# BAYESIAN STATISTICS

## Chapter 2

Instructor: Seokho Lee (이석호)

Hankuk University of Foreign Studies

# 2. R

## 2.1. Introduction to R

**R** is a powerful statistical language that provides various statistical methods (linear/nonlinear modeling, statistical hypothesis test, time-series analysis, multivariate data analysis, etc.) and extensive graphical methods.

**R** is almost similar with **S** language. But **R** is a public software which is developed under the GNU project. Thus, **R** is freely downloaded through the Internet and anyone can contribute to the **R** development.

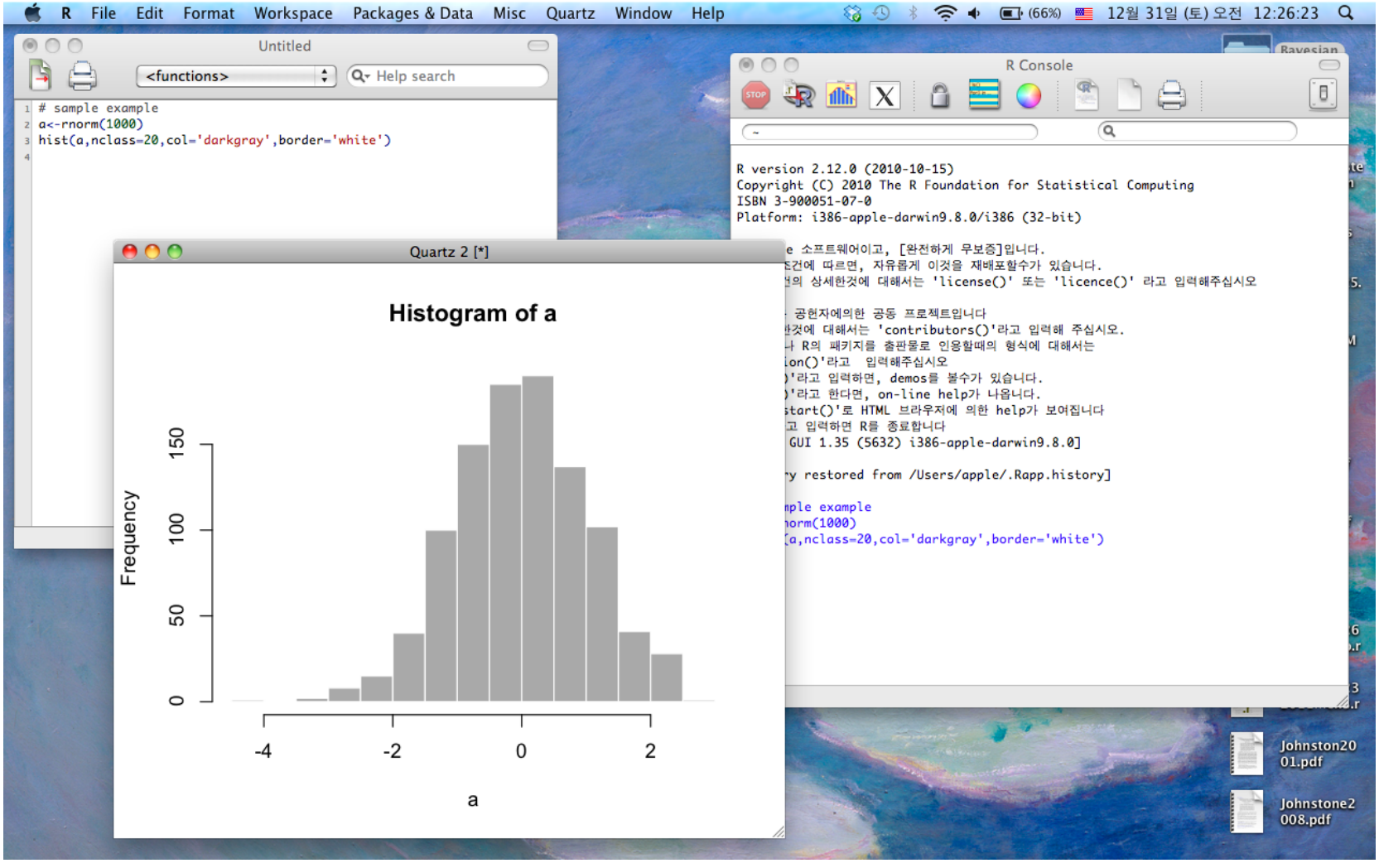
**R** is not a statistical program strictly, but a platform that provides the environment for implementing statistical methodologies through **packages**

The website of "The R project for Statistical Computing" (http://www.r-project.org/) provides the source codes, installation files, and user manuals.

Study the basic manual (click here) by yourself!!!

## 2.2. Basic instruction

### 2.2.1. Start and end R



**Some commands**

- `q()`: quit R session with asking whether you'd like to save workspace image
- `help(persp)` or `?persp`: show the help document on persp() function
- `source("c:/work/sample.R")`: execute R commands in the text file named "sample.R" in the directory "c:/work/"
- `sink("c/work/record.txt")`: send all text outputs in console to the file "c:/work/record.txt". If you type `sink()`, then close the file
- `objects()` or `ls()`: display all objects in the current session
- `rm(x,y,temp,foo)`: remove the objects defined in the session by the names 'x', 'y', 'temp', and 'foo'

## Example

```
> x <- c(1,2,3); y <- c(4,5,6)
> z <- c(x,y,7)
> z
[1] 1 2 3 4 5 6 7
```

## Example

```
> x <- 1:3 ; y <- c(2,2,2)
> x+y
[1] 3 4 5
> x-y
[1] -1  0  1
> x*y
[1] 2 4 6
> x/y
[1] 0.5 1.0 1.5
> x^y
[1] 1 4 9
```

## Example

```
> x <- 1:3 ; y <- c(2,2,2,2,2) # c(1,2,3,1,2) + c(2,2,2,2,2)
> x+y
[1] 3 4 5 3 4
Warning message:
In x + y : longer object length is not a multiple of shorter object length
> y <- c(2,2,2,2,2,2)
> x+y # c(1,2,3,1,2,3) + c(2,2,2,2,2,2)
[1] 3 4 5 3 4 5
> 3-y # c(3,3,3,3,3,3) - c(2,2,2,2,2,2)
[1] 1 1 1 1 1 1
```

## Example

```
> sqrt(-4) # error
[1] NaN
Warning message:
In sqrt(-4) : NANs가 작성되었습니다
> sqrt(-4+0i) # conduct the complex number arithmetic
[1] 0+2i
```

| function | result |
|---|---|
| `log(x)` | the natural logarithm of a vector `x` |
| `exp(x)` | exponentiate a vector `x` |
| `sin(x)` | sine value of a vector `x` |
| `cos(x)` | cosine value of a vector `x` |
| `tan(x)` | tangent value of a vector `x` |
| `sqrt(x)` | square root of a vector `x` |
| `sort(x)` | sort the elements of a vector `x` in an increasing order |
| `max(x)` | the maximum of the elements in a vector `x` |
| `min(x)` | the minimum of the elements in a vector `x` |
| `length(x)` | the length of a vector `x` |
| `sum(x)` | sum of all elements in a vector `x` |
| `prod(x)` | product of all elements in a vector `x` |
| `mean(x)` | average of the elements in a vector `x` |
| `var(x)` | sample variance of the elements in a vector `x` |

Table: some useful arithmetic functions

## Example

```
> x <- 1:5 ; z <- x>3
> z
[1] FALSE FALSE FALSE  TRUE  TRUE
> sum(z) # number of TRUE, i.e., number of the elements in x that are larger than 3
[1] 2
```

| operator | meaning | usage | operator | meaning | usage |
|---|---|---|---|---|---|
| < | less than | x>3 | == | equal | x==y |
| <= | less than or equal to | x>=3 | != | not equal | x!=y |
| > | greater than | x<3 | & | and, intersection | (x>3) & (y<2) |
| >= | greater than or equal to | x<=3 | \| | or, union | (x>3) \| (y<2) |

Table: logical operators

## Example

```
> x <- c(0.1, 0.2, 0.3, 0.4)
> x[3]
[1] 0.3
> x[1:3]
[1] 0.1 0.2 0.3
> x[c(1,3,5)]
[1] 0.1 0.3  NA
```

## Example

```
> y <- c(-3, -2, -1, 0, 1, 2)
> y[y<0] <- -y[y<0]
> y
[1] 3 2 1 0 1 2
```

## Example

```
> z <- array(1:20, dim=c(4,5))
> z
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    5    9   13   17
[2,]    2    6   10   14   18
[3,]    3    7   11   15   19
[4,]    4    8   12   16   20
```

## Example

```
> z <- array(1:20, dim=c(4,5))
> z[2,3]
[1] 10
> i <- array(c(1,2,3,3,2,1), dim=c(3,2))
> i
     [,1] [,2]
[1,]    1    3
[2,]    2    2
[3,]    3    1
> z[i]
[1] 9 6 3
```

## Example

```
> matrix(c(1:8), 2, 4)
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
> matrix(3,2,3)
     [,1] [,2] [,3]
[1,]    3    3    3
[2,]    3    3    3
```

## Example

```
> x <- c(1,2,3); y <- c(4,5,6)
> A <- cbind(x,y)
> A
     x y
[1,] 1 4
[2,] 2 5
[3,] 3 6
> A[1,2]
y
4
> B <- rbind(x,y)
> B
  [,1] [,2] [,3]
x    1    2    3
y    4    5    6
```

## Example

```
> a <- c(1,2,3); b <- c(1,2,3)
> a%o%b
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    2    4    6
[3,]    3    6    9
```

## Example

```
> A <- matrix(0,3,2)
> A
     [,1] [,2]
[1,]    0    0
[2,]    0    0
[3,]    0    0
> t(A)
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
```

## Example

```
> dnorm(0, mean=0, sd=1)
[1] 0.3989423
> pnorm(0, mean=0, sd=1)
[1] 0.5
> qnorm(0.5, mean=0, sd=1)
[1] 0
> rnorm(5, mean=0, sd=1)
[1]  0.3931224  0.1660678 -1.0957104  1.2508052 -0.0653233
```

| distribution | R name | arguments | distribution | R name | arguments |
|---|---|---|---|---|---|
| beta | beta | shape1, shape2, ncp | binomial | binom | size, prob |
| Cauchy | cauchy | location, scale | chi-square | chisq | df, ncp |
| exponential | exp | rate | F | f | df1, df2, ncp |
| gamma | gamma | shape, scale | geometric | geom | prob |
| hypergeometric | hyper | m, n, k | log-normal | lnorm | meanlog, sdlog |
| logistic | logis | location, scale | negative binomial | nbinom | size, prob |
| normal | norm | mean, sd | Poisson | pois | lambda |
| t | t | df, ncp | uniform | unif | min, max |
| Weibull | weibull | shape, scale | Wilcoxon | wilcox | m, n |

Table: Probability distributions in R

## Example

```
> x <- 1:4
> if ( sum(x) > 12 ) {
+ y <- 1
+ z <- 0
+ } else {
+ y <- 2
+ z <- 1
+ }
> y
[1] 2
> z
[1] 1
```

## Example

```
> x <- c(1,2,3,4); y <- rep(2,4)
> x < y
[1]  TRUE FALSE FALSE FALSE
> (x < y) & (x <= y) # logical operation is applied elementwisely
[1]  TRUE FALSE FALSE FALSE
> (x < y) && (x <= y) # logical operation is applied only to the first element
[1] TRUE
```

## Example

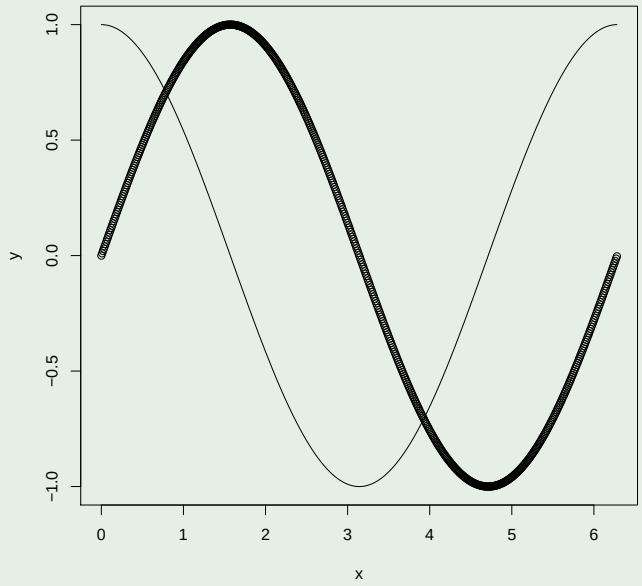
```
> s <- 0; p <- 1
> for ( i in 1:10 ) {
+ s <- s+i
+ p <- p*i
+ }
> s
[1] 55
> p
[1] 3628800
```

## Example

```
> fcn <- function(x,y)
+ {
+ ans <- sqrt(x*x+y*y)
+ return(ans)
+ }
> fcn(3,4)
[1] 5
> ans
에러:오브젝트 'ans'가 없습니다
```

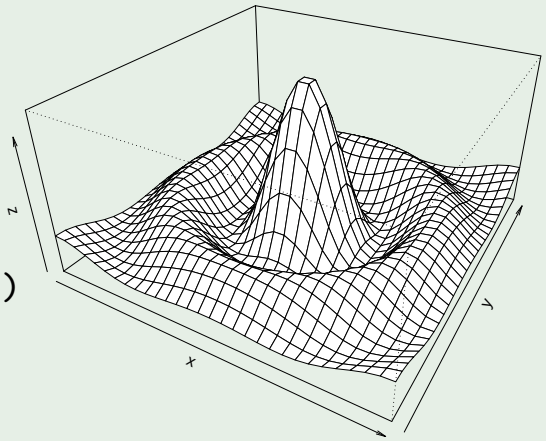
## Example

```
> x <- seq(from=0, to=2*pi, by=0.01)
> y <- sin(x); z <- cos(x)
> plot(x,y)
> lines(x,z)
> dev.copy2pdf(file='fig2-1.pdf')
```



## Example

```
> x <- seq(-10,10,length=30); y <- x
> f <- function(x,y){
+ r <- sqrt(x^2+y^2); 10*sin(r)/r }
> z <- outer(x,y,f)
> persp(x,y,z,theta=30,phi=30,expand=0.5)
> dev.copy2pdf(file='fig2-2.pdf')
```



# Example

```
> par(mfrow=c(1,3))
> x <- seq(from=0, to=2*pi, by=0.01)
> y <- sin(x)
> plot(x,y,type='l',main='sine curve')
> y <- cos(x)
> plot(x,y,type='l',main='cosine curve')
> x <- seq(from=0,to=1.5,by=0.01)
> y <- tan(x)
> plot(x,y,type='l',main='tangent curve')
> dev.copy2pdf(file='fig2-3.pdf')
```