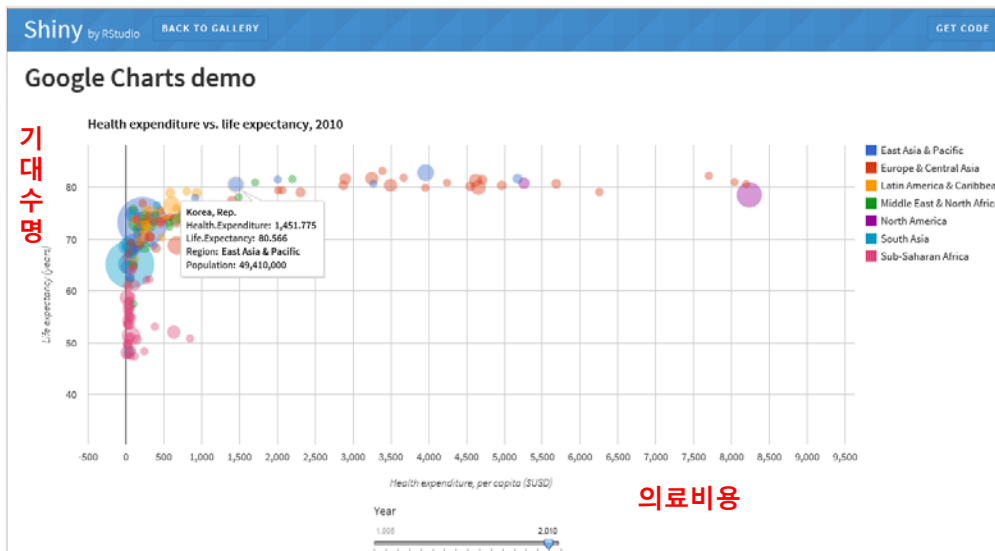
### • R은 C++, 자바, 파이썬 등 다른 프로그램과 쉽게 연동

- R의 개발 : R 프로그램은 1995년에 S언어를 개선하여 [Ross Ihaka and Robert Gentleman](#) 가 개발
- 전세계에서 (학계, 연구기관) 범용적으로 사용
- 기업체에서도 R을 탑재엔진으로 한 기업형분석 플랫폼개발



<https://developers.sap.com/tutorials/mlb-hxe-setup-r.html>

- R을 이용한 구글차트 분석맵 (세계의 기대수명과 의료비용, 1995-2011)



<http://shiny.RStudio.com/gallery/google-charts.html>

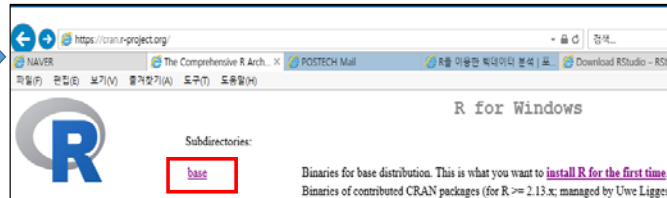
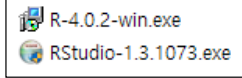
# 1. R 프로그램 설치

1-1. R의 소개 및 설치

## • R 프로그램 : Base 프로그램의 설치 Step1

<https://cran.r-project.org/>

R\_Rstudio\_zip

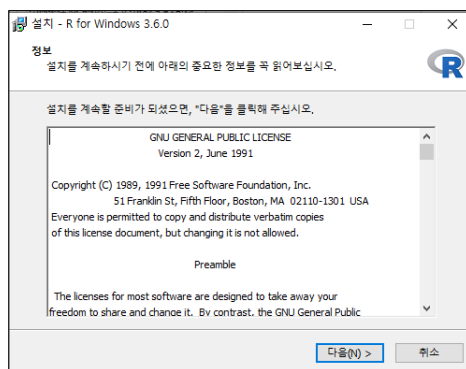


[Download R 4.0.2 for Windows](#) (84 megabytes, 32/64 bit)  
[Installation and other instructions](#)  
[New features in this version](#)

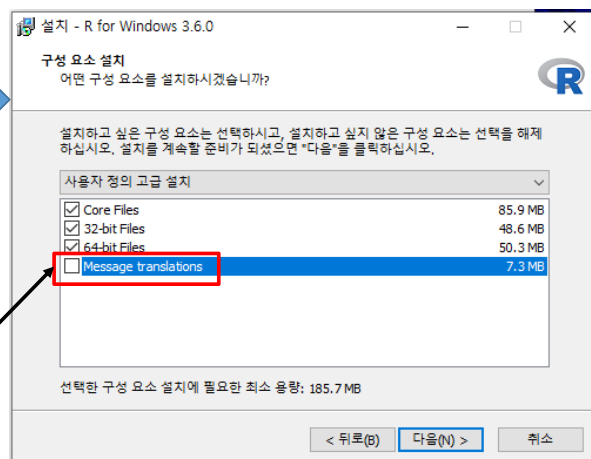
\* R Base program의 버전은 계속 update됨

# 1. R 프로그램 설치

1-1. R의 소개 및 설치



여기서 ☒표를 제거하면 영문메뉴로 프로그램이 설치됩니다.  
(파일메뉴 및 로그 메시지를 영문으로 사용할 것을 권장!!)



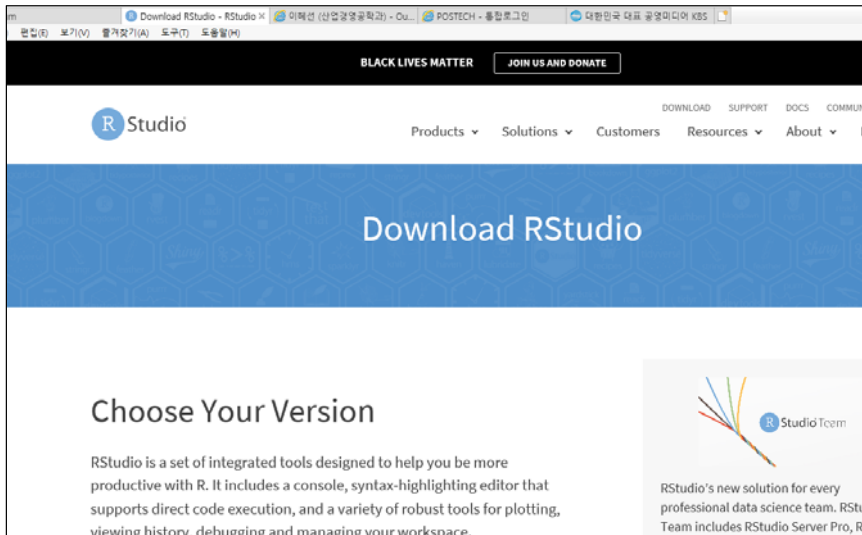
## 2. RStudio 설치

1-1. R의 소개 및 설치

- RStudio : R 프로그램 실행을 지원하는 통합툴 프로그램 (프로그램 입력, 작업정보관리, 그래픽, 패키지 윈도우)

**Step2**

<https://www.RStudio.com/products/RStudio/download/>



## 2. RStudio 설치

1-1. R의 소개 및 설치

### • Rstudio 데스크탑 무료버전 설치

	RStudio Desktop Open Source License <b>Free</b>	RStudio Desktop Commercial License <b>\$995</b> /year	RStudio Server Open Source License <b>Free</b>
	<b>DOWNLOAD</b> <a href="#">Learn more</a>	<b>BUY</b> <a href="#">Learn more</a>	<b>DOWNLOAD</b> <a href="#">Learn more</a>
Integrated Tools for R	✓	✓	✓
Priority Support		✓	
Access via Web Browser			✓
Enterprise Security			
Project Sharing			

RStudio Desktop 1.3.1073 - [Release Notes](#)

1. Install R. RStudio requires R 3.0.1+. **Step1**
2. Download RStudio Desktop. Recommended for your system:

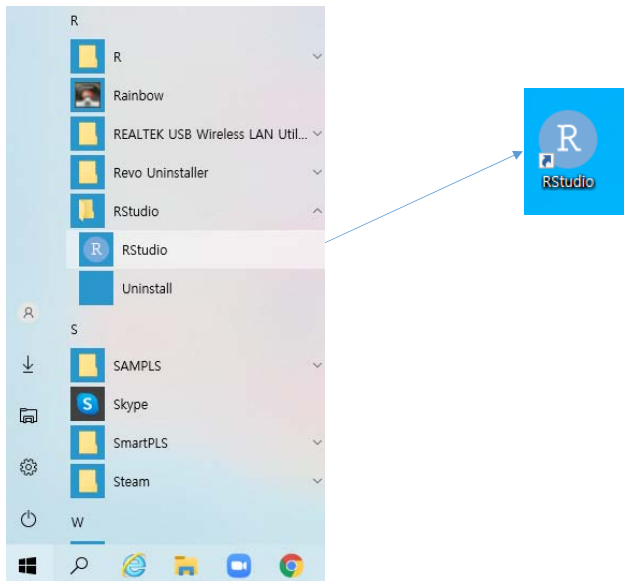
**DOWNLOAD RSTUDIO FOR WINDOWS**  
1.3.1073 | 171.62MB **Step2**

Requires Windows 10/8/7 (64-bit)

## 2. RStudio 설치

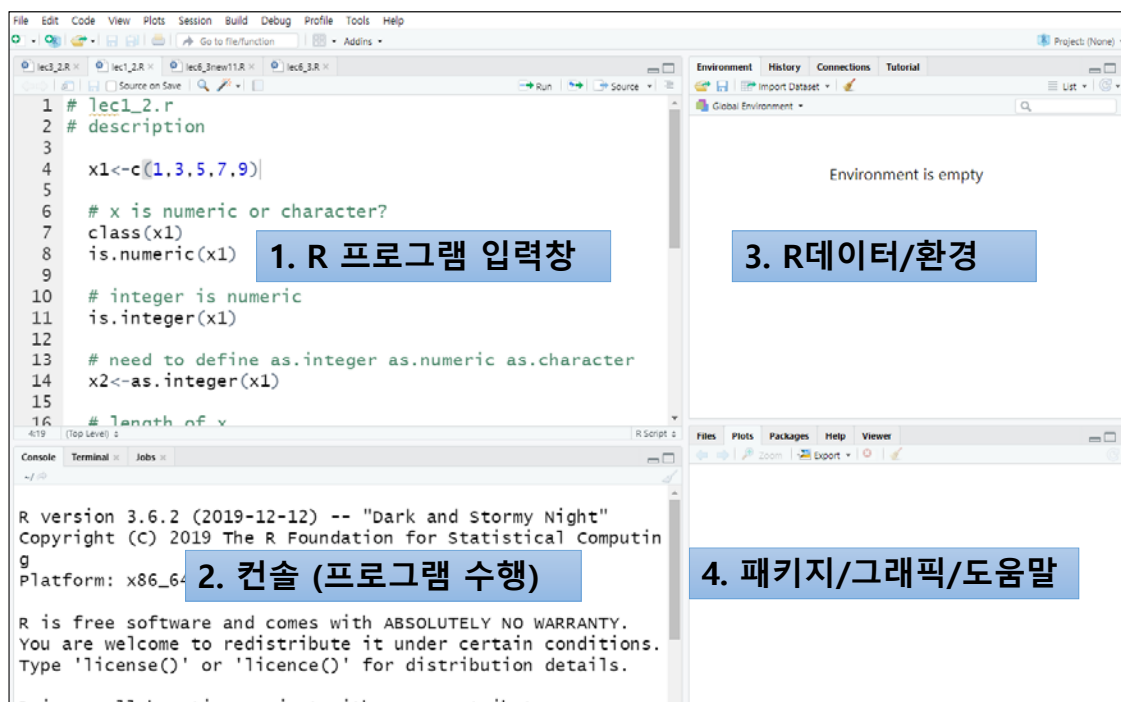
1-1. R의 소개 및 설치

### • RStudio 설치 후 실행 아이콘



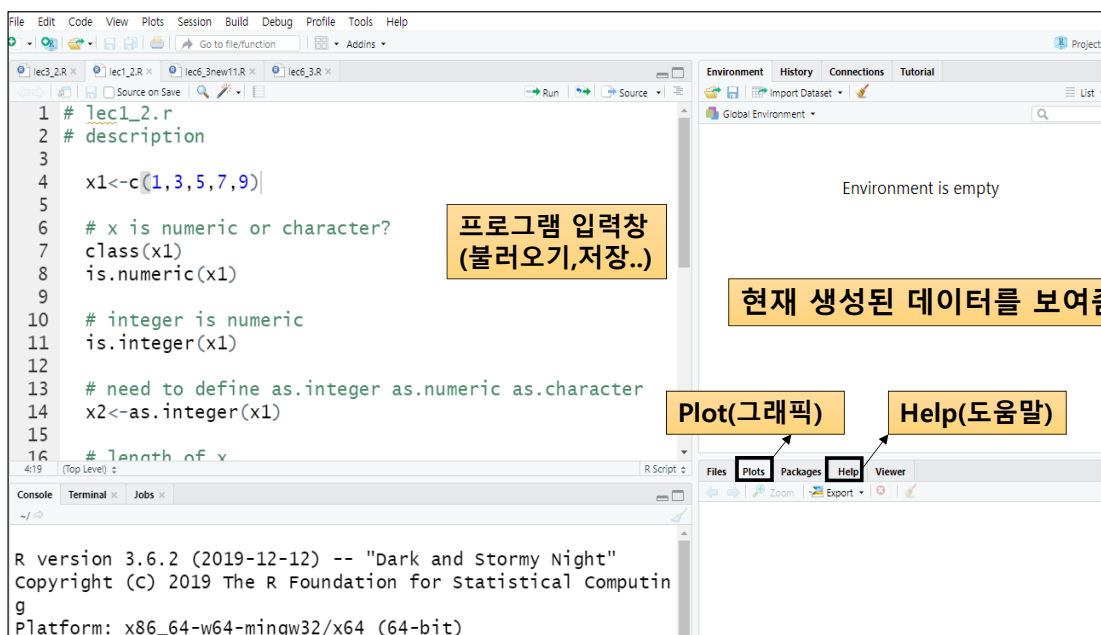
## RStudio 화면구성

1-1. R의 소개 및 설치

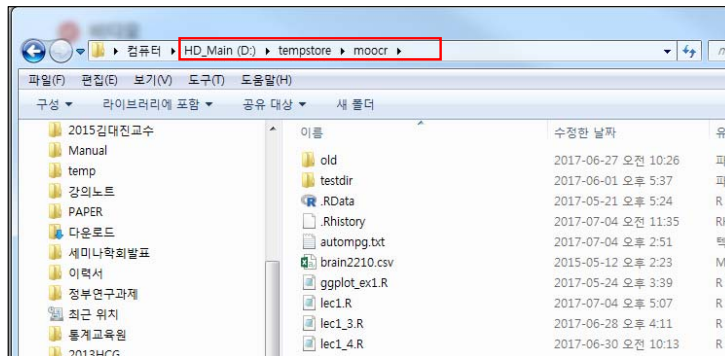


## Wk1-2 : RStudio 레이아웃과 실행

## RStudio의 레이아웃

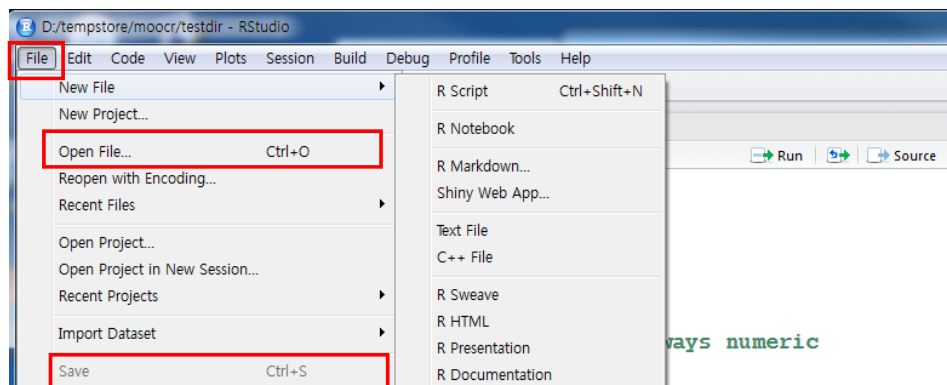


- \*.r (R 프로그램)을 지정한 폴더에 저장



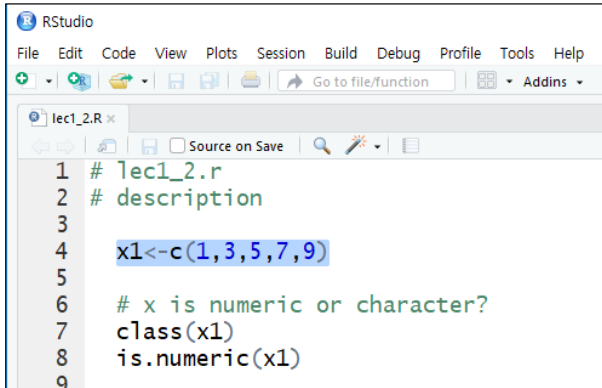
## R 프로그램 열기/저장

- R프로그램 열기/저장
  - File\_Open File (lec1\_2.r을 열기)
  - File\_Save





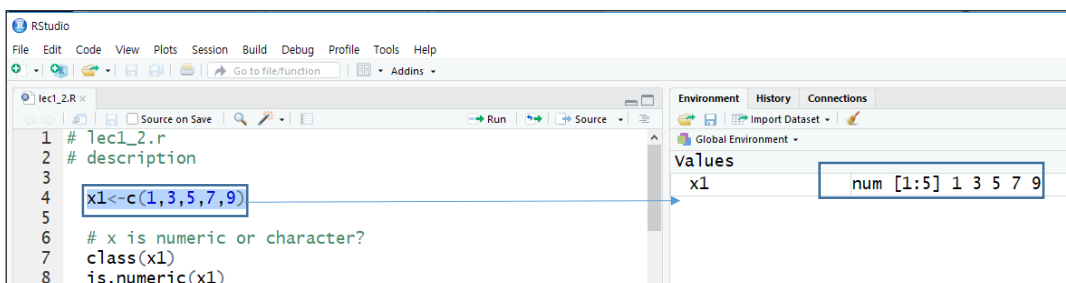
- R프로그램을 수행하는 단축키 : (Ctrl&Enter)
- 수행하고자 하는 프로그램 부분을 선택하고 수행
- 예제 : 첫줄을 선택하고 수행=> (1,3,5,7,9)을 가진 벡터 x1 생성



```

1 # lec1_2.r
2 # description
3
4 x1<-c(1,3,5,7,9)
5
6 # x is numeric or character?
7 class(x1)
8 is.numeric(x1)
9
    
```

- lec1\_2.r의 첫째줄을 수행하면, 환경창에 x1 데이터가 생성됨을 보여줌



## • 프로그램 코드설명 (lec1\_2.r)

```
# lec1_2.r
# description
x1<-c(1,3,5,7,9)
# x is numeric or character?
class(x1)
is.numeric(x1)
# integer is numeric
is.integer(x1)
# need to define as.integer as.numeric as.character
x2<-as.integer(x1)
# length of x
length(x1)
# x is a vector?
is.vector(x1)
# class - character
"I like apple"
class("I like apple")
```

# 으로 시작하는 줄은 프로그램에 대한 설명  
(자동적으로 초록색 폰트로 바뀜)

검정색 폰트의 줄은 R프로그램코드

# R 도움말의 활용

## • help (도움말)의 활용

- Help창을 선택하고 검색란에 'integer'를 치면 어떻게 사용하는지 매뉴얼페이지가 자동적으로 연결됨
- 콘솔창에 help(integer)라고 치면 동일함 (예제 : help(boxplot) )

```
> help(integer)
>
>
>
>
```



## Wk1-3 : 기본스크립트와 함수

## 기본스크립트와 함수

### • 프로그램 열기(File\_Open file\_lec1\_2.r)

#### 프로그램 편집창

```
# lec1_2.r
# description

x1<-c(1,3,5,7,9)

# x is numeric or character?
class(x1)
is.numeric(x1)

# integer is numeric
is.integer(x1)

# need to define as.integer as.numeric
x2<-as.integer(x1)

# length of x
length(x1)

# x is a vector?
is.vector(x1)

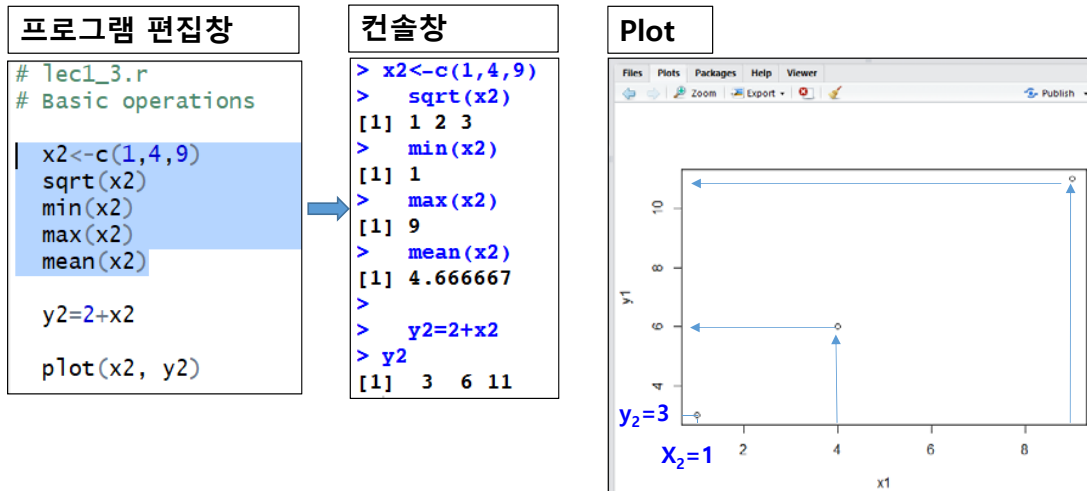
# class - character
"I like apple"
class("I like apple")
```



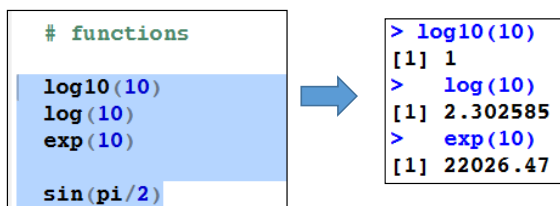
#### 콘솔창

```
> x1<-c(1,3,5,7,9)
> x1
[1] 1 3 5 7 9
> class(x1) <- X의 범주는? (숫자 혹은 문자)
[1] "numeric"
> is.numeric(x1) <- X는 숫자?
[1] TRUE
> length(x1) <- X의 길이는?
[1] 5
> is.vector(x1) <- X는 벡터?
[1] TRUE
```

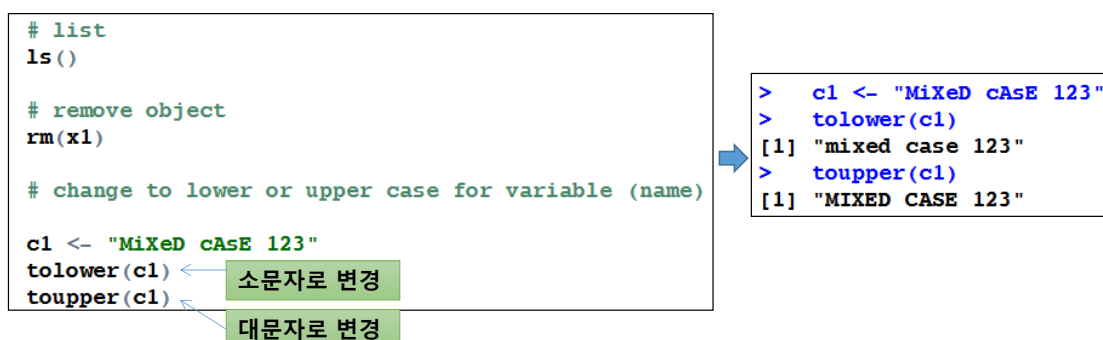
## • 기본연산 (더하기, 곱하기, 제곱근, 최소값, 최대값, 평균)



## • 함수 (로그, 지수, 사인, 코사인)



## • 소문자, 대문자 변경, ls(), rm()



## • 정규분포로부터 데이터생성 함수 : rnorm

```
# generating data from distributions
# 100 values from normal distribution with mean=0, sd=1
x3<-rnorm(100)
head(x3)
mean(x3)
sd(x3)
min(x3)
max(x3)

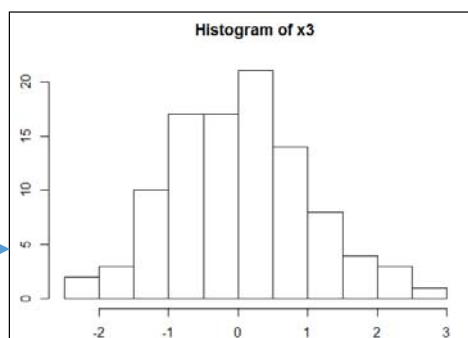
# 10000 values from normal
x4<-rnorm(10000)
mean(x4)
sd(x4)
```

```
X<-rnorm(n, mean, std)
X<-rnorm(100,0,5)
X<-rnorm(100,5,5)
```

```
> x3<-rnorm(100)
> head(x3)
[1] 1.3759941 -0.1978637 0.9209631 1.1059498 -0.8540965
[6] 1.2116593
> mean(x3)
[1] -0.1774837
> sd(x3)
[1] 1.076244
> min(x3)
[1] -3.345342
> max(x3)
[1] 1.880073
>
> # 10000 values from normal distribution with mean=0, sd=1
> x4<-rnorm(10000)
> mean(x4)
[1] 0.009022544
> sd(x4)
[1] 1.00527
```

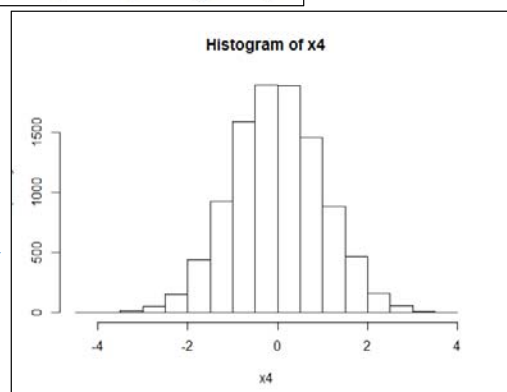
# 기본스크립트와 함수

```
x3<-rnorm(100)
# head(x3)
mean(x3)
sd(x3)
min(x3)
max(x3)
hist(x3)
```



Q1 : X3과 X4의 히스토그램을 보면 어떤 데이터가 더 정규분포 형태를 보이나요?

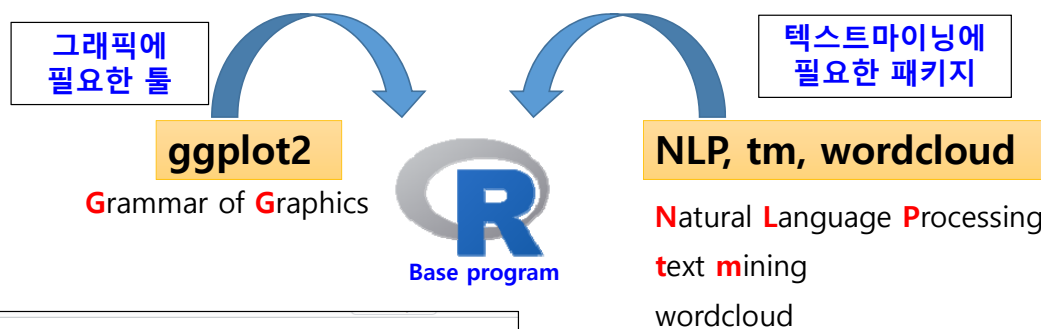
```
# 10000 values from normal
x4<-rnorm(10000)
mean(x4)
sd(x4)
hist(x4)
```



## Wk1-4 : R 추가패키지 사용법

### 1. R 추가패키지 설치

- R프로그램은 기본 program에 추가적으로 필요한 분석 툴을 설치



```
ggplot2-package (ggplot2)                                R Documentation
ggplot2: Create Elegant Data Visualisations
Using the Grammar of Graphics

Description
A system for 'declaratively' creating graphics, based on "The Grammar of Graphics". You
provide the data, tell 'ggplot2' how to map variables to aesthetics, what graphical primitives
to use, and it takes care of the details.

Author(s)
Maintainer: Hadley Wickham hadley@rstudio.com
Authors:
  • Winston Chang winston@rstudio.com
```

# 1. R 추가패키지 설치

1-4. R 추가패키지 사용법

## • 추가패키지 설치 (install.packages)

프로그램 편집창

```
# lec1_4.r : Install package
# install package
install.packages("ggplot2")
library(ggplot2)
help(ggplot2)

# install scatterplot3d
install.packages("scatterplot3d")
library(scatterplot3d)
```

1  
ggplot2라는 패키지를 설치

2  
scatterplot3d 패키지 설치

컨솔창

```
https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/hyeseon/Documents/R/win-librar
y/3.6'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/ggp
lot2_3.1.1.zip'
Content type 'application/zip' length 3638378 bytes (3.5 MB)
downloaded 3.5 MB

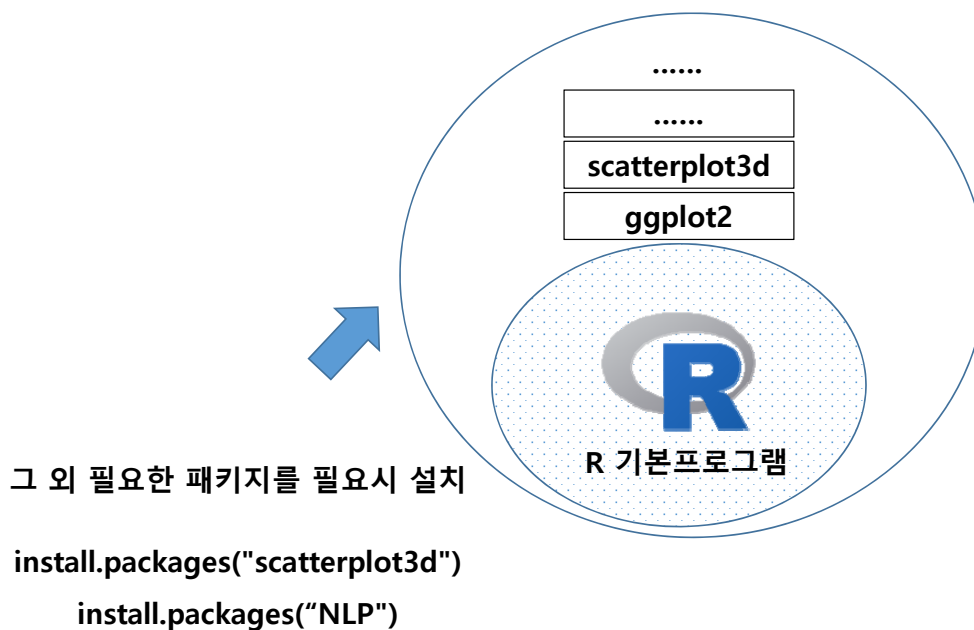
package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:\Users\hyeseon\AppData\Local\Temp\Rtmp062V2c\downloade
d_packages
```

\* 해당패키지를 사용할때 library(scatterplot3d)는 사용할때마다 필요함 !!! (설치는 한번만 함)

# 2. R 기본프로그램과 추가패키지

1-4. R 추가패키지 사용법



### 3. R 추가패키지 사용 예제

1-4. R 추가패키지 사용법

#### • 3d plot을 그리고자 할때 : "scatterplot3d" 패키지

```
# lec1_4.r : Install package

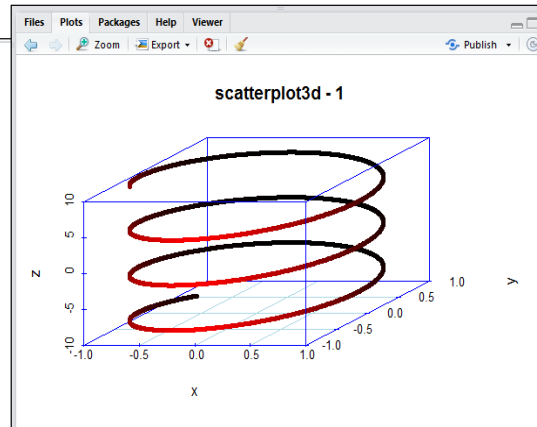
# install package

install.packages("ggplot2")
library(ggplot2)
help(ggplot2)

# install scatterplot3d

install.packages("scatterplot3d")
library(scatterplot3d)
help(scatterplot3d)

# example program using scatterplot3d
z <- seq(-10, 10, 0.01)
x <- cos(z)
y <- sin(z)
scatterplot3d(x, y, z, highlight.3d=TRUE, col.axis="blue",
              col.grid="lightblue", main="scatterplot3d - 1", pch=20)
```

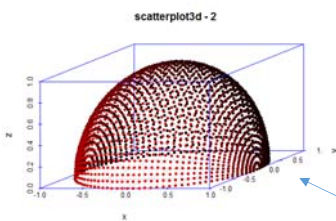


### 3. R 추가패키지 사용 예제

1-4. R 추가패키지 사용법

#### • help매뉴얼의 예제 프로그램 사용

`help(scatterplot3d)`



```
Files Plots Packages Help Viewer

R: 3D Scatter Plot - Find in Topic

persp, plot, par.

Examples

## On some devices not all colors can be displayed.
## Try the postscript device or use highlight.3d = FALSE.

## example 1
z <- seq(-10, 10, 0.01)
x <- cos(z)
y <- sin(z)
scatterplot3d(x, y, z, highlight.3d=TRUE, col.axis="blue",
              col.grid="lightblue", main="scatterplot3d - 1", pch=20)

## example 2
temp <- seq(-pi, 0, length = 50)
x <- c(rep(1, 50) %*% t(cos(temp)))
y <- c(cos(temp) %*% t(sin(temp)))
z <- c(sin(temp) %*% t(sin(temp)))
scatterplot3d(x, y, z, highlight.3d=TRUE,
              col.axis="blue", col.grid="lightblue",
              main="scatterplot3d - 2", pch=20)
```

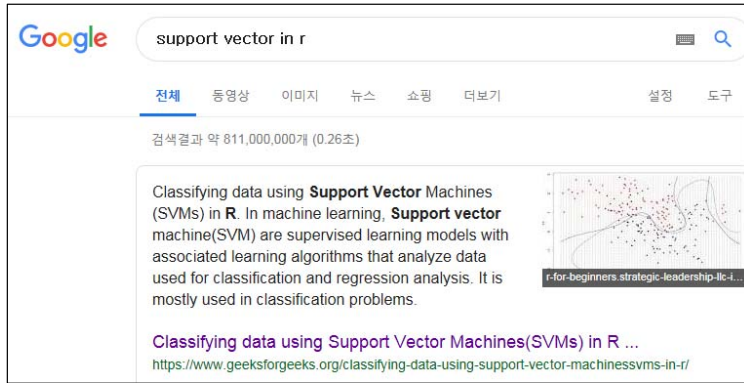


## 4. R 추가패키지 검색

1-4. R 추가패키지 사용법

### • 웹에서 검색 (패키지 이름을 모를때)

예제 1 : Google에서 "support vector in r" 로 찾을수 있음.  
서포트벡터머신(support vector machine)을 R에서 수행하는 방법을 검색할수 있음.



1

서포트벡터머신 수행  
을 위해 추가패키지  
"e1071"정보가 나옴

## 4. R 추가패키지 검색

1-4. R 추가패키지 사용법

서포트벡터머신 (support vector machine)  
수행을 위한 패키지 "e1071"설치

```
# to find out the package in google  
# example 1 : support vector machine method  
  
# step1 : install package  
install.packages('e1071')  
library(e1071)
```

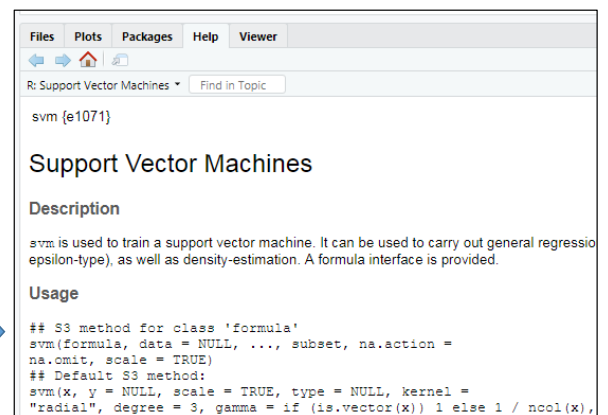
2

웹탐색에서 아래와 같은 설명을 보고 "svm"함수에 대해 help 메뉴를  
사용하여 상세한 설명을 볼수 있다

#### • Fitting SVM to the training set

```
# Fitting SVM to the Training set  
install.packages('e1071')  
library(e1071)  
  
classifier = svm(formula = Purchased ~ .,  
                 data = training_set,  
                 type = 'C-classification',  
                 kernel = 'linear')
```

```
# step2  
help("mean")  
help("svm")
```



	단위별 학습내용 (Week2)
wk2-1	벡터 및 행렬생성
wk2-2	객체이름과 범주형변수
wk2-3	벡터와 행렬의 연산
wk2-4	간단한 함수생성 및 루프(for, while)

## Wk2-1 : 벡터 및 행렬생성

## • 벡터의 생성 (lec2\_1.r)

### 프로그램 편집창

```
# lec2_1.r

# vector
x<-c(1,3,5,7,9)
x[3]

# subset of vector :
x[-1]

# subset of vector :
x1<-x[-c(1,2)]
x1

# subset of vector :
x2<-x[-c(1:3)]
x2
```

### 콘솔창

```
> # vector
> x<-c(1,3,5,7,9)
> x[3]
[1] 5
>
> # subset of vector : delete the first element
> x[-1]
[1] 3 5 7 9
>
> # subset of vector : delete the first two element
> x1<-x[-c(1,2)]
> x1
[1] 5 7 9
> # subset of vector : delete the 1st to the 3rd element
> x2<-x[-c(1:3)]
> x2
[1] 7 9
```

x[-1]은 첫번째 값을 삭제하라는 의미

x[-(1,2)]은 첫번째, 두번째값 삭제

x[-(1:3)]은 첫번째부터 세번째값까지 삭제

## • 벡터 (seq함수 사용)

sequence

```
# create vector using 'seq'
# sequence of 20 values
y1<-seq(0,10, length=20)
# sequence of (1 to 10) by 0.5
y2<-seq(0,10, by=0.5)
```

y1: 0부터 10까지, 20개의 값을 생성

y2: 0부터 10까지 0.5씩 간격을 두고 값을 생성

```
> y1<-seq(0,10, length=20)
> y1
[1] 0.0000000 0.5263158 1.0526316 1.5789474 2.1052632
[6] 2.6315789 3.1578947 3.6842105 4.2105263 4.7368421
[11] 5.2631579 5.7894737 6.3157895 6.8421053 7.3684211
[16] 7.8947368 8.4210526 8.9473684 9.4736842 10.0000000
> y2<-seq(0,10, by=0.5)
> y2
[1] 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
[12] 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0
```

## • 벡터 (rep 함수 사용)

↑  
replication

```
# using rep
z1<-rep(1:4, 2)
z1
```



1부터 4까지 두번을 반복하여 생성하라는 의미

```
> z1<-rep(1:4, 2)
> z1
[1] 1 2 3 4 1 2 3 4
```

```
z2<-rep(1:2,5)
z2
```



1부터 2까지 다섯번을 반복하여 생성하라는 의미

```
> z2<-rep(1:2,5)
> z2
[1] 1 2 1 2 1 2 1 2 1 2
```

## • 생성된 데이터 확인

RStudio interface showing the following code in the script editor:

```
16
17
18 # create vector using 'seq'
19 # sequence of 20 values
20 y1<-seq(0,10, length=20)
21 # sequence of (1 to 10) by 0.5
22 y2<-seq(0,10, by=0.5)
23
24 # using rep
25 z1<-rep(1:4, 2)
26 z1
27
28 z2<-rep(1:2,5)
29 z2
30
```

The Environment pane on the right shows the following values:

Variable	Class	Length	Values
y1	num	[1:20]	0 0.526 1.053 1.579
y2	num	[1:21]	0 0.5 1 1.5 2 2.5 3
z1	int	[1:8]	1 2 3 4 1 2 3 4
z2	int	[1:10]	1 2 1 2 1 2 1 2 1 2

## • 벡터 결합 (행과 열을 기준)

**cbind : column bind (열 기준으로 결합)**

```
# combine vectors in a row or column
c1<-c(2,4,6,8,10)
c2<-cbind(x, c1)
c2
```

(5\*2)인 행렬이 됨

```
> x<-c(1,3,5,7,9)
> c1<-c(2,4,6,8,10)
> c2<-cbind(x, c1)
> c2
```

	x	c1
[1,]	1	2
[2,]	3	4
[3,]	5	6
[4,]	7	8
[5,]	9	10

**rbind : row bind (행으로 결합)**

```
c3<-rbind(x,c1)
c3
```

(2\*5)인 행렬이 됨

```
> c3<-rbind(x,c1)
> c3
```

	[,1]	[,2]	[,3]	[,4]	[,5]
x	1	3	5	7	9
c1	2	4	6	8	10

# 행렬의 생성

## • 행렬의 생성 (matrix 함수 이용) - 행의 수, 열의 수 입력

matrix 함수를 이용하여 1부터 10까지의 숫자로 2행의 행렬을 생성

```
# create matrix
# two row matrix with 1 to 10
m1<-matrix(1:10, nrow=2)
m1
```

```
> m1<-matrix(1:10, nrow=2)
> m1
```

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

matrix 함수를 이용하여 1부터 6까지의 숫자로 3개열의 행렬을 생성,  
1열부터 채우는 것이 default

```
# three columns matrix with 1:6
m2<-matrix(1:6, ncol=3)
m2
```

```
> m2<-matrix(1:6, ncol=3)
> m2
```

	[,1]	[,2]	[,3]
[1,]	1	3	5
[2,]	2	4	6

## • 행렬의 생성 (matrix 함수 이용) - 행의 수, 열의 수 입력

matrix함수를 이용하여 1부터 6까지의 숫자로 2개 행의 행렬을 생성,  
1열부터 채우는것이 default, 여기서는 byrow=T이므로 1행부터 채워서 생성

```
# matrix filled by rows, default: filled by columns
m3<-matrix(1:6, nrow=2, byrow=T)
m3
# help (matrix)|
```

```
> m3<-matrix(1:6, nrow=2, byrow=T)
> m3
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

## • 고차원 행렬 (array를 이용하여 생성)

```
# higher order of array
a1<-array(c(1:18), dim=c(3,3,2))
a1
```

```
> a1<-array(c(1:18), dim=c(3,3,2))
> a1
, , 1
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

, , 2
      [,1] [,2] [,3]
[1,]   10   13   16
[2,]   11   14   17
[3,]   12   15   18
```

```
a1[, ,1]
a1[, ,2]
```

```
> a1[, ,1]
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```



## Wk2-2 : 객체이름과 범주형변수

### 벡터 생성과 이름주기

- 벡터생성 및 벡터이름 주기 (names)

(0,1)값을 갖는 벡터 gender에 0=female, 1=male이라는 값을 부여

```
# lec2_2.r
# Naming vector and matrix, Data frame

# Give name to a vector
gender<-c(0,1)
names(gender)<-c("female", "male")
gender
```

컨솔창

```
> gender<-c(0,1)
> names(gender)<-c("female", "male")
> gender
female   male
      0      1
```

- gender변수는 (0,1)로 입력된 경우 => gender를 factor변수로 정의 필요

gender변수는 factor변수로 인식하지 못함 : (0, 1)로 입력되었으므로

```
# define as a factor variable
is.factor(gender)
```



```
> is.factor(gender)
[1] FALSE
```

- as.factor(변수이름) : 어떤 변수를 factor변수로 정의할 때

gender변수는 factor변수로 정의 => is.factor(gender)로 확인하면  
gender는 factor변수로 정의되어있음을 볼 수 있다

```
gender<-as.factor(gender)
is.factor(gender)
```



```
> is.factor(gender)
[1] TRUE
```

- 범주형변수 생성 (factor 사용)

size라는 변수를 생성 : (S, M, L, XL)의 값을 갖는 범주형변수(factor)를 생성

```
#categorical variables : factor
size<-c("S", "M", "L", "XL")
# define size as a factor (categorical variable)
size_factor<-factor(size)

size_factor
```



```
> size<-c("S", "M", "L", "XL")
> # define size as a factor (categorical variable)
> size_factor<-factor(size)
> size_factor
[1] S M L XL
Levels: L M S XL
```

factor( )는 범주형변수로 정의하는 함수

순서가 없음!

질문 : size\_factor가 범주형변수인가?

답 : size\_factor는 범주형변수이다 (위에서 factor로 정의했음)

```
is.factor(size_factor)
```



```
> is.factor(size_factor)
[1] TRUE
```



- 범주형변수 생성 (factor 사용) – 순서를 정의한 factor생성

size\_factor3 : (S, M, L, XL)의 값을 갖으며, S<M<L<XL의 순서가 정의된 factor !!

```
# give order for categorical variable
size_factor3 <- factor(size, ordered = TRUE,
                       levels = c("S", "M", "L", "XL"))
size_factor3
```

```
> size_factor3
[1] S M L XL
Levels: S < M < L < XL
```

## 행렬의 차원

- 행렬 생성하고 차원 알아보기

### 프로그램 편집창

```
# generate matrix from normal
x<-matrix(rnorm(12),nrow=4)
x
```

정규분포(평균=0, 편차=1)에서 12개의 객체를 임의적으로 뽑아서 4\*3행렬을 생성

### 콘솔창

```
> x<-matrix(rnorm(12),nrow=4)
> x
      [,1]      [,2]      [,3]
[1,]  0.8620431  1.6045946  1.2542515
[2,]  0.3862094  1.4962942 -0.5672157
[3,] -1.2422042  0.8604586 -0.2076393
[4,] -0.5251557  3.4582131  0.6079610
```

### x행렬의 차원 알아보기

```
# check dimension of x
dim(x)
```

```
> dim(x)
[1] 4 3
```

\*rnorm (random sampling from normal distribution)

- 행렬의 속성

행렬 x 는 (4\*3), x가 data frame은 아님

```
# generate matrix from normal
x<-matrix(rnorm(12),nrow=4)
x
# check dimension of x
dim(x)

# data frame
is.data.frame(x)
# matrix x is not data frame
```

- as.data.frame(x)는 x을 데이터로 인식

```
# define x as a data frame
x<-as.data.frame(x)
# then x is a data frame
is.data.frame(x)
```



## Wk2-3 : 벡터와 행렬의 연산

### 기본연산

#### • 기본 연산 기호

Operator	Description
+	addition
-	subtraction
*	multiplication
/	division
<b>^ or **</b>	exponentiation
<b>x %% y</b>	modulus (x mod y) 5%%2 is 1
<b>x %/ y</b>	integer division 5%/2 is 2

Operator	Description
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
!=	not equal to
!x	Not x
<b>x   y</b>	x OR y
<b>x &amp; y</b>	x AND y
<b>isTRUE(x)</b>	test if X is TRUE

### • 더하기, 곱하기, 나누기 (몫, 나머지)

2의 3승 ( $2^3$ ,  $2^{**}3$ )

4의 3승 ( $4^3$ ,  $4^{**}3$ )

%% (7을 5로 나누었을때 나머지)

%% (7을 5로 나누었을때 몫)

```
# calculation
2^3
4**3

7%%5
7%/5
```

➔

```
> 2^3
[1] 8
> 4**3
[1] 64
> 7%%5
[1] 2
> 7%/5
[1] 1
```

# 행렬의 연산

## 2-3. 벡터와 행렬의 연산

### • 행렬의 연산

Operator or Function	Description		
<b>A * B</b>	Element-wise multiplication	<b>y&lt;-svd(A)</b>	Single value decomposition of <b>A</b> . <b>y\$d</b> = vector containing the singular values of <b>A</b> <b>y\$u</b> = matrix with columns contain the left singular vectors of <b>A</b> <b>y\$v</b> = matrix with columns contain the right singular vectors of <b>A</b>
<b>A %*% B</b>	Matrix multiplication	<b>R &lt;- chol(A)</b>	Choleski factorization of <b>A</b> . Returns the upper triangular factor, such that <b>R'R = A</b> .
<b>A %o% B</b>	Outer product. <b>AB'</b>	<b>y &lt;- qr(A)</b>	QR decomposition of <b>A</b> . <b>y\$qr</b> has an upper triangle that contains the decomposition and a lower triangle that contains information on the Q decomposition. <b>y\$rank</b> is the rank of <b>A</b> . <b>y\$raux</b> a vector which contains additional information on Q. <b>y\$pivot</b> contains information on the pivoting strategy used.
<b>crossprod(A,B)</b>	<b>A'B</b> and <b>A'A</b> respectively.	<b>cbind(A,B,...)</b>	Combine matrices(vectors) horizontally. Returns a matrix.
<b>crossprod(A)</b>		<b>rbind(A,B,...)</b>	Combine matrices(vectors) vertically. Returns a matrix.
<b>t(A)</b>	Transpose		
<b>solve(A)</b>	Inverse of <b>A</b> where <b>A</b> is a square matrix.		
<b>y&lt;-eigen(A)</b>	<b>y\$val</b> are the eigenvalues of <b>A</b> <b>y\$vec</b> are the eigenvectors of <b>A</b>		

<http://www.statmethods.net/advstats/matrix.html>

## • 전치행렬(transpose) 구하기 (t)

```
# example lec2_1.r
m2<-matrix(1:6, ncol=3)
m2

# transpose of m2
tm2<-t(m2)
tm2
```



```
> m2<-matrix(1:6, ncol=3)
> m2
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
>
> # transpose of m2
> tm2<-t(m2)
> tm2
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
```

전치행렬은 행과 열을 바꾼 행렬

m2는 (2\*3)행렬,  
tm2는 (3\*2)행렬

## • determinant 구하기 (det)

determinant 식

$$|A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

```
# determinant of matrix,
d1<-matrix(1:4, nrow=2, byrow=T)
d1
det(d1)
```



```
> d1<-matrix(1:4, nrow=2, byrow=T)
> d1
      [,1] [,2]
[1,]    1    2
[2,]    3    4
> det(d1)
[1] -2
```

$$d1 = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \text{의 determinant는 } -2$$

## • 역행렬(inverse) 구하기 (solve)

$$d1 = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$\text{inverse}(d1) = \begin{bmatrix} -2.0 & 1.0 \\ 1.5 & -0.5 \end{bmatrix}$$

```
#inverse of matrix
d1_inv<-solve(d1)
d1_inv

# d1*inv(d1)=identity matrix
d1%%d1_inv
```



```
> d1_inv<-solve(d1)
> d1_inv
      [,1] [,2]
[1,] -2.0  1.0
[2,]  1.5 -0.5
> d1%%d1_inv
      [,1] [,2]
[1,]  1 1.110223e-16
[2,]  0 1.000000e+00
```

d1\*d1의 역행렬 = 단위행렬 (대각행렬이 1인 행렬)

## • 역행렬을 이용한 방정식 해 구하기 (solve)

$$\begin{aligned} 3x + 2y &= 8 \\ x + y &= 2 \end{aligned}$$



$$a = \begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}, b = \begin{bmatrix} 8 \\ 2 \end{bmatrix}$$

방정식의 해를 구하기 위해 a(행렬)와 b(벡터)를 생성

```
#solve equation
# 3x+2y=8, x+y=2

# matrix a, b
a <- matrix(c(3,1,2,1),nrow=2,ncol=2)
b <- matrix(c(8,2),nrow=2,ncol=1)
a
b
```

solve함수를 이용해 x와 y의 해를 찾음 (답 : x=4, y=-2)

solve(a,b)



```
> solve(a,b)
      [,1]
[1,]  4
[2,] -2
```

### • 매뉴얼 보기 : help(solve)

The screenshot shows an R console window on the left and the R documentation for the `solve` function on the right.

**Console:**

```
44 # x=4, y=-2
45 solve(a,b)
46 # to see solve
47 help(solve)
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

**Help Documentation:**

**Solve a System of Equations**

**Description**

This generic function solves the equation  $a \cdot x = b$  for  $x$ , where  $b$  can be either a vector or a matrix.

**Usage**

```
solve(a, b, ...)
```

**Arguments**

- a**: a square numeric or complex matrix containing the coefficients of the linear system. Logical matrices are coerced to numeric.
- b**: a numeric or complex vector or matrix giving the right-hand side(s) of the linear system. If missing,  $b$  is taken to be an identity matrix and `solve` will return the inverse of  $a$ .
- tol**: the tolerance for detecting linear dependencies in the columns of  $a$ . The default is `.Machine$double.eps`. Not currently used with complex matrices  $a$ .
- LINPACK**: logical. Defunct and ignored.

### • 고유치(eigenvalue)와 고유벡터(eigenvector)

```
# example for eigen value and eigen vector
# already centered matrix
x<-matrix(c(-3,-2,0, 1, 2, 2, -3, -3, 0, 2, 2, 2, 5,7,4,0,-5,-11), nrow =6, ncol=3)
x
dim(x)
```

(6\*3)의 행렬 x,  
행렬 x의 차원

```
> x
      [,1] [,2] [,3]
[1,]   -3   -3    5
[2,]   -2   -3    7
[3,]    0    0    4
[4,]    1    2    0
[5,]    2    2   -5
[6,]    2    2  -11
> dim(x)
[1] 6 3
```

### • 고유치(eigenvalue)와 고유벡터(eigenvector)

```
# eigen value and eigen vector
el<-eigen(t(x)%*%x)
el
```



```
> el<-eigen(t(x)%*%x)
> el
eigen() decomposition
$values
[1] 273.546962 13.845220 0.607818

$vectors
      [,1]      [,2]      [,3]
[1,] -0.2525343 0.5487321 0.79694382
[2,] -0.2841664 0.7452586 -0.60319073
[3,] 0.9249194 0.3787911 0.03227211
```

여기서  $t(x)\%*\%x$ 는 공분산행렬이라고 할수있음





## Wk2-4 : 간단한 함수생성 및 루프 (for, while)

### 간단한 함수 생성

#### • 함수 생성 (lec2\_4.r)

1 square 라는 함수의 생성

##### 프로그램 편집창

```
# lec2_4.r : loop, for, while
# create a simple function
# square function

square<-function(x){
  return(x*x)
}
square(9)
square(1:3)
```

##### 컨솔창

```
> square<-function(x){
+   return(x*x)
+ }
> square(9)
[1] 81
> square(1:3)
[1] 1 4 9
```

square라는 함수생성됨

square(9)=81

square(1:3)=(1, 4, 9)

## • 함수 생성

### 2 dif 라는 함수의 생성

#### 프로그램 편집창

```
dif<-function(x,y){
  return(x-y)
}
dif(20,10)
```

#### 컨솔창

```
> dif<-function(x,y){
+   return(x-y)
+ }
> dif(20,10)
[1] 10
```

dif 라는 함수생성됨

dif(20,10)=10

### 3 rootdif 라는 함수의 생성

```
rootdif<-function(x,y){
  return(sqrt(x-y))
}
rootdif(20,10)
```

```
> rootdif<-function(x,y){
+   return(sqrt(x-y))
+ }
> rootdif(20,10)
[1] 3.162278
```

rootdif 라는 함수생성됨

rootdif(20,10)=3.162278

## • 기존 함수의 코드보기 (예 : round 함수)

소수점 자리를 조정

#### 프로그램 편집창

```
# round off the decimal point
round(5.14846)
round(5.14846, 2)
```

#### 컨솔창

```
> round(5.14846)
[1] 5
> round(5.14846, 2)
[1] 5.15
```

```
# to see the function "round"
round
```

기존의 함수 round를 보기 위해서는  
컨솔창에 round를 치면 됨

```
> round
function (x, digits = 0) .Primitive("round")
```

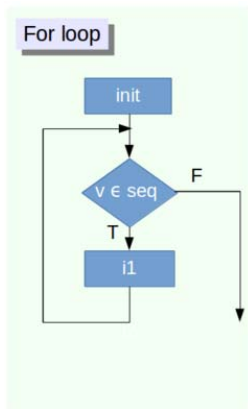
```
round(rootdif(20,10))
round(rootdif(20,10),2)
```

rootdif(20,10)=3.162278

```
> round(rootdif(20,10))
[1] 3
> round(rootdif(20,10),2)
[1] 3.16
```

소수점 둘째자리로 선택

## • For를 사용한 루프 (예제 1)



**for (1에서 10까지 )**  
**조건 if (3으로 나누었을때 나머지가 1)**  
**next (다음숫자로 루프를 돌림)**

```

# for 1 to 10
# if remainder=1 when deviding by 3
# then go to next number
# %%

for(i in 1:10){
  if(i%%3 == 1){
    next()
  }
  print(i)
}
  
```

컨솔창

```

> for(i in 1:10){
+   if(i%%3 == 1){
+     next()
+   }
+   print(i)
+ }
[1] 2
[1] 3
[1] 5
[1] 6
[1] 8
[1] 9
  
```

- 1부터 10까지의 숫자를 반복
- 3으로 나눴을 때의 나머지가 1인 경우  
next() → 현재 숫자에서 다음 숫자로 넘어감
- 넘어가지 않은 경우 print(i)를 통해 결과를 반환

## • For를 사용한 루프 (예제 2)

**i=1부터 10까지 1,2,3..으로 더해가며**  
**프린트하는데**  
**i>보다 크면 수행(print(i))를 멈춤**

```

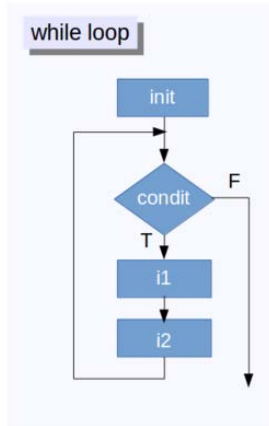
# for loop example2
# stop loop after i>5
# %%
for (i in 1:10) {
  i<-i+1
  print(i)
  if (i>5){
    # stop loop after i>5
    break
  }
}
  
```

Console ~/ ↻

```

> for (i in 1:10) {
+   i<-i+1
+   print(i)
+   if (i>5){
+     # stop loop after i>5
+     break
+   }
+ }
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
  
```

- while을 사용한 루프
- : 1부터 5까지의 숫자 출력



y가 5보다 적을때는 {expression}부분을 수행

```
# while loop
# while (condition) {expression}
y=0
while(y <5){ print( y<-y+1) }
```

컨솔창

```
> y=0
> while(y <5){ print( y<-y+1) }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

<https://www.datacamp.com/community/tutorials/tutorial-on-loops-in-r#gs.Oo6LckE>

	단위별 학습내용 (Week3)
wk3-1	R 데이터 생성 (불러들이기)
wk3-2	R 데이터활용 I (subset, 내보내기)
wk3-3	R 데이터 활용 II (dplyr활용)
wk3-4	데이터핸들링

## Wk3-1 : R 데이터 생성

# 1. 데이터 불러들이기 (csv, txt)

3-1. R 데이터생성 (불러들이기)

## • csv파일 불러들이기 (read.csv)

```
# 1. Read csv file : brain weight data  
brain<-read.csv("brain2210.csv")  
head(brain)  
dim(brain)
```

파일이름

csv : comma separated value  
데이터 저장시 범용 형태임

## • xls 파일

\*.xls파일인 경우 데이터를 csv(comma separated value)로 저장한 다음 read.csv 함수를 사용하여 R데이터로 불러들이는게 편리함

## • txt 파일 불러들이기 (read.table)

```
# 2. Read txt file with variable name
```

```
car<-read.table(file="autompg.txt", na=" ", header=TRUE)
```

파일이름

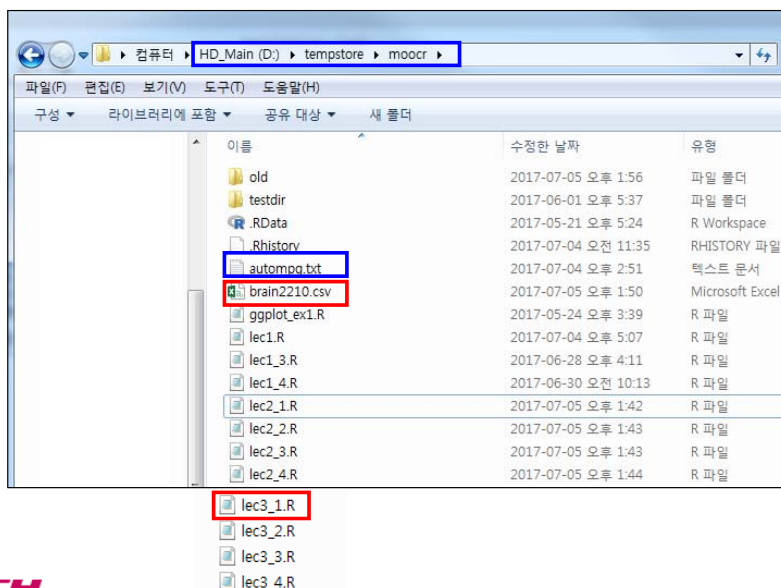
첫번째 줄은 변수이름

# 2. 데이터 저장 폴더

3-1. R 데이터생성 (불러들이기)

## • 데이터와 프로그램 저장 폴더 지정 (영문으로 폴더이름 생성)

예: D:/tempstore/moocr



brain2210.csv

	A	B	C	D
1	wt	sex		
2	1607	m		
3	1157	m		
4	1248	m		
5	1310	m		
6	1398	m		
7	1237	m		
8	1232	m		
9	1343	m		
10	1380	m		
11	1274	m		
12	1245	m		
13	1286	m		
14	1508	m		
15	1105	m		
16	1123	m		

## 2. 데이터 저장 폴더

3-1. R 데이터생성 (불러들이기)

### • 현재 프로그램 작업폴더 (setwd) :

```
# lec3_1.r
# Working directory
# Data import, frequency, histogram,
# attach, detach

# set working directory
# change working directory
setwd("D:/tempstore/moocr")

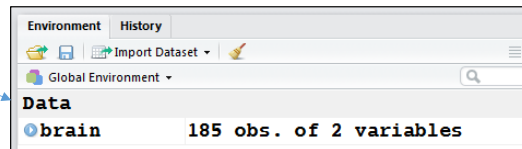
# check the current working directory
getwd()

# 1. Read csv file : brain weight data
brain<-read.csv("brain2210.csv")
head(brain)
dim(brain)
```

setwd : set working directory

brain2210.csv는 D:/tempstore/moocr에  
들어있으므로 working directory를 여기로 설정 !!

\*.csv를 R로 불러들인후 환경창을  
보면 brain이라는 이름의 R데이터  
가 생성되어있음



## 3. 데이터 불러들일때 tip

3-1. R 데이터생성 (불러들이기)

### • 데이터를 불러들일때 몇가지 tips

1. Working directory를 설정 : setwd("데이터가 저장되어있는 폴더")

2. 데이터를 불러들이고 확인

2-1. head(데이터이름) : 첫번째 6줄을 프린트해줌

2-2. dim(데이터이름) : 데이터의 관측치수와 변수의 갯수를 알려줌

```
# lec3_1.r
# Working directory
# Data import, frequency, histogram,
# attach, detach

# set working directory
# change working directory
setwd("D:/tempstore/moocr")

# check the current working directory
getwd()

# 1. Read csv file : brain weight data
brain<-read.csv("brain2210.csv")
head(brain)
dim(brain)
```



```
> head(brain)
   wt sex
1 1607  m
2 1157  m
3 1248  m
4 1310  m
5 1398  m
6 1237  m
> dim(brain)
[1] 185 2
```

## 4. 데이터와 변수이름

3-1. R 데이터생성 (불러들이기)

### • attach 사용

attach(데이터이름) : 데이터이름을 따로 지정하지 않아도 됨.

```
# 3. example for using 'attach'
# to get frequency of male and female (brain data)
table(brain$sex)

# using the command 'attach'
attach(brain)

# get frequency of male and female
table(sex)
```

table(변수) : 빈도 구하기  
(male과 female 몇명씩?)

```
> # to get frequency of male and female (brain data)
> table(brain$sex)

f    m
77 108

> attach(brain)
The following objects are masked from brain

sex, wt

> # get frequency of male and female
> table(sex)
sex
f    m
77 108
```

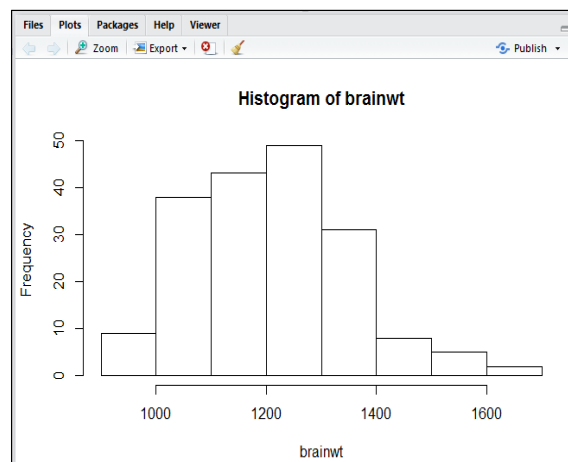
## 5. 데이터분석 활용

3-1. R 데이터생성 (불러들이기)

### • 데이터 알아보기 (히스토그램) : hist(변수이름)

```
# histogram of brain weight
# hist(brain$wt)
hist(wt)

detach(brain)
```



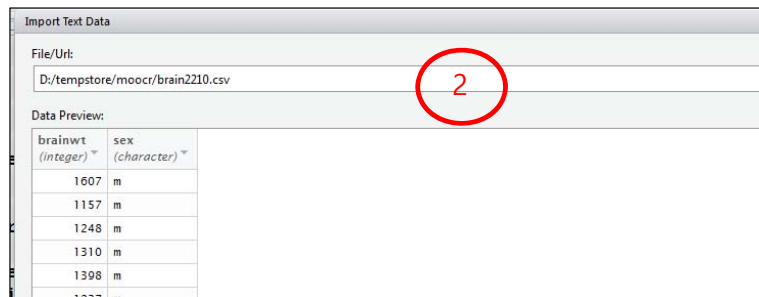
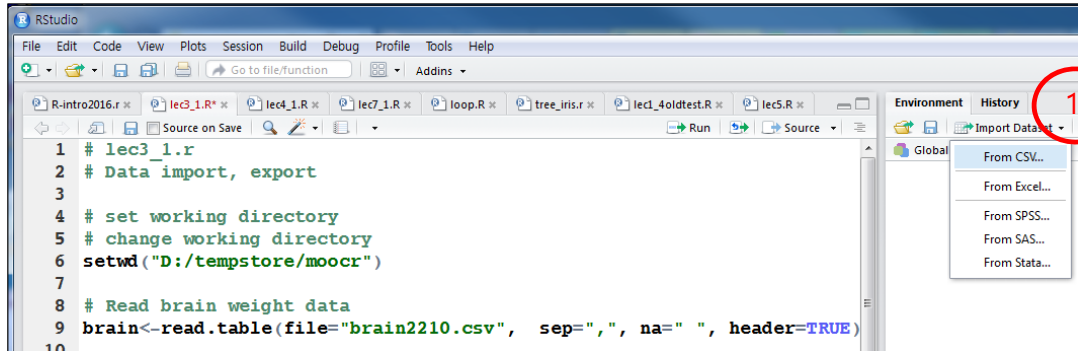
- attach(데이터이름) : 현재 세션에서 나오는 변수들은 그 '데이터'의 변수로 인식한다는 의미
- detach(데이터이름) : attach를 풀어줌



## 6. 데이터 불러들이기 (메뉴 선택방식)

3-1. R 데이터생성 (불러들이기)

### • 데이터 불러들이기



## Wk3-2 : R 데이터 활용 I (subset, 내보내기)

### 데이터 추출

#### • 데이터 추출 – subset(데이터이름, 조건)

예제1 : brain 데이터에서 female만 있는 subset 데이터생성

```
lec3_2.r
# Data read in & out
# import, export

# set working directory
# change working directory
setwd("D:/tempstore/moocr")

brain<-read.csv("brain2210.csv")
head(brain)

attach(brain)

# subset with female
# brainf<-subset(brain, sex=='f') after attach(brain)
brainf<-subset(brain, sex=='f')
mean(brainf$wt)
```

brain 데이터에서 female(여성)만 추출하여 brainf로 저장

```
> mean(brainf$wt)
[1] 1117.169
```

## 데이터 추출

3-2. R 데이터활용 1

### • 데이터 추출 – subset(데이터이름, 조건)

예제 2 : brain 데이터에서 wt<1300 이하인 데이터 생성

```
# subset with wt<1300
brain1300<-subset(brain,brain$wt<1300)

# same subset of brain1300
# brain1300<-subset(brain,!brain$wt>=1300)
summary(brain1300)
```

```
> brain1300<-subset(brain,brain$wt<1300)
> summary(brain1300)
      wt      sex
Min.   : 915   f:74
1st Qu.:1074   m:64
Median :1155
Mean   :1145
3rd Qu.:1230
Max.   :1289
```

원래 데이터 brain은  
여(74), 남(64)명

## 데이터 추출

3-2. R 데이터활용 1

### • 데이터 추출 – subset(데이터이름, 조건)

```
# subset with female
# brainf<-subset(brain, sex=='f')
brainf<-subset(brain, sex=='f')
mean(brainf$wt)
sd(brainf$wt)

# subset with male
brainm<-subset(brain, sex=='m')
mean(brainm$wt)
sd(brainm$wt)
```



```
> brainf<-subset(brain, sex=='f')
> mean(brainf$wt)
[1] 1117.169
> sd(brainf$wt)
[1] 98.97094
>
> # subset with male
> brainm<-subset(brain, sex=='m')
> mean(brainm$wt)
[1] 1270.741
> sd(brainm$wt)
[1] 129.22
```

요약표

	female	male
mean	1117.17	1270.74
sd	98.97	129.22

## 요약통계치 (그룹별)

3-2. R 데이터활용 1

### • 요약통계치 (그룹별) – aggregate(변수~그룹, 데이터, 함수)

```
# 'aggregate' for statistics by group
aggregate(wt~sex, data=brain, FUN=mean)
aggregate(wt~sex, data=brain, FUN=sd)
```

mean(평균)

sd(표준편차)

```
> aggregate(wt~sex, data=brain, FUN=mean)
  sex      wt
1  f 1117.169
2  m 1270.741
> aggregate(wt~sex, data=brain, FUN=sd)
  sex      wt
1  f  98.97094
2  m 129.21997
```

남녀별 뇌 무게 평균

남녀별 뇌 무게의 표준편차

그룹별 평균, 표준편차, 최소값, 최대값등을 비교하고자 하면  
aggregate를 이용하는것이 더 편리함

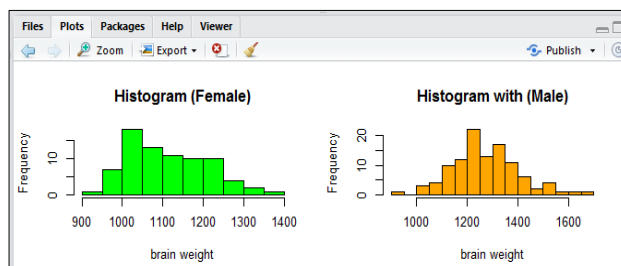
## 데이터 추출과 활용

3-2. R 데이터활용 1

### • 추출한 데이터의 활용 (그룹별 히스토그램)

```
# histogram for female and male
# 2*2 multiple plot
par(mfrow=c(2,2))
brainf<-subset(brain,brain$sex=='f')
hist(brainf$wt, breaks = 12,col = "green",cex=0.7,

# subset with male
brainm<-subset(brain,brain$sex=='m')
hist(brainm$wt, breaks = 12,col = "orange", main="")
```



## 데이터 추출과 활용

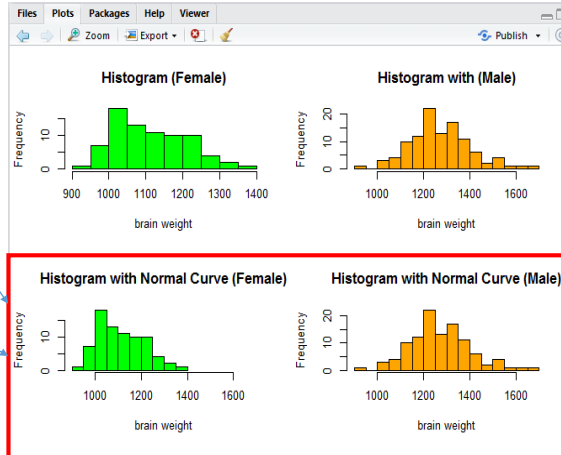
3-2. R 데이터활용 1

### • 추출한 데이터의 활용 (그룹별 히스토그램) - 눈여겨보기

```
# histogram with same scale
hist(brainf$wt, breaks = 12,col = "green",cex=0.7, main="Histogram (Female)")
hist(brainm$wt, breaks = 12,col = "orange", main="Histogram (Male)")
```

???  
또 다른문제는?

\* 남녀간 동일범위로  
설정하고 히스토그램  
을 그리고 비교한다!!



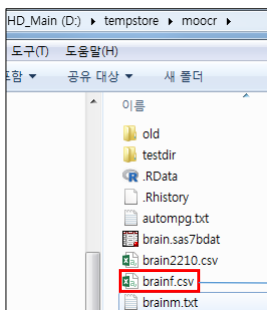
## 데이터 내보내기

3-2. R 데이터활용 1

### • csv로 내보내기 (write.table, write.csv)

```
# export csv file - write out to csv file
write.table(brainf,file="brainf.csv", row.names = FALSE, sep=" ", na=" ")
```

```
write.csv(brainf,file="brainf.csv", row.names = FALSE)
```



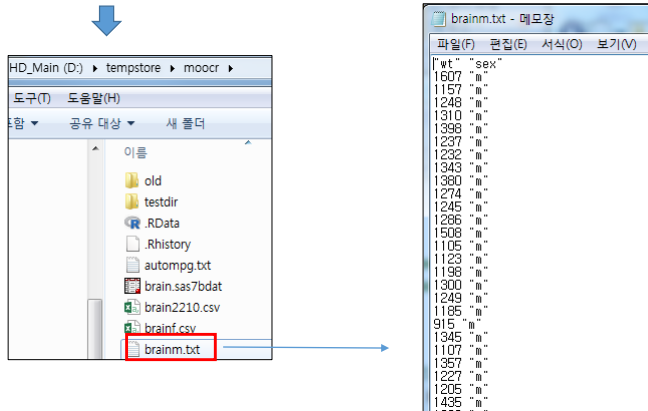
brainf.csv - Excel					
	wt	sex			
1	1125	f			
2	1027	f			
3	1112	f			
4	983	f			
5	1090	f			
6	1247	f			
7	1045	f			
8	983	f			
9	972	f			
10	1045	f			
11					

## 데이터 내보내기

3-2. R 데이터활용 1

### • txt 로 내보내기 (write.table)

```
# export txt file - write out to csv file
write.table(brainm, file="brainm.txt", row.names = FALSE, na=" ")
```



## Wk3-3 : R 데이터 활용 II (dplyr 활용)

### dplyr 활용

- dplyr (데이터핸들링을 편리하게 수행할 수 있는 패키지)
- 코드가 읽고 쓰기 쉽고 데이터를 빠르게 핸들링 할 수 있음

#### dplyr 패키지의 주요 함수

**select** : 일부변수를 선택

**filter** : 필터링 기능 (조건에 맞는 데이터 추출)

**mutate** : 새로운 변수 생성

**group\_by** : 그룹별 통계량을 얻을때

**summarize** : 요약통계량 (mean, min, max, sum)

**arrange** : 행 정렬시 사용

# dplyr 활용 - 데이터 핸들링

## • dplyr 패키지 설치 및 기본 설정

### 프로그램 편집창

```
# lec3_3.r
# Data handling
# Data analysis with autompg.csv

# data manipulation package
# select, filter, group by, summarise in dplyr package
install.packages("dplyr")
library(dplyr)

# set working directory
# change working directory
setwd("D:/tempstore/moocr")

# Read txt file with variable name
# http://archive.ics.uci.edu/ml/datasets/Auto+MPG

# Data reading in R
car<-read.csv(file="autompg.csv")
attach(car)

# change to tbl_df class
car<-tbl_df(car)
```

Step0 : 분석을 위한 설정  
(install, library, setwd)

Step1 : 데이터핸들링  
(csv파일 불러들이기, ...)

tbl\_df() : 빅데이터에 유용함.  
df는 데이터프레임을 의미

# 오픈 데이터 활용

## • UCI repository data

UCI Machine Learning Repository  
Center for Machine Learning and Intelligent Systems

**Auto MPG Data Set**  
Download: [Data Folder](#), [Data Set Description](#)

Abstract: Revised from CMU StatLib library, data concerns city-cycle fuel...

Data Set Characteristics: Multivariate, Number of Instances: 392

Attribute Characteristics: Categorical, Real, Number of Attributes: 9

Associated Tasks: Regression, Missing Values?: No

Source: This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University.

Data Set Information: This dataset is a slightly modified version of the dataset provided in the StatLib library. The original dataset is available at <http://www.statlib.org/datasets/Auto+MPG>. The data concerns city-cycle fuel consumption in miles per gallon, to be predicted from 7 other features.

autompg.csv

	A	B	C	D	E	F	G	H	I
1	mpg	cyl	disp	hp	wt	accler	year	origin	carname
2	18	8	307	17	3504	12	70	1	chevrolet d
3	15	8	350	35	3693	11.5	70	1	buick skylar
4	18	8	318	29	3436	11	70	1	plymouth s
5	16	8	304	29	3433	12	70	1	amc rebel s
6	17	8	302	24	3449	10.5	70	1	ford torino
7	15	8	429	42	4341	10	70	1	ford galaxie
8	14	8	454	47	4354	9	70	1	chevrolet ir
9	14	8	440	46	4312	8.5	70	1	plymouth f
10	14	8	455	48	4425	10	70	1	pontiac cat
11	15	8	390	40	3850	8.5	70	1	amc ambas
12	15	8	383	37	3563	10	70	1	dodge chal
13	14	8	340	34	3609	8	70	1	plymouth v

<http://archive.ics.uci.edu/ml/datasets/Auto+MPG>



### • autmpg 데이터

1. mpg: continuous (연비 : 연속형변수)
2. cylinders: multi-valued discrete (실린더 : 정수값)
3. displacement: continuous (배기량 : 연속형변수)
4. horsepower: continuous (마력 : 연속형변수)
5. weight: continuous (무게 : 연속형변수)
6. acceleration: continuous (가속 : 연속형변수)
7. year: multi-valued discrete (모델연도 : 정수값)
8. origin: multi-valued discrete (정수값)
9. car name: string (unique for each instance) (차종류 이름)

### • 변수특성 변경 (as.numeric, as.integer, factor)

# 데이터 활용 II

## 3-3. R 데이터 활용 II

### 1. 데이터 불러들이기

```
# Data reading in R
car<-read.csv(file="autmpg.csv")
attach(car)

# Change to tbl_df class
car<-tbl_df(car)

head(car)
dim(car)
str(car)
```

R 데이터 이름은? : car

car 데이터의 수는? : 398

car 는 몇개의 변수? : 9

```
> head(car)
# A tibble: 6 x 9
  mpg   cyl  disp    hp  wt     accler  year origin carname
<dbl> <int> <dbl> <dbl> <dbl> <dbl> <int> <int> <fct>
1    18     8   307    17  3504     12     70     1 chevrolet chevelle malibu
2    15     8   350    35  3693    11.5    70     1 buick skylark 320
3    18     8   318    29  3436     11     70     1 plymouth satellite
4    16     8   304    29  3433     12     70     1 amc rebel sst
5    17     8   302    24  3449    10.5    70     1 ford torino
6    15     8   429    42  4341     10     70     1 ford galaxie 500

> dim(car)
[1] 398  9
```

# 데이터 활용 II

## 2. 데이터의 전체 구조 파악하기 : str(데이터이름)

```
# Data reading in R
car<-read.csv(file="autompg.csv")
attach(car)

# Change to tbl_df class
car<-tbl_df(car)

head(car)
dim(car)
str(car)
```

car 변수들 중 실수값으로 저장된 변수들은? 4개

car 변수들 중 정수값으로 저장된 변수들은? 4개

car 변수들중 factor로 저장된 변수는? 1개

```
> str(car)
tibble [398 x 9] (S3: tbl_df/tbl/data.frame)
 $ mpg      : num [1:398] 18 15 18 16 17 15 14 14 15 ...
 $ cyl      : int [1:398] 8 8 8 8 8 8 8 8 8 8 ...
 $ disp     : num [1:398] 307 350 318 304 302 429 454 440 455 390 ...
 $ hp       : num [1:398] 17 35 29 29 24 42 47 46 48 40 ...
 $ wt       : int [1:398] 3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
 $ accler   : num [1:398] 12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
 $ year     : int [1:398] 70 70 70 70 70 70 70 70 70 70 ...
 $ origin   : int [1:398] 1 1 1 1 1 1 1 1 1 1 ...
 $ carname  : Factor w/ 305 levels "amc ambassador brougham",...: 50 37 232 15 162
 142 55 224 242 2 ...
```

# 데이터 활용 II

## 3. 데이터 요약하기 : summary(데이터이름)

```
# summary
summary(car)
```

최소값  
25%값  
중위수  
평균  
75%값  
최대값

```
> summary(car)

      mpg          cyl          disp          hp
Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 1.00
1st Qu.:17.50   1st Qu.:4.000   1st Qu.:104.2   1st Qu.:26.00
Median :23.00   Median :4.000   Median :148.5   Median :60.50
Mean   :23.51   Mean   :5.455   Mean   :193.4   Mean   :51.39
3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:262.0   3rd Qu.:79.00
Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :94.00

      wt          accler          year          origin
Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
1st Qu.:2224   1st Qu.:13.82   1st Qu.:73.00   1st Qu.:1.000
Median :2804   Median :15.50   Median :76.00   Median :1.000
Mean   :2970   Mean   :15.57   Mean   :76.01   Mean   :1.573
3rd Qu.:3608   3rd Qu.:17.18   3rd Qu.:79.00   3rd Qu.:2.000
Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000

      carname
ford pinto      : 6
amc matador     : 5
ford maverick   : 5
toyota corolla  : 5
amc gremlin     : 4
amc hornet      : 4
(Other)         :369
```

## 4. 데이터의 요약통계치 (빈도 구하기) : table(데이터이름)

```
table(origin)
table(year)
```



```
> table(origin)
origin
 1    2    3
249  70  79
```

attach(데이터이름) : 현재 세션에서 나오는 변수들은 그 '데이터'의 변수로 인식  
=> 데이터이름을 안써줘도 됨!

```
> table(year)
year
70 71 72 73 74 75 76 77 78 79 80 81 82
29 28 28 40 27 30 34 28 36 29 29 29 31
```

1970년도부터 1982년까지의 차량

## 5. 데이터의 요약통계치 (평균, 표준편차) 구하기

개별변수의 평균 구하기

```
# mean and standard deviation
mean(mpg)
mean(hp)
mean(wt)
```



```
> mean(mpg)
[1] 23.51457
> mean(hp)
[1] 51.38945
> mean(wt)
[1] 2970.425
```

# dplyr 활용 - 데이터 추출 (select)

- 변수 추출 : select(데이터, 변수이름, ...)

car 데이터에서 mpg, hp 변수만 추출

```
# 6.1 subset data : selecting a few variables
set1<-select(car, mpg, hp)
head(set1)
```

```
> head(set1)
  mpg hp
1  18 17
2  15 35
3  18 29
4  16 29
5  17 24
6  15 42
```

- 변수 추출 : `select(데이터, 변수이름, ... )`

car 데이터에서 **mpg로 시작하는** 변수를 제외하고 set2 라는 데이터를 생성

```
# 6.2 subset data : Drop variables with -  
set2<-select(car, -starts_with("mpg"))  
head(set2)
```

`starts_with()`: 변수 시작



## Wk3-4 : 데이터핸들링

### dplyr 활용 - 데이터 핸들링

#### • dplyr 패키지 설치 및 기본 설정

##### 프로그램 편집창

```
# lec3_3.r
# Data handling
# Data analysis with autompg.csv

# data manipulation package
# select, filter, group by, summarise in dplyr package
install.packages("dplyr")
library(dplyr)

# set working directory
# change working directory
setwd("D:/tempstore/moocr")

# Read txt file with variable name
# http://archive.ics.uci.edu/ml/datasets/Auto+MPG

# Data reading in R
car<-read.csv(file="autompg.csv")
attach(car)

# Change to tbl_df class
car<-tbl_df(car)
```

Step0 : 분석을 위한 설정  
(install, library, setwd)

Step1 : 데이터핸들링  
(csv파일 불러들이기, ... )

tbl\_df() : 빅데이터에 유용함.  
df는 데이터프레임을 의미

## dplyr 활용 - 데이터 추출 (filter)

- 조건식에 맞는 데이터 추출 : filter(데이터, 변수조건, ... )

car 데이터에서 mpg가 30보다 큰 행 추출

```
# 4. subset data : filter mpg>50
set3<-filter(car, mpg>30)
head(set3)
```



```
> head(set3)
# A tibble: 6 x 9
  mpg   cyl  disp    hp  wt  accler  year  origin carname
<dbl> <int> <dbl> <dbl> <int> <dbl> <int> <int> <fct>
1    31     4    71     62  1773    19     71     3  toyota corolla 1200
2    35     4    72     66  1613    18     71     3  datsun  1200
3    31     4    79     64  1950    19     74     3  datsun  b210
4    32     4    71     62  1836    21     74     3  toyota corolla 1200
5    31     4    76     53  1649   16.5    74     3  toyota corona
6    32     4    83     58  2003    19     74     3  datsun  710
```

## dplyr 활용 - 변수생성 (mutate)

- 변수 생성 : mutate(새로운 변수이름=기존변수 활용)

%>%(파이프 연산자) 연산자 사용하여 연결

```
# 5. create a derived variable
set4 <- car %>% ①
  filter(!is.na(hp)) %>% ②
  mutate(hp_km = hp*1.609) ③
head(set4)
```

Window

%>%의 단축키

Mac

Shift + Ctrl + M

Shift + Cmd + M

파이프연산자 : 앞에서부터 ①,②,③ 순서대로  
수행하여 데이터전처리를 하고 set4라는 이름  
으로 저장

- filter car데이터 hp열의 NA가 아닌 모든 데이터 추출

is.na() NA여부 판단하는 함수 (! 기호는 부정하는 기호)

- mutate 기존의 hp열 사용하여 새로운 hp\_km열 생성

```
# A tibble: 6 x 10
  mpg   cyl  disp    hp  wt  accler  year  origin carname hp_km
<dbl> <int> <dbl> <dbl> <int> <dbl> <int> <int> <fct> <dbl>
1    18     8   307    17  3504    12     70     1  chevrolet chevelle malibu 27.4
2    15     8   350    35  3693   11.5    70     1  buick skylark 320 56.3
3    18     8   318    29  3436    11     70     1  plymouth satellite 46.7
4    16     8   304    29  3433    12     70     1  amc rebel sst 46.7
5    17     8   302    24  3449   10.5    70     1  ford torino 38.6
6    15     8   429    42  4341    10     70     1  ford galaxie 500 67.6
```

# dplyr 활용 - 데이터 요약통계치

- 데이터 요약통계치(평균 구하기) : `summarize(mean(변수이름))`

mpg, hp, wt의 평균값 구하기

```
# mean and standard deviation
car %>%
  summarize(mean(mpg), mean(hp), mean(wt))
```



```
> car %>%
+   summarize(mean(mpg), mean(hp), mean(wt))
# A tibble: 1 x 3
  `mean(mpg)` `mean(hp)` `mean(wt)`
  <dbl>      <dbl>      <dbl>
1    23.5      51.4      2970.
```

몇 개 변수들의 평균값 한번에 구하기

```
# mean of some variables
select(car, 1:6) %>%
  colMeans()
```



```
> select(car, 1:6) %>%
+   colMeans()
      mpg      cyl      disp
23.514573  5.454774 193.425879
      hp      wt      accler
51.389447 2970.424623 15.568090
```

1~6열 추출함

- colMeans 데이터를 열로 재구성하여 평균값 구함

# dplyr 활용 - 데이터 요약통계치

- 벡터화 요약치 : `summarize_all(FUN)`

열 추출해 기술통계치 구하고 요약치 보여줌

```
# table with descriptive statistics
a1 <- select(car, 1:6) %>% summarize_all(mean)
a2 <- select(car, 1:6) %>% summarize_all(sd)
a3 <- select(car, 1:6) %>% summarize_all(min)
a4 <- select(car, 1:6) %>% summarize_all(max)
table1 <- data.frame(rbind(a1,a2,a3,a4))
rownames(table1) <- c("mean","sd","min","max")
table1
```

참고: `tbl_df()` 적용결과

```
# A tibble: 6 x 9
  mpg  cyl  disp  hp  wt  accler  year  origin  carname
<dbl> <int> <dbl> <dbl> <int> <dbl> <int> <int> <fct>
1    18     8   307   17  3504   12    70     1 chevrolet chevelle malibu
2    15     8   350   35  3693   11.5  70     1 buick skylark 320
3    18     8   318   29  3436   11    70     1 plymouth satellite
4    16     8   304   29  3433   12    70     1 amc rebel sst
5    17     8   302   24  3449   10.5  70     1 ford torino
6    15     8   429   42  4341   10    70     1 ford galaxie 500
```

• tibble 행이름(rownames) 지원하지 않음

data.frame을 `tbl_df`로 전환시켰으므로  
data.frame으로 원상복귀시켜서 행 이름을 바꾼다.



```
> table1 <- data.frame(rbind(a1,a2,a3,a4))
> rownames(table1) <- c("mean","sd","min","max")
> table1
      mpg      cyl      disp      hp      wt      accler
mean 23.514573  5.454774 193.4259 51.38945 2970.4246 15.568090
sd   7.815984  1.701004 104.2698 29.93236  846.8418  2.757689
min   9.000000  3.000000  68.0000  1.00000 1613.0000  8.000000
max  46.600000  8.000000 455.0000 94.00000 5140.0000 24.800000
```

- 그룹별 통계량 얻기 : `group_by(변수), summarize( __=FUN())`

## 그룹별 요약통계량 구하기

```
# summary statistics by group variable
car %>%
  group_by(cyl) %>%
  summarize(mean_mpg = mean(mpg, na.rm = TRUE))
```



```
# A tibble: 5 x 2
  cyl mean_mpg
<int> <dbl>
1     3    20.6
2     4    29.3
3     5    27.4
4     6    20.0
5     8    15.0
```

- `group_by` car데이터의 cyl열을 그룹으로 묶음

- `summarize()` cyl그룹의 mpg 평균을 구함

→ `na.rm = TRUE` 통계 분석 시 결측 값을 제외함

remove

요약통계량을 구할 때

`group_by`와 `summarize` 함께 사용하는 경우 많음

함수	요약통계량
mean	평균
min	최솟값
max	최댓값
sum	합계
var	분산
sd	표준편차
median	중앙값
n	빈도



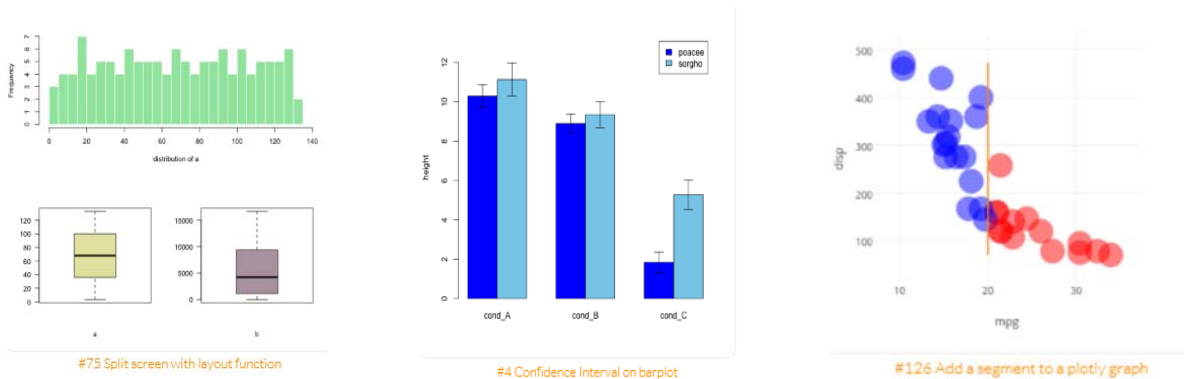


	<b>단위별 학습내용 (Week4)</b>
<b>wk4-1</b>	<b>R 그래픽 기초 I (히스토그램)</b>
<b>wk4-2</b>	<b>R 그래픽 기초 II (상자그림, 파이차트)</b>
<b>wk4-3</b>	<b>R 그래픽 기초 III (산점도)</b>
<b>wk4-4</b>	<b>그래픽과 레이아웃</b>

## Wk4-1 : R 그래픽 기초 I

- R그래픽, 히스토그램 -

- 데이터 시각화 : 정보의 요약된 형태를 그래프로 전달
- 빅데이터의 시각화를 통해 새로운 사실(인과관계) 발견
- 시각적 요약을 통해 한걸음 더 나아간 통찰력 (Insight)



<http://www.r-graph-gallery.com/all-graphs/> 참조

## R 그래프 – 데이터의 분포

1. 히스토그램 (histogram) : 1차원 (univariate, 일변량)
2. 상자그림 (Boxplot) : 1차원 (데이터의 분포를 파악)
3. 막대그림 (Bar plot) : 1차원 (범주형데이터의 빈도분포)
4. 파이차트 (pie chart) : 1차원 (각 범주별 비율을 그림으로)
5. 산점도 (scatterplot) : 2차원 (x와 y간의 관계를 해석)

R 그래픽 사이트 :

- <http://www.r-graph-gallery.com/>
- <http://www.cookbook-r.com/Graphs/>

# 1. 히스토그램 (1차원)

4-1. R 그래픽 기초 1

## • Brain데이터를 이용한 그래프

```
# lec4_1.r : Basic Graphics I

# set working directory
# change working directory
setwd("D:/tempstore/moocr")

# Read brain data (lec3_1.R)
brain<-read.csv(file="brain2210.csv")

head(brain)
dim(brain)

attach(brain)
```

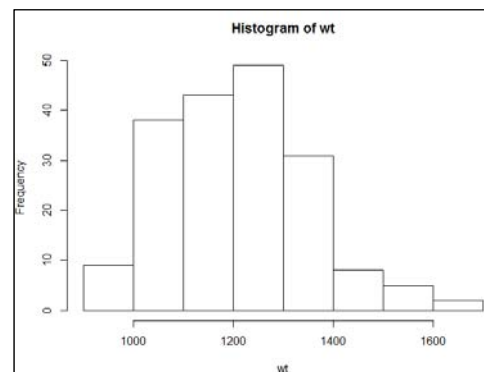
데이터 불러들이기 단계 :  
setwd, read.csv

# 1. 히스토그램 (1차원)

4-1. R 그래픽 기초 1

## • 히스토그램 : hist(변수이름)

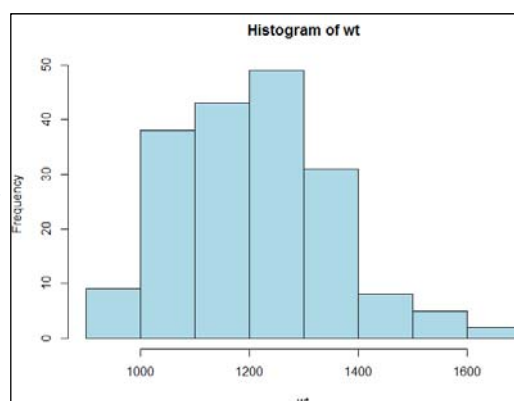
```
# 1. histogram
# 1-1. histogram with no options
# hist(brain$wt)
hist(wt)
```



색상 선택

```
help(hist)
```

```
hist(wt, col = "lightblue")
```



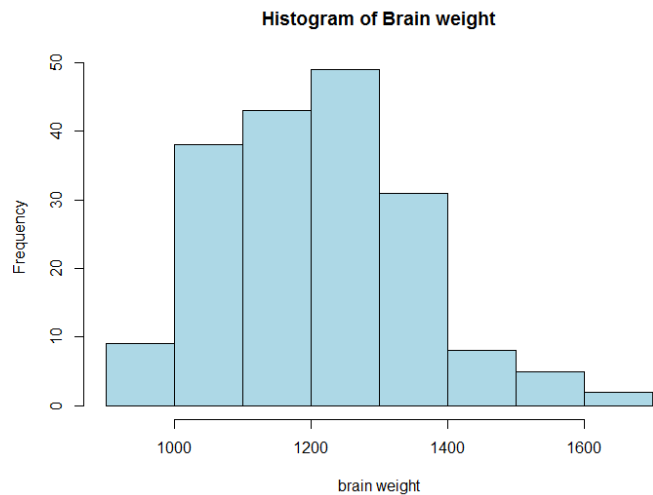
# 1. 히스토그램 (1차원)

- 히스토그램 (색과 제목) : `hist(변수이름, col="colname", main=" ")`

```
# 1-2. histogram with color and title, legend
hist(wt, breaks = 10, col = "lightblue", main="Histogram of Brain weight")
```

`col="colname", main="그림제목"`

예시 : `col="blue",  
main="Histogram of weight"`



# 1. 히스토그램 (1차원)

- 색 (657가지 색)

`colors()` → 모든 색의 이름을 볼수 있음

- `grep("단어", colors(), value=TRUE)` : '단어'가 포함된 색을 검색해줌

```
# see rgb values for 657 colors, choose what you like
colors()

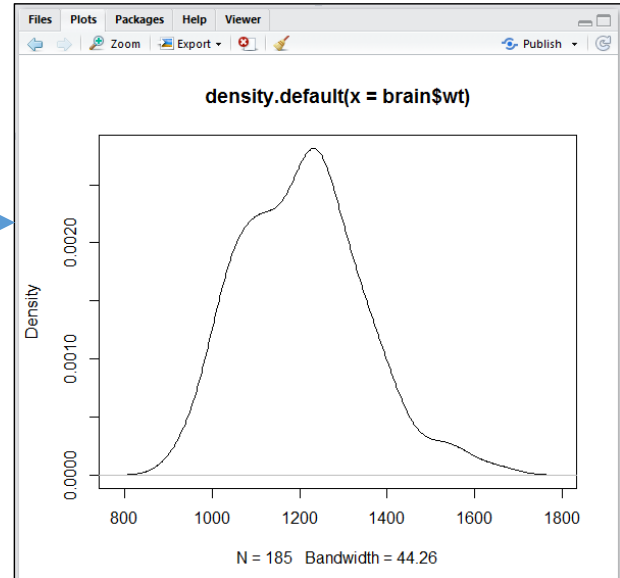
# select colors including "blue"
grep("blue", colors(), value=TRUE)
```

```
> # select colors including "blue"
> grep("blue", colors(), value=TRUE)
[1] "aliceblue"      "blue"           "blue1"
[4] "blue2"          "blue3"          "blue4"
[7] "blueviolet"     "cadetblue"      "cadetblue1"
[10] "cadetblue2"     "cadetblue3"     "cadetblue4"
[13] "cornflowerblue" "darkblue"       "darkslateblue"
[16] "deepskyblue"    "deepskyblue1"   "deepskyblue2"
[19] "deepskyblue3"   "deepskyblue4"   "dodgerblue"
```

# 1. 히스토그램 (1차원)

## • 밀도함수 그려보기

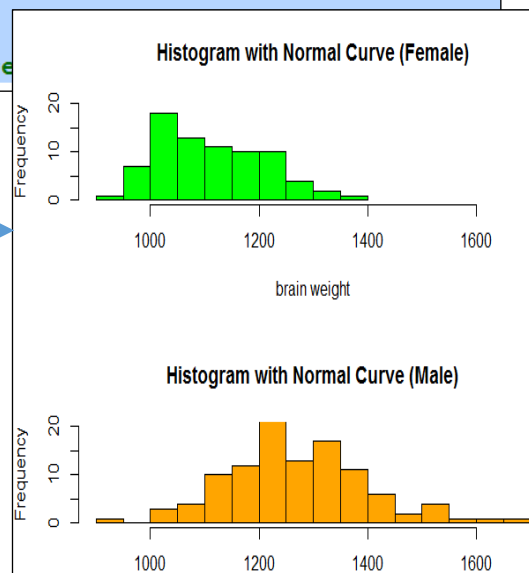
```
# 1-3. fit function (find density function)
par(mfrow=c(1,1))
d <- density(brain$wt)
plot(d)
```



# 1. 히스토그램 (1차원)

## • 그룹별 히스토그램 (동일한 x축, y축 범위) : xlim, ylim

```
# 1-4. histogram with same scale
# 2 multiple plot
par(mfrow=c(2,1))
brainf<-subset(brain,brain$sex=='f')
hist(brainf$wt, breaks = 12,col = "green", xlim=c(900,1700),ylim=c(0,20),
      main="Histogram with Normal Curve (Female)",
      las=1)
brainm<-subset(brain,brain$sex=='m')
hist(brainm$wt, breaks = 12,col = "orange", xlim=c(900,1700),ylim=c(0,20),
      main="Histogram with Normal Curve (Male)",
      las=1)
```



예 : xlim=c(0,100), ylim(0,100)

par(mfrow=c(2,1)) : 그래프화면의 분할을 행(row)는 2행으로, 열은 1열로 하라는 의미



## Wk4-2 : R 그래픽 기초 II

- 상자그림, 파이차트 -

## 2. 상자그림 (Boxplot, 1차원)

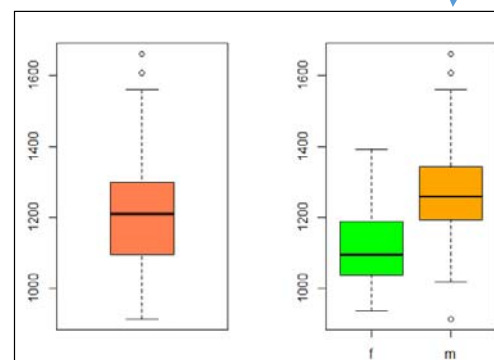
2-1. 상자그림 : `boxplot(변수이름, col=c("colname"))`

```
# 2. boxplot
par(mfrow=c(1,2))
# 2-1. boxplot for all data
boxplot(brain$wt, col=c("coral"))
```

2-2. 그룹별 상자그림 : `boxplot(변수이름~그룹이름, col=c("col1", "col2"))`

```
# boxplot by gender (female, male)
boxplot(brain$wt~brain$sex, col = c("green", "orange"))
```

`par(mfrow=c(1,2))` : 그래프화면의 분할을 행(row)는 1행으로, 열은 2열로 하라는 의미

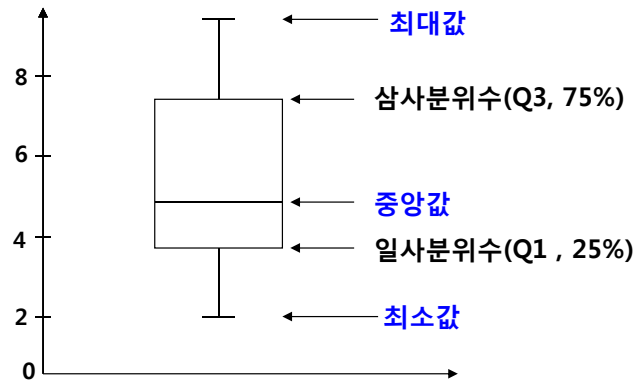


## 2. 상자그림 (Boxplot, 1차원)

4-2. R 그래픽 기초 II

### •상자그림 설명

데이터의 분포를 사분위수를 중심으로 설명해주는 그림



\* : Q1, Q3로부터  $\pm 1.5$  IQR 넘는값 (이상치로 볼수 있음)

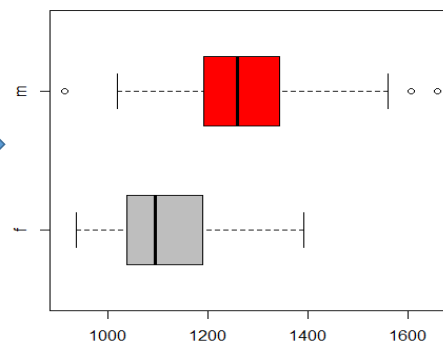
## 2. 상자그림 (Boxplot, 1차원)

4-2. R 그래픽 기초 II

2-3. 수평 상자그림 : `boxplot(변수이름, col=c("colname"), horizontal=TRUE,)`

```
# 2-3 horizontal boxplot
par(mfrow=c(1,1))
boxplot(brain$wt~brain$sex, boxwex=0.5, horizontal=TRUE, col = c("grey", "red"))
```

수평으로 상자그림을 그릴수 있음



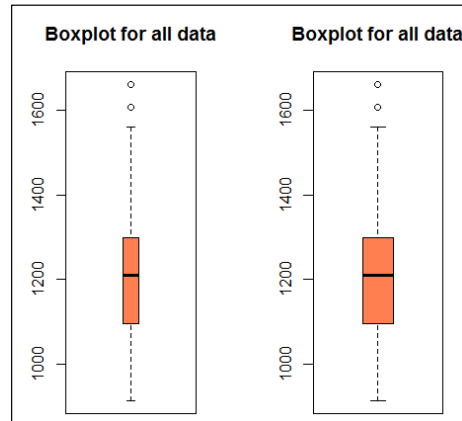
## 2. 상자그림 (Boxplot, 1차원)

4-2. R 그래픽 기초 II

2-4. 상자그림 : `boxplot(변수이름, col=c("colname"), boxwex= )`

```
# 2-4 box width boxwex (width of box)
par(mfrow=c(1,2))
boxplot(brain$wt, boxwex = 0.25, col=c("coral"), main="Boxplot for all data")
boxplot(brain$wt, boxwex = 0.5, col=c("coral"), main="Boxplot for all data")
```

boxwex : 그림상자의 폭을 조정



## 2. 상자그림 (Boxplot, 1차원)

4-2. R 그래픽 기초 II

2-5. 상자그림에 기술통계치 넣기 : 관측치수(n) 넣기

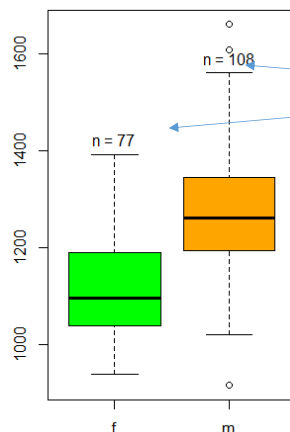
```
# 2-5 add text (n) over a boxplot
par(mfrow=c(1,2))
a<-boxplot(brain$wt~brain$sex, col = c("green", "orange"))
text(c(1:nlevels(brain$sex)), a$stats[nrow(a$stats),]+30, paste("n = ",table(brain$sex),sep=""))
```

글자위치

```
> summary(a)
      Length Class      Mode 
stats    10   integer numeric
n         2    -none-  numeric
conf      4    -none-  numeric
out        3    -none-  numeric
group      3    -none-  numeric
names     2    -none-  character
```

```
> a$stats
      [,1] [,2]
[1,]  937 1020
[2,] 1038 1193
[3,] 1096 1260
[4,] 1190 1344
[5,] 1392 1561
```

nrow(a\$stats)  
여기서 nrow=5이므로



```
> table(brain$sex)
  f  m
77 108
```



## • autompg 데이터 (lec3\_3.R에서 사용)

1. mpg: continuous (연비 : 연속형변수)
2. cylinders: multi-valued discrete (실린더 : 정수값)
3. displacement: continuous (배기량 : 연속형변수)
4. horsepower: continuous (마력 : 연속형변수)
5. weight: continuous (무게 : 연속형변수)
6. acceleration: continuous (가속 : 연속형변수)
7. year: multi-valued discrete (모델연도 : 정수값)
8. origin: multi-valued discrete (정수값)
9. car name: string (unique for each instance) (차종류 이름)

```
# 3. scatterplot after attach command

# use autompg data (lec3_3.R)
# set working directory
setwd("D:/tempstore/moocr")
car<-read.csv("autompg.csv")
head(car)
```

```
Console D:/tempstore/moocr/
mpg cyl disp hp wt accler year origin
1 18 8 307 130.0 3504 12.0 70 1
2 15 8 350 165.0 3693 11.5 70 1
3 18 8 318 150.0 3436 11.0 70 1
4 16 8 304 150.0 3433 12.0 70 1
5 17 8 302 140.0 3449 10.5 70 1
6 15 8 429 198.0 4341 10.0 70 1
      carname
1 chevrolet chevelle malibu
2 buick skylark 320
3 plymouth satellite
4 amc rebel sst
5 ford torino
```

## 3. 막대그림

## • barplot(변수빈도, col=c("col1", "col2", ..))

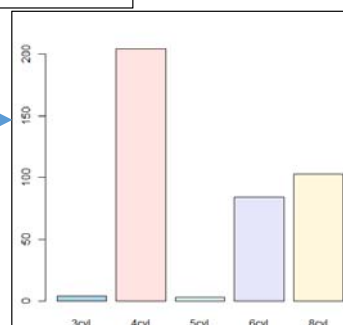
막대그림을 그리기 위해서는 우선 table(변수이름)을 이용하여 빈도를 계산함

```
# 3. bar plot with cylinder count (lec3_3.R)
# par(mfrow=c(1,1))
table(car$cyl)
freq_cyl<-table(cyl)
names(freq_cyl) <- c("3cyl", "4cyl", "5cyl", "6cyl",
"8cyl")
barplot(freq_cyl, col = c("lightblue", "mistyrose", "lightcyan",
"lavender", "cornsilk"))
```

autompg 데이터의 cylinder 빈도

```
> table(car$cyl)

 3    4    5    6    8 
4 204    3  84 103
```



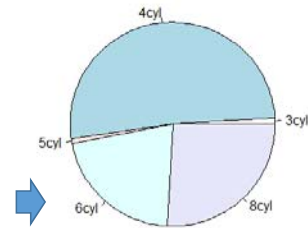
## 4. 파이차트

4-2. R 그래픽 기초 II

- `pie(변수빈도, labels=c(" ", ..))`

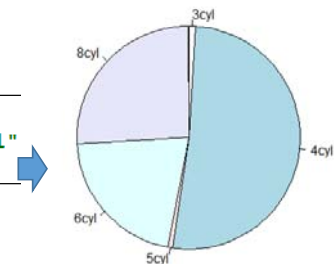
파이차트를 그리기 위해서는 우선 `table(변수이름)`을 이용하여 빈도를 계산함

```
# 4. pie chart
# You can also custom the labels:
freq_cyl<-table(cyl)
names(freq_cyl) <- c ("3cyl", "4cyl", "5cyl", "6cyl", "8cyl")
# 4-1 pie chart
pie(freq_cyl)
```



시계방향으로 파이차트를 그리기 위해서는

```
# 4-2 pie chart clockwise
pie(freq_cyl, labels = c("3cyl", "4cyl", "5cyl", "6cyl", "8cyl",
clockwise = TRUE)
```



## 4. 파이차트

4-2. R 그래픽 기초 II

- `pie(변수빈도, labels=c(" ", ..))`

연습과제 : `autopg`의 `subset` (cylinder가 4,6,8인 차종만 분석)의 파이차트

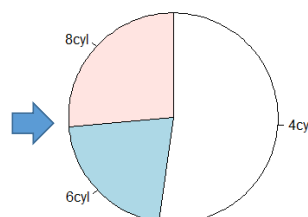
```
> table(car$cyl)
 3    4    5    6    8 
4 204    3   84  103
```

← `autopg` 데이터의 cylinder 빈도를 보면 3cyl과 5cyl은 소수차량임

```
# 4-3 pie chart of subset
# subset with cylinder (4,6,8) - refresh creating subset data lec3_2.R
car1<-subset(car, cyl==4 | cyl==6 | cyl==8)
table(car1$cyl)
```

```
> table(car1$cyl)
 4    6    8 
204   84  103
```

lec4\_2.R을 참고하여 수행하면



## Wk4-3 : R 그래픽 기초 III

- 산점도 -

## autompg(차의 연비) 데이터

### • autompg 데이터 (lec3\_3.R에서 사용)

1. mpg: continuous (연비 : 연속형변수)
2. cylinders: multi-valued discrete (실린더 : 정수값)
3. displacement: continuous (배기량 : 연속형변수)
4. horsepower: continuous (마력 : 연속형변수)
5. weight: continuous (무게 : 연속형변수)
6. acceleration: continuous (가속 : 연속형변수)
7. year: multi-valued discrete (모델연도 : 정수값)
8. origin: multi-valued discrete (정수값)
9. car name: string (unique for each instance) (차종류 이름)

```
# 3. scatterplot after attach command

# use autompg data (lec3_3.R)
# set working directory
setwd("D:/tempstore/moocr/")
car<-read.csv("autompg.csv")
head(car)
```



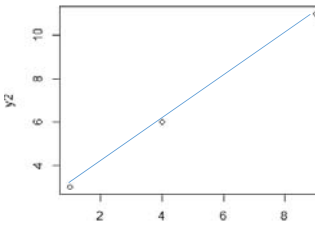
```
Console D:/tempstore/moocr/ ↗
mpg cyl disp hp wt accler year origin
1 18 8 307 130.0 3504 12.0 70 1
2 15 8 350 165.0 3693 11.5 70 1
3 18 8 318 150.0 3436 11.0 70 1
4 16 8 304 150.0 3433 12.0 70 1
5 17 8 302 140.0 3449 10.5 70 1
6 15 8 429 198.0 4341 10.0 70 1
      carname
1 chevrolet chevelle malibu
2 buick skylark 320
3 plymouth satellite
4 amc rebel sst
5 ford torino
```

## 5. 산점도 (scatterplot) 2차원

4-3. R 그래픽 기초 III

### • 산점도 : plot(x, y)

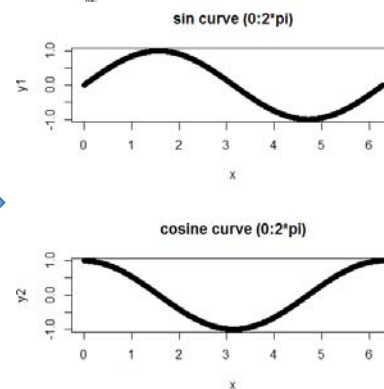
```
# 5-1 simple plot from lec1.r
par(mfrow=c(1,1))
x2<-c(1,4,9)
y2=2+x2
plot(x2, y2)
```



x와 y간의 관계를 보여주는 그래프

```
# 5-2 another simple plot
par(mfrow=c(2,1))
x<-seq(0, 2*pi, by=0.001)
y1<-sin(x)
plot(x,y1, main="sin curve (0:2*pi)")

y2<-cos(x)
plot(x,y2,main="cosine curve (0:2*pi)" )
```



## 5. 산점도 (scatterplot) 2차원

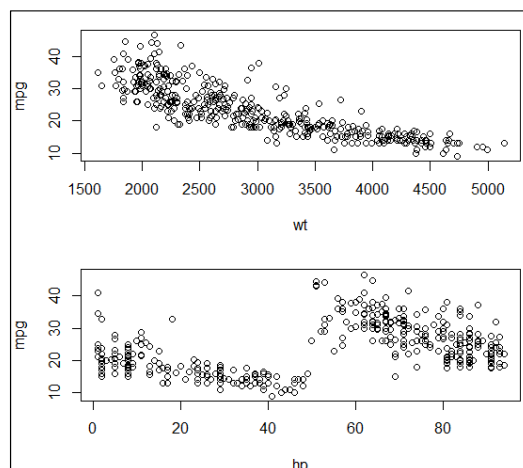
4-3. R 그래픽 기초 III

- wt(차의 무게)과 mpg(연비)간의 산점도 : plot(wt, mpg)
- hp(마력)과 mpg(연비)간의 산점도 : plot(hp, mpg)

```
# scatterplot of autmpg data
# 5-3 autmpg data (relations)
par(mfrow=c(2,1))
plot(wt, mpg)
plot(hp, mpg)
```

산점도에 대한 해석과 설명

- 차의 무게가 무거울수록 연비는 낮다.
- 마력과 연비간의 산점도에서는 두개의 클러스터가 보임(클러스터내에서는 마력이 높을수록 연비가 낮음)



## 5. 산점도 (scatterplot) 2차원

4-3. R 그래픽 기초 III

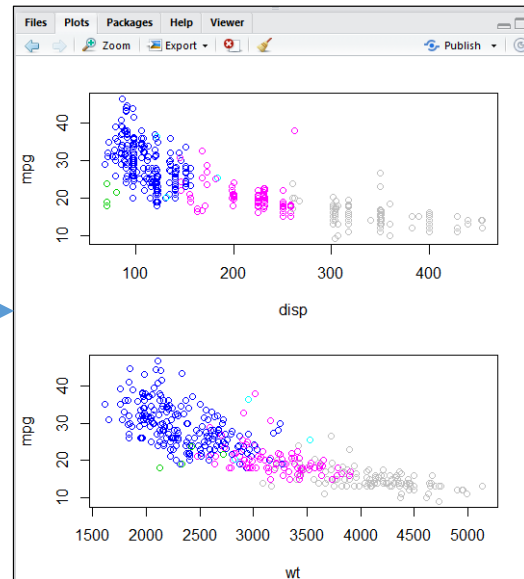
• `plot(x, y, col=as.integer(그룹변수))`

색으로 표시

```
# 5-4 scatterplot coloring group variable
par(mfrow=c(2,1), mar=c(4,4,2,2))
plot(displ, mpg, col=as.integer(car$cyl))
plot(wt, mpg, col=as.integer(car$cyl))
```

autompg 데이터의 cylinder 빈도

```
> table(car$cyl)
 3    4    5    6    8 
 4  204  3   84  103
```



3 cylinder와 5cylinder를 제외하고 분석하면

## 5. 산점도 (scatterplot) 2차원

4-3. R 그래픽 기초 III

• Conditioning plot : `coplot(y~x | z)` z는 factor(그룹)

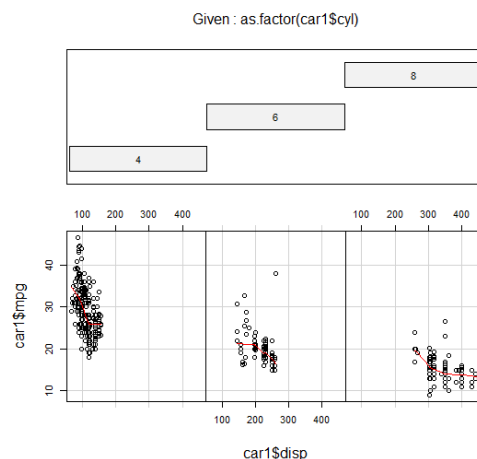
그룹에 따른 (x와 y)간 산점도

Subset 데이터 활용 : 4,6,8cylinder

```
# 5-5 Conditioning plot : separating scatterplot by factor(group) variable
car1<-subset(car, cyl==4 | cyl==6 | cyl==8)
coplot(car1$mpg ~ car1$displ | as.factor(car1$cyl), data = car1,
       panel = panel.smooth, rows = 1)
```

그룹별 산점도

- cylinder에 따른 차이를 보여줌
- 4cyl, 6cyl, 8cyl별로 (배기량과 연비) 간 관계를 구체적으로 해석할수 있음



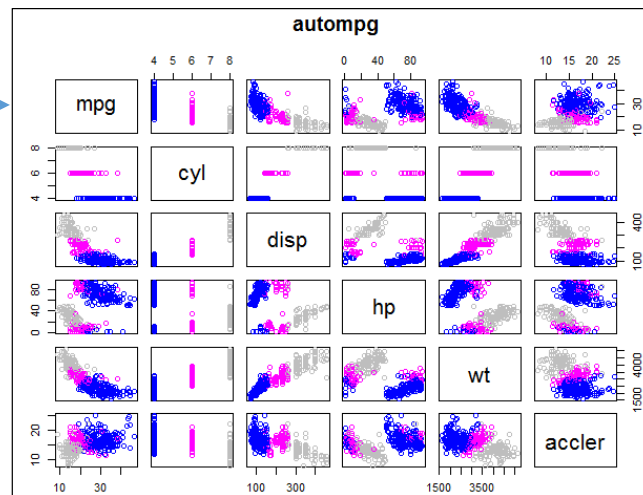
## 5. 산점도 (scatterplot) 2차원

4-3. R 그래픽 기초 III

- pairwise scatterplot : pairs(변수리스트)

Subset 데이터 활용 : 4,6,8cylinder

```
# 5-6 cross-tab Plot to see how explanatory variables are related each other
pairs(car1[,1:6], col=as.integer(car1$cyl), main = "autompg")
```



## 5. 산점도 (scatterplot) : 예측모형

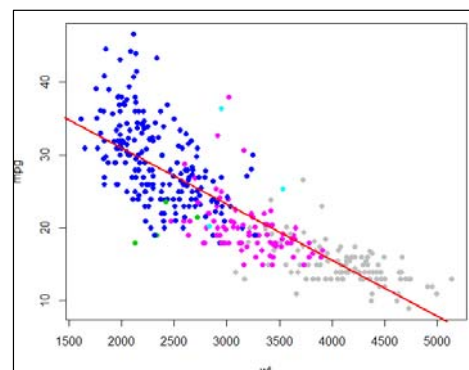
4-3. R 그래픽 기초 III

- 최적 적합 함수 추정 (선형회귀모형, 비선형회귀모형)

lm(y변수~x변수) : 여기서 lm은 linear model(선형모형)의 약자

abline : add line (선을 추가하는 함수)

```
# 5-7 scatterplot with best fit lines
par(mfrow=c(1,1))
plot(wt, mpg, col=as.integer(car$cyl), pch=19)
# best fit linear line
abline(lm(mpg~wt), col="red", lwd=2, lty=1)
```



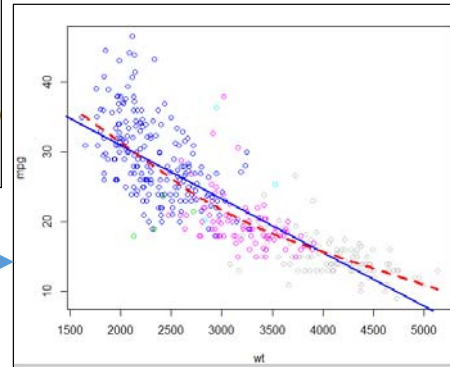
## 5. 산점도 (scatterplot) : 예측모형

### • 최적 적합 함수 추정 (비선형회귀모형, lowess 이용)

lowess : locally-weighted polynomial regression (see the references).

```
# 5-7 scatterplot with best fit lines
plot(wt, mpg, col=as.integer(car$cyl))
# best fit linear line
abline(lm(mpg~wt), col="blue", lwd=2, lty=1)

# lowess : smoothed line, nonparametric fit line
lines(lowess(wt, mpg), col="red", lwd=3, lty=2)
help(lowess)
```



참고문헌 : lowess

Cleveland, W. S. (1979) Robust locally weighted regression and smoothing scatterplots. *J. American Statistical Association* **74**, 829–836.  
 Cleveland, W. S. (1981) LOWESS: A program for smoothing scatterplots by robust locally weighted regression. *The American Statistician* **35**, 54



## Wk4-4 : 그래픽과 레이아웃

### - 그래픽 옵션 -

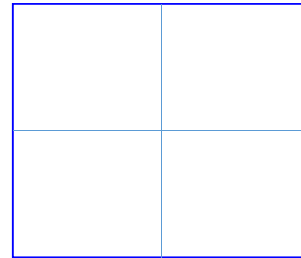
## 그래프의 기본함수

1. 그래프 종류 : `plot()`, `barplot()`, `boxplot()`, `hist()`, `pie()`, `persp()`
2. 그래프 구성시 조정사항 : 점, 선의 종류, 글자크기, 여백조정 등의 옵션을 조정
3. 점그리기: `points()`
- 4 선그리기: `lines()`, `abline()`, `arrows()`
5. 문자출력: `text()`
6. 도형: `rect()`, `polygon()`
7. 좌표축: `axis()`
8. 격자표현: `grid()`



- par() 그래프의 출력을 조정 - 그래프 화면의 분할, 마진, 글자 크기, 색상등 설정
- pty="s" (x축과 y축을 동일비율로 설정, square) pty="m" (최대크기로 설정, maximal)
- legend = c("name1", "name2")
- bty="o" (box type 그래프의 상자모양을 설정) o, l, 7, c, u
- pch=1(default) point character (1=동그라미, 2=세모, ..., 19=채운동그라미)
- lty=(solid가 default) (line type, 1=직선, 2=점선)
- lwd = 1, 2.. (선의 굵기)
- cex=1(default) (character expansion) 문자나 점의 크기, 숫자가 클수록 글자크기 커짐
- mar (아래, 왼쪽, 위쪽, 오른쪽)

(예시) par(mfrow=c(2,2), mar=c(2,2,2,2))



## 선 그리기

- abline(h=위치, v=위치, col="colname")

```
# lec4_4.r : Graphics and layout

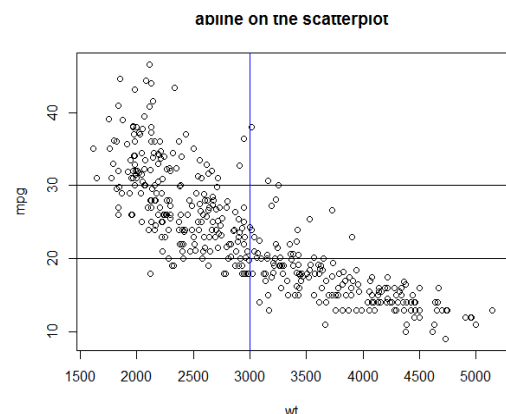
# set working directory
setwd("D:/tempstore/moocr")

# autompg data
car<-read.csv("autompg.csv")
head(car)

attach(car)
```

```
# scatterplot of wt and mpg
plot(wt, mpg, main = "abline on the scatterplot")
# horizontal
abline(h = 20)
abline(h = 30)
# vertical
abline(v = 3000, col="blue")
```

수평선을 y축 위치 20과 30에  
수직선을 x축 위치 3000에, 색은 파란색

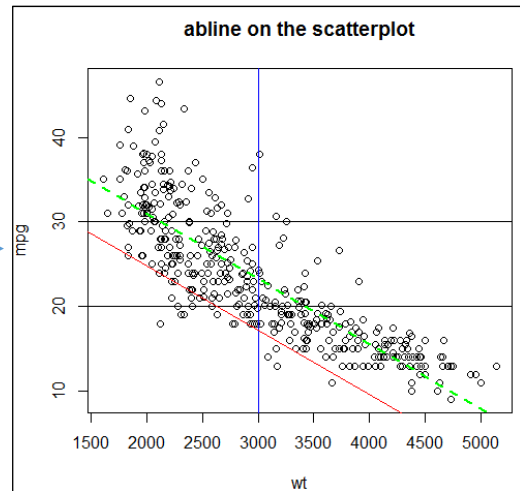


- `abline(절편값, 기울기값, lty=1, lwd=1, col='colname)`

`lty=1` (직선), `lty=2`(점선), `lwd=1` (line width, 숫자 클수록 선 굵어짐)

```
# y = a + bx
abline(a = 40, b = -0.0076, col="red")

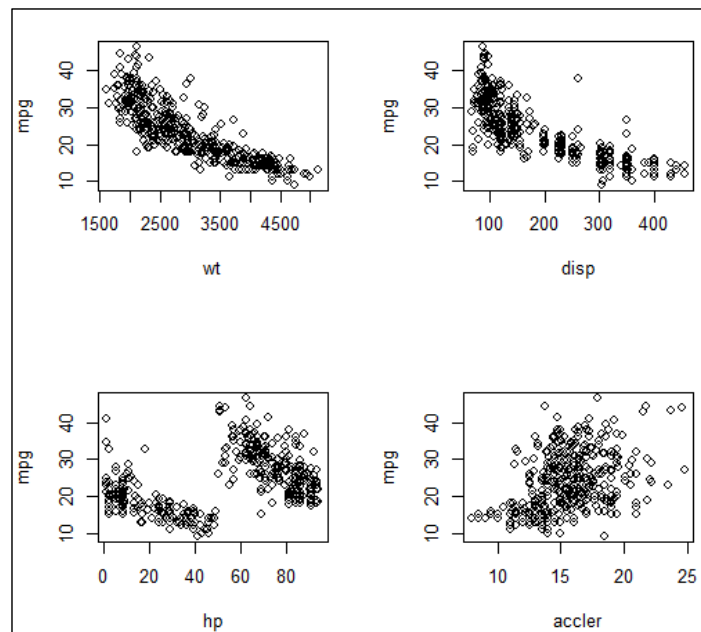
# linear model coefficients, lty (line type)
# linear model (mpg=f(wt))
z <- lm(mpg ~ wt, data = car)
z
abline(z, lty = 2, lwd = 2, col="green")
```



## layout함수

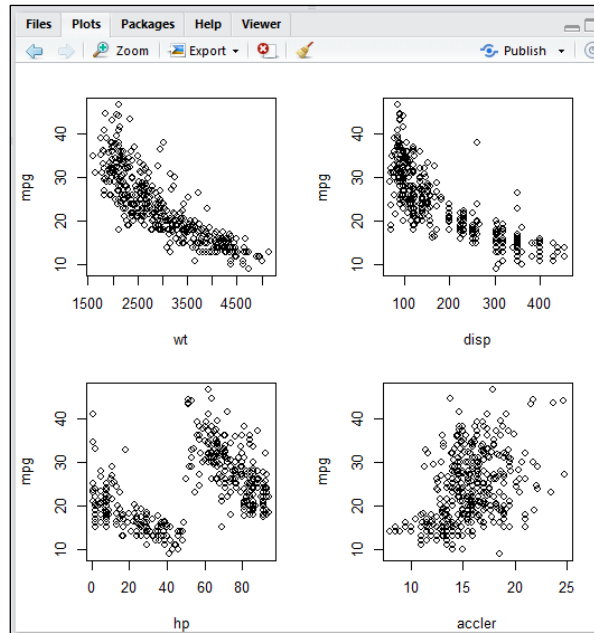
- `par(mfrow=c(2,2))`

```
# layout
# 2*2 mulitple plot
par(mfrow=c(2,2))
plot(wt, mpg)
plot(dis, mpg)
plot(hp, mpg)
plot(accler, mpg)
```



- margin 조정 : mar(아래, 왼쪽, 위쪽, 오른쪽)

```
# 2*2 mulitple plot adjusting margin
par(mfrow=c(2,2), mar=c(4,4,2,2))
plot(wt, mpg)
plot(dis, mpg)
plot(hp, mpg)
plot(accler, mpg)
```

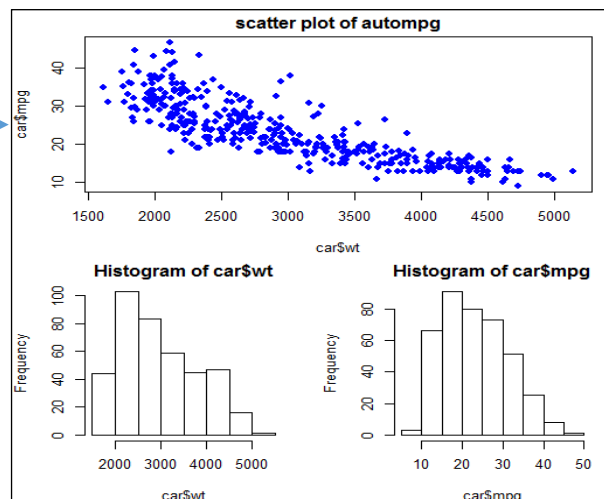
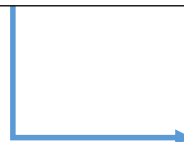


- layout조정

```
# top 1 plot, bottom 2 plot
(m <- matrix(c(1, 1, 2, 3), ncol = 2, byrow = T))
layout(mat = m)
plot(car$wt, car$mpg, main = "scatter plot of autmpg", pch = 19, col = 4)
hist(car$wt)
hist(car$mpg)
```

layout 행렬 m

```
> m
      [,1] [,2]
[1,]     1     1
[2,]     2     3
```

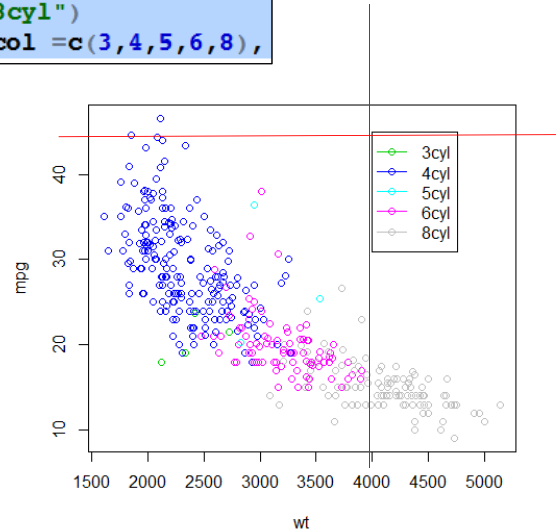


- legend(x축위치, y축위치, legend=범례라벨, pch=1, col=c(번호 혹은 색으로 지정), lty=1)

```
# legend

# 5-4 scatterplot coloring group variable
plot(wt, mpg, col=as.integer(car$cyl))
labels = c("3cyl", "4cyl", "5cyl", "6cyl", "8cyl")
legend(4000, 45, legend = labels, pch = 1, col = c(3, 4, 5, 6, 8),
```

x축위치, y축위치 - 그리면서 조정



## R 그래픽

- 히스토그램과 밀도함수(histogram and density)
- 상자그림(Boxplot),
- 파이차트(Pie chart), 막대그림(Bar plot)
- 산점도 (scatterplot)

- ggplot2를 이용한 그래픽
- 덴드로그램, 애니메이션
- 지도분석 (Map)
- 3D, 히트맵

