STAT 3027

2. Vector

① When are [brackets] used and when are (parathesed) used? ↓ index ↓ combine

② which methods are there to remove names?

③ what are the difference between <- = and == ?

★ V <- c(2.71, 5, 3.14)

combination

length(V)

V

index	value
i	V[i]
1	2.71
2	5
3	3.14

words <- c("tree", "and", "chainsaw")

words[1]

★ Basic Types. Specifying Constants

numeric: (real number) 3.14e2 = 3.14×10^2

character: = 314

Escape sequence.

\" (double quote)

String = paste(sep = " ", " ", " ")

cat (string)

cat (sep = " ", " ", " ")

logical: TRUE and FALSE

Sum (V>3)

⇒ 2

Vector (mode = "logical", length = 0)

↳ creates a vector of the given mode & length

[logical 0]

=5

FALSE --- FALSE

numric =5

0 0 0 0 0

char

Change a vector's type

a(.numric())

as. character ()

as. logical

as. numeric (V>3)

integer

complex

raw

W= c ("34", "12", "45")

W

"34" "12" "45"

Sum (W)

↳ error.

W. numbers = as. numric(W)

W. numbers

34 12 45

sum(w.numbers)

91

sum(as.numeric(w))

91

★ Names attribute and a few functions

name(x) gets or set.

name(v) = cc("e", "five", "pi")

v

e	five	pi
2.71	5.00	3.14

name(v) = NULL # remove names

== y = c(burger = 2.50, fries = 1.50)

y

burger	fries
2.5	1.5

names(y) = NULL

2.5 1.5

A FEW Functions!

x <- c(2, 11, 15, 17)

Operators

%/% quotient

%% remainder

sqrt(sum((x - mean(x))^2) / (n-1))

seq (10, 15, by=2)
c(10, 12, 14)
seq(10, 15, length.out=3)
10, 12.5, 15.0

matching: %in%
3 %in% c(2,7)
FALSE TRUE FALSE

☆ Indexing

4. x[v]
↓
negative ⇒ exclude.
indice = which(x < 14)
index 의 약자

x[indices]
x[x<14]
☆ which can be omitted.
~~x[(x%%2)==0]~~

3. Vector (continued) and List

SORTING Functions
sort(x, decreasing = FALSE)
x = c(12, 11, 16, 11)
⇒ 11, 11, 12, 16

r = rank(x, ties.method = "average")

第几大

"first"

v = order(x, ..., decreasing = FALSE)

the index of smallest elements in x.

x[v] is sorted.

Structure, summary, quantile

str(object)

summary(object)

quantile

r = quantile(x, probs = c(0.25, 0.75))

NULL and NA are special values

names(x)

x[3] = NA

sum(x) = NA

sum(x, na.rm = TRUE)

is.null(), is.na()

File input/output

scan(file = "", what = numeric(1))

write(x, file = "data")

List:

not necessarily of the same type

can be recursive

unlist(y, use.names=FALSE)

4. Data Frame work

(R's fundamental data structure)

A data frame is (≈) a list of vectors

Categorical Variables

Factor =). a vector of categorical values

factor(x, levels, labels = levels)

table(...)

Manipulation Examples

m[,1]

row column

m[1:3, 1:3]

dim(m)

n.rows = dim(m)[1]

n.cols = length(m)

head(m)

tail(m)

rownames(m)

mtcars[30,]

m = median(mtcars)

sorted = m[order(mtcars$cyl, mtcars$disp)]

write.table(x, file="x.txt", sep=",")

read.csv( )

Formula.

5. Graphics

1. Comon parameters

formula, data.

main, sub, xlab, ylab

xlim, ylim,

pch

col

2. Numeric data

boxplot (x)

boxplot (mtcars$mpg, main = "Car mileage", ylab = "
ylim = c (0, 40))

by group

y軸范围

boxplot (mpg ~ factor (cyl), data = , xlab.
ylab =);

Stripchart (x, method = "overplot")

Strip chart (x ~ g)

overplot

jitter. > duplicate

hist (x, breaks = "Sturges", freq = NULL)

density

☆ cex. axis = 2

调尺寸 cex. lab = 2.

plot (x, y)

poi

x = 1:5; y = 2*x; plot(x, y)

? points

lines (x = c(1, 3, 5, 7, 9)

y = c (8, 1, 4, 1, 8)

col = "red")

plot(density(x))

pairs (x)

curve (expr = x*sin(1/x), from = -pi/6, to = pi/6, n=200)

curve (expr = x*1, add = TRUE, col = "red")

Legends

legend (x, y, legend, col = par(no.readonly))

legend ("top", legend = c("x*sin(1/x)", "x"),
col = c("black", "red"), lty=c(1,1))

? legend

expression(..)

? plot.match

e.g. plot(x, y)

stem (plot())

categorical data:

barplot (height, name s, arg.l=NULL)

Any Variable mosaicplot (x)

count = table( );

barplot (counts, name.arg = c( ))

Multiple figures

matrix(data, nrow, ncol, byrow = FALSE)

layout(mat)

layout.show(3)

Write graphical output to a file

dev.off(b)

x = rnorm(100)

pdf("hist.pdf")

plot.hist

dev.off(2)

Test and Intervals

One mean or the Difference of Two mean

out = t.test(x, y = NULL, alternative = "two.sided", mu = 0, conf.level = .95) to test mu = 0

$parameter: degree of freedom

$statistic: Student's test statistic

$p.value

$conf.int

$estimate

str(out)

output value

two means

F Test for Equality of Variances

out = var.test(x, y, ratio = 1, alternative = "two.sided", conf.level = .95)

Ho $\frac{\sigma_x^2}{\sigma_y^2}$ = ratio

$ parameter

$ statistic $F = \frac{s_x^2}{s_y^2}$

$ p.value

$ conf.int

chi-Squared Tests

Goodness of fit

counts <- c(...) probs <- c(...)

out = chisq.test(x = counts, p = probs)

$ parameter

$ statistic

Independence / Homogeneity

$ expected: expected counts under Ho

chisq.test();

One Proportion or the Difference of Two Proportions

dist = prop.test(x, n, p=p1, alternative = "two.sided",
conf.level = .95)

2-test $H_0: p = p_0 = p$

$\hat{p}$ estimate $\hat{p} = x/n$

2-vectors x and n

$H_0: p_1 - p_2 = 0 \quad [p_1 = p_2]$

$\hat{p} = \frac{x_1 + x_2}{n_1 + n_2}$

7. Regression

1. Simple linear Regression

$y = mx + b$

cor(x, y) / correlation ⇒ 数

lm(y~x, data)

A = cor(x) ⇒ matrix A

cor(x[,i], y[,])

2. lm(y ~ x, data)

anova(m)

m$coefficients[1]

abline (a, b) $y = a + bx$

= abline (reg=m)

abline (a = mean (car$ dist)

horizontal

---

predict (model)

y.hat = predict (m, newdata=d)

points (x=d$speed, y=y.hat, pch=19, cex=3)

---

plot (m$ fitted. values, m$residuals)

abline (0, 0)

QQplot.

N(μ= mean(residuals), σ= sd(residuals))

qqline (x)

e.g x = rnorm (n=100) ; qqnorm (x); qqline(x)

w = rexp(100), random qqv

or use plot (m)

---

multiple Linear Regression

m1 = lm(y ~ x1+x2+ ... , data=)

confint (m,

8. Simulation

① repeatable

set.seed(seed)

set.seed(0); a = norm(1)

set.seed(0); b = norm(1)

② Repeat calculation n times,

replicate(n, expr)

③ ? distribution

$\bar{X} \approx N(\mu, \frac{\sigma^2}{n})$

t = replicate(N, {x = norm(n, mean =

symbols text(x=col, y=row, label= use.chr, cex=4)

identify(xx, yy, n=1, plot=FALSE)

unique( )%in%

function

parity = ifelse((x%%2)==0, "even", "odd")

2. loop: while( ){ } repeat { } if( ){break}

Loop one more times: repeat { expression
if (condition) {
break
}
}

input: xx = scan(what = character / numeric(), n=1, quiet=TRUE)

skip: if (cond) {next}

3. Apply Function

tapply(X= , FUN=) treating data frame as a list of vectors

☆ sapply(X= , FUN=) ⇒ vector, matrix, array

multiple argument apply | mapply(FUN, x, y, ...) take the several vectors in and applies FUN to all first elements and then to all second elements

apply(X= , MARGIN, FUN, ...) =1 rows =2 columns

tapply(X= , INDEX= , FUN= ...) subset of vector

tapply(X = mtcars$mpg, INDEX = mtcars$cyl, FUN=) 分类!

tapply(X= , INDEX = list(mtcars$cyl, mtcars$gear), FUN=)

4 Matrix m = matrix(data = vector, nrow= , ncol= , byrow = TRUE, extra probs=c(

dimnames = list(c(colname), c(rowname)) ↓数字 replicate(,)

cbind(matrix,) rbind(matrix,)

m[row(m) == col(m)] main diagonal

m[row(m) + col(m) == r+c] diagonal through (r, c)

m t + reverse.

A * B element-wise product A %*% B matrix product

solve(a=A, b=b) $A\vec{x} = \vec{b}$ A %*% x % x — matrix-vector product

5. pattern

grep(pattern, x, ignore.case=FALSE, value=FALSE) index of elements of char

grep(pattern= ... TRUE! ⇒ values

sub(pattern, replacement = , x=) a copy of x after replacing 1st occurre of pattern with replacement

gsub(" ", " ", x, TRUE) all

\d 0123456789 \w is a 不与匹配

gsub(pattern="[aeiou]", replacement="", x=a) # strip vowels (letter)

gsub(pattern="[^aeiou]", replacement="", x=a) # strip non-vowels

- ^ matches the beginning of a line ($ matches the end) e.g

- grep(pattern="^r", x=a) grep(pattern="^r", ignore.case=TRUE, x=a)

- \\< matches the beginning of a word (\\>) end

e.g grep(pattern="e\\>", x=a)

repetition. {n} exactly n times {n,} n or more times * ({0,1}, + {1,}) {n,m} n-m times inclusive ?⇒{0,1} or "optional"

e.g. grep(pattern="\\d{4}$", x=a) 4 digits, end of lines

grep(pattern="□\\d{4}$", x=a) # space, 4 digits, end of line

grep(pattern=" \\d{4,5}$", x=a) # space, 4 or 5 digits, end of lie

\N refers to what the Nth enclosed expression matched

sub(pattern="(\\w*), (\\w+) +(\\d+)(\\w*)", replacement="\\2, \\1, \\4 \\3", x=)

| means or grep(pattern="Joe|Jack", x=a) grep(pattern="J(o|a)", x=a)

\\ ⇒ ?regex

-Splitting strings strsplit(x=a, split=",") strsplit(x=a, split=" +") =)拆分所有的space

⇒ write(x=csv, file="") d=read.csv("", header=FALSE, col.names=c( ))

6. ggplot ① qplot(data=, x= col, y=, geom="boxplot")

① ggplot 1. gg1<- ggplot(data=trees, aes(x=, y=)) coord_flip() flip

gg1 ⇔ + geom_boxplot() + geom_jitter() + violin()

2. +xlab("") +ylab("") +labs(title="") + theme(plot.title=element_text(size=22, vjust=2, hjus

3. ggplot(aes(x, y, fill= x or y) + guides(fill=FALSE)

4. reorder species by mean ggplot(data, aes(x=reorder(x, y, fun=mean), y=,)

5. outlier shape +geom_boxplot(outlier.shape=21) 6. change to log scale +scale_y_log10(labels=function(b){paste(b,"in.")}) 7. color brewer -scale_fill_brewer(palette="Set1")

geom_density() geom_point() + geom_smooth() + geom_line() aes(color=factor( ) shape=factor() +facet_grid(.~) coincide with +scale_x_sqrt(limits=c(0,50))

7. Web scrap TEAM <- "http:__" lines<-readLines(TEAM)

ten.lines<-grep(pattern="", x=lines, value=TRUE) team.names<-sub(x=ten.lines, pattern="(*) replace="\\1") link<-paste0(__,__,__) table<-readHTMLTable(link)

table[[3]][,] [ ] as.numeric(as.character( ))

Vector sort( ) order x[order(x)] na.rm=TRUE list[[ ]]

write.table(x, file="",...) table=read.table(...)

CSV .csv

stopifnot(function( ) == "")

turn a vector into a character ⇒ paste(v, sep="", collapse="")

⇒ =data.frame(,)