This course was developed as a part of the VLIR-UOS Cross-Cutting project s:

- Statistics: 2011-2016, 2017.
- Statistics: 2017.
- Statistics for development : 2018-2020.

The >eR-Biostat initiative

Making R based education materials in statistics accessible for all

An introduction to R: Short Version (2017)

Part 2: basic programming

Developed by

Dan Lin (Hasselt University) and Ziv Shkedy (Hasselt University)

LAST UPDATE: 15/10/2017

Visit us on Facebook ER-BioStat

GitHub https://github.com/eR-Biostat

Email: erbiostat@gmail.com

twitter @erbiostat

Overview

1. Basic programming in R: objects in R
2. Reading external datasets
3. Programming in R: a for loop
4. Programming in R: user functions
5. Application of a for loop: bootstrap.

Chapter 1
Basic programming
Objects in R

Simple objects

Assign the value of 5 to the R object x

```
> x <- 5
> x
[1] 5
> x^2
[1] 25
> x + 6
[1] 11
```

Vectors

A function in R that creates a vector: **c()**

```
> x<-c("A","A","A","A","B","B","B","B")
> x
[1] "A" "A" "A" "A" "B" "B" "B" "B"
```

```
> y<-c(10,11,9,15,3,5,7,2)
> y
[1] 10 11  9 15  3  5  7  2
```

```
> y[1]
[1] 10
> y[2]
[1] 11
```

The first element in the vector y

The second element in the vector y

Index vectors

```
y[x=="A"]
```

All the elements in y
for which x=A

```
> ya<-y[x=="A"]
> ya
[1] 10 11  9 15
```

```
> yb<-y[x=="B"]
> yb
[1] 3 5 7 2
```

```
> tapply(y,x,mean)
    A     B
11.25  4.25
```

Data frames

A data structure which contains more than 1 object.

Objects can be numeric objects and character objects

```
> z<-data.frame(x,y)
> z
  x  y
1 A 10
2 A 11
3 A  9
4 A 15
5 B  3
6 B  5
7 B  7
8 B  2
```

The $

The object x in z

```
> z$x
[1] A A A A B B B B
Levels: A B
> z$y
[1] 10 11  9 15  3  5  7  2
```

Matrix

```
> w<-c(1,2,40,2,3,9,200,4,6000)
> matw<-matrix(w,3,3)
> matw
     [,1] [,2] [,3]
[1,]    1    2  200
[2,]    2    3    4
[3,]   40    9 6000
```

Rows and columns

X_{ij}=x[i,j]

```
> w1<-matw[1,]
> w2<-matw[,2]
> w1
[1]    1    2 200
> w2
[1] 2 3 9
```

The matrix reloaded

```
> matw+10
     [,1] [,2] [,3]
[1,]   11   12  210
[2,]   12   13   14
[3,]   50   19 6010
```

```
>
[1]    1    3 6000
```

The inverse matrix

```
> solve(matw)
             [,1]        [,2]          [,3]
[1,] -0.687854189  0.39056517  0.0226680962
[2,]  0.453361924  0.07658141 -0.0151631184
[3,]  0.003905652 -0.00271864  0.0000382907
```

```
> solve(matw)%*%(matw)
             [,1]          [,2]          [,3]
[1,] 1.000000e+00  9.714451e-17  1.998401e-15
[2,] 5.551115e-17  1.000000e+00 -8.104628e-15
[3,] 4.336809e-19 -4.336809e-19  1.000000e+00
```

Example: data frame

```
> x<-c(25,36,21)
> gender<-c("M","M","F")
> data.frame(x,gender)
   x gender
1 25      M
2 36      M
3 21      F
```

Example: an R object of a data frame

```
> x<-c(25,36,21)
> gender<-c("M","M","F")
> xdat<-data.frame(x,gender)
> xdat
   x gender
1 25      M
2 36      M
3 21      F
> xdat$gender
[1] M M F
Levels: F M
```

Practical session

- Create the folowig data frame:

A	100
B	99
C	105
D	35
E	0
F	250

Chapter 2
Reading external datasets

Read an external file (text file)

```
> spwh3<-read.table('c:\\projects\\wseda\\spwh3.txt',
                     header=FALSE,na.strings="NA", dec=".")

> dim(spwh3)
[1] 60  4

> spwh3<-data.frame(spwh3)
> names(spwh3)<-c("id","y","x1","gender")
```

Read an external file (csv file)

```
> gsw.ts <- read.csv("C:/projects/NBA/GSW_TS.csv",header = FALSE,sep =";")
```

The data

```
> spwh3
   id         y x1 gender
1   1 10.111368  1      0
2   2  9.948930  1      0
3   3 10.322560  1      0
4   4 10.241052  1      0
5   5  9.911427  1      0
6   6  9.357969  1      0
7   7 10.649141  1      0
8   8 10.150197  1      0
9   9  9.403218  1      0
10 10  8.027072  1      0
11 11 20.020056  1      1
```

The sleep data in R

```
> sleep
   extra group ID
1    0.7     1  1
2   -1.6     1  2
3   -0.2     1  3
4   -1.2     1  4
5   -0.1     1  5
.     .      .  .
14   0.1     2  4
15  -0.1     2  5
16   4.4     2  6
17   5.5     2  7
18   1.6     2  8
19   4.6     2  9
20   3.4     2 10
```

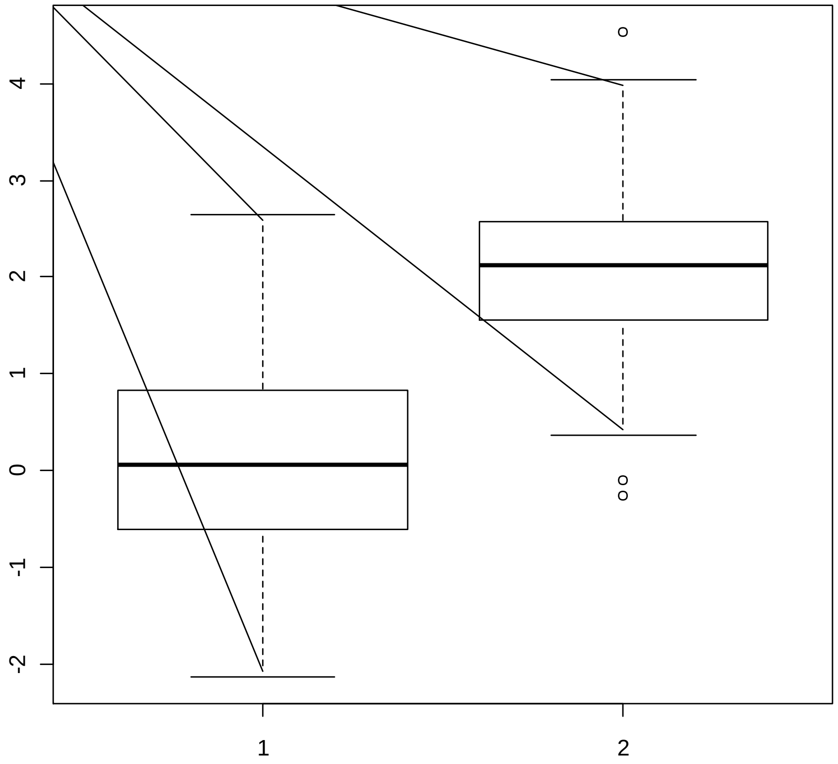
Two samples t-test

```
> y1<-spwh3$y[spwh3$gender==0]
> y2<-spwh3$y[spwh3$gender==1]
> t.test(y1,y2)

        Welch Two Sample t-test

data:  y1 and y2
t = -9.1428, df = 58, p-value = 7.715e-13
alternative hypothesis: true difference in means is not
equal to 0
95 percent confidence interval:
 -12.229889  -7.836547
sample estimates:
mean of x mean of y
 14.99933  25.03254
```

Two samples t-test




```
> y1<-rnorm(100,0,1)
> y2<-rnorm(57,2,1)
> boxplot(y1,y2)
```

Two samples t-test

```
> t.test(y1,y2)

        Welch Two Sample t-test

data:  y1 and y2
t = -14.2203, df = 126.176, p-value < 2.2e-16
alternative hypothesis: true difference in means is not
equal to 0
95 percent confidence interval:
 -2.290641 -1.730980
sample estimates:
 mean of x  mean of y
-0.0063866  2.0044240
```

R object for the output

```
> t.t<-t.test(y1,y2)
```

```
> summary(t.t)
            Length Class  Mode
statistic   1      -none- numeric
parameter   1      -none- numeric
p.value     1      -none- numeric
conf.int    2      -none- numeric
estimate    2      -none- numeric
null.value  1      -none- numeric
alternative 1      -none- character
method      1      -none- character
data.name   1      -none- character
```

R object for the output

```
> t.t

        Welch Two Sample t-test

data:  y1 and y2
t = -14.2203, df = 126.176, p-value < 2.2e-16
alternative hypothesis: true difference in means is not
equal to 0
95 percent confidence interval:
 -2.290641 -1.730980
sample estimates:
 mean of x  mean of y
-0.0063866  2.0044240
> t.t$p.value
[1] 5.570543e-28
> t.t$statistic
        t
-14.22034
```

Practical session

- Create the following text file:

A	100
B	99
C	105
D	35
E	0
F	250

and read it to R as an external file

Chapter 3
Programming I: A for loop

A for loop

```
for(i in 1:B)
{
```

Here you ask from R to do the same thing B times…..

```
}
```

Generate 1000 samples from N(2,1)

A random sample from N(2,1).
Sample size=10.

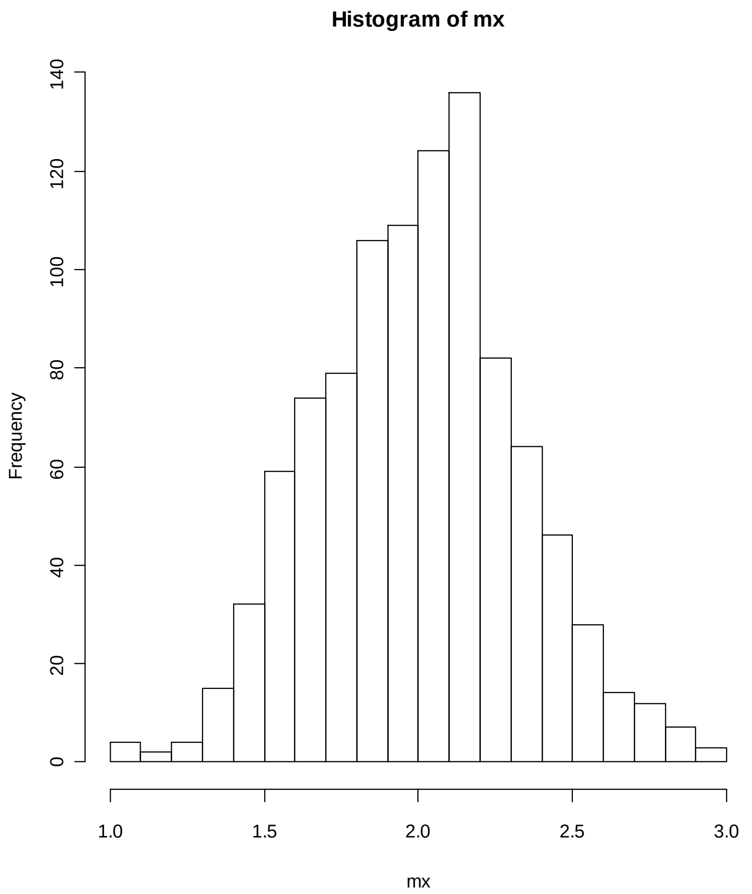
$$X_i \sim N(\mu = 2, \sigma = 1)$$

```
> x<-rnorm(10,2,1)
> x
 [1]  2.1531462  2.4426189  0.8080064  1.4051178  1.9392356  0.6466574
 [7]  0.7519918 -0.1097367  2.3338487  3.7598694
> x<-rnorm(10,2,1)
> x
 [1] 2.9694328 1.1065506 1.5612572 0.3904008 1.6890423 3.7319756 0.9026146
 [8] 1.7763012 2.4356002 0.9643299
> x<-rnorm(10,2,1)
> x
 [1] 2.1888795 2.6353313 2.7131707 1.2311123 2.8258664 0.8101126 2.1533630
 [8] 2.9126222 0.4085356 1.5586004
> mean(x)
[1] 1.943759
```

Generate 1000 samples from N(2,1)

`mx`: A vector of 1000 numbers

```
> mx<-c(1:1000)
> for(i in 1:1000)
+ {
+ x<-rnorm(10,2,1)
+ mx[i]<-mean(x)
+ }
> hist(mx,nclass=25)
```



A random sample from N(2,1).
Sample size=10.

Repeat this 1000 times

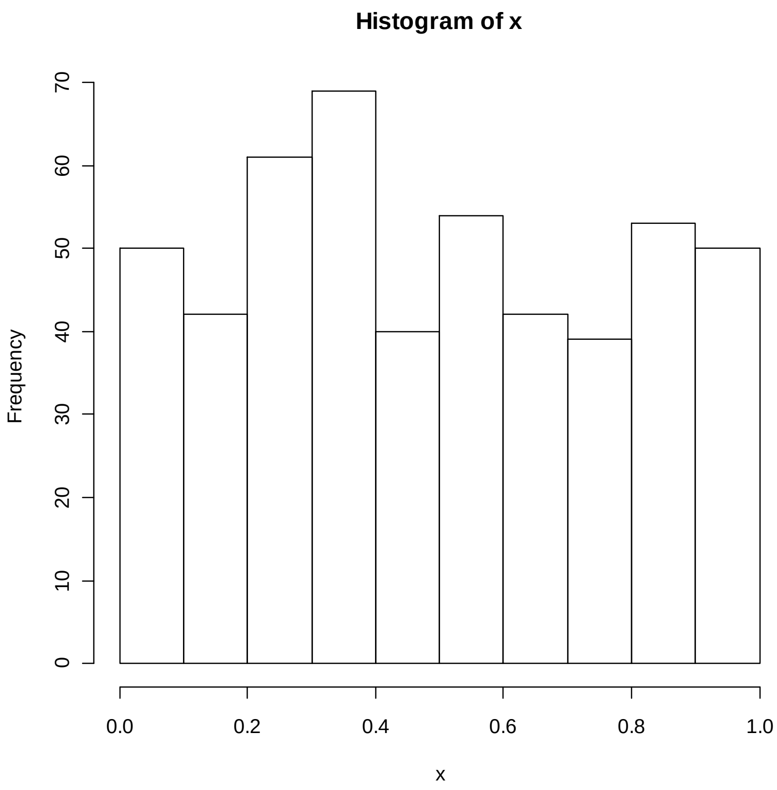
Example: distribution of the minimum in uniform distribution

- Generate 1000 samples (n=50) from U(0,1).
- Calculate the minimum of each sample.
- Estimate the density of the minimum.

Example: distribution of the minimum in uniform distribution

- Generate 1000 samples (n=50) from U(0,1).
- Calculate the minimum of each sample.

```
> x<-runif(500,0,1)
> hist(x)
> min(x)
[1] 0.004631357
```



Example: distribution of the minimum in uniform distribution

- Estimate the density of the minimum.

```
for(i in 1:B)
{
```

Generate 1000 samples (n=50) from U(0,1).
Calculate the minimum of each sample.

```
}
```

Example: distribution of the minimum in uniform distribution

- Estimate the density of the minimum.

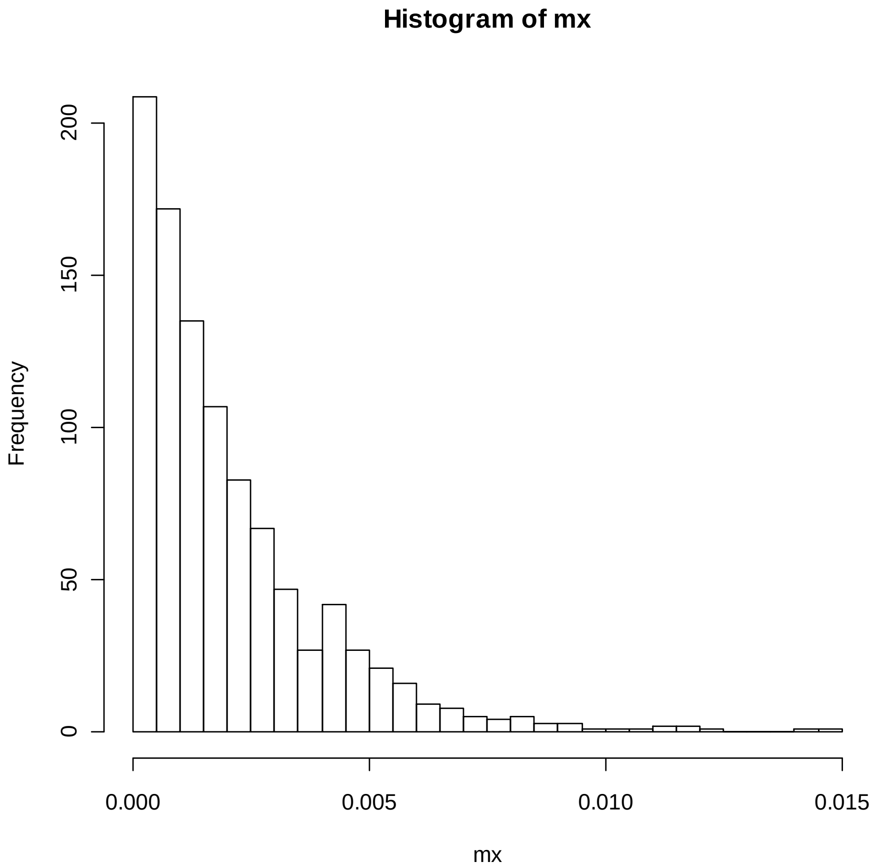
```
for(i in 1:B)
{
```
Generate 1000 samples (n=50) from U(0,1).
Calculate the minimum of each sample.
```
}
```

```
> mx<-c(1:1000)
> for(i in 1:1000)
+ {
+ x<-runif(500,0,1)
+ mx[i]<-min(x)
+ }
```

Example: distribution of the minimum in uniform distribution

- Estimate the density of the minimum.

```
> mx<-c(1:1000)
> for(i in 1:1000)
+ {
+ x<-runif(500,0,1)
+ mx[i]<-min(x)
+ }
>hist(mx)
```



Practical session

- Make a for loop that print your name 500 times.

Chapter 4

Programming in R II: User functions

Generate s random sample pf size 1000 from N(0,3)

```
> x<-rnorm(100,0,3)
> mean(x)
[1] 0.3080260
> median(x)
[1] 0.4176008
> quantile(x)
        0%        25%        50%        75%       100%
-5.9877043 -1.7844439  0.4176008  1.5712923  8.5930491
```

A user function: general form

```
function name<-function(x)
{
```

R commands (what do you what that the function will do for you……)

```
}
```

A user function: general form

```
function name<-function(x)
{



}
```

Input: a sample

Output:

- Descriptive statistics: mean, median, quantiles
- Graphical output: histogram

R functions for the output:

```
mean()
median()
quantile(x)
hist()
```

A user function: general form

A small program to produce the output:

```
mean.x<-mean(x)
med.x<-median(x)
quantile(x)
hist(x)
```

See slide 39.

A user function: example

```
fch20<-function(x)
{
mean.x<-mean(x)
med.x<-median(x)
q.x<-quantile(x)
hist(x)
return(mean.x,med.x,q.x)
}
```

The input

The output

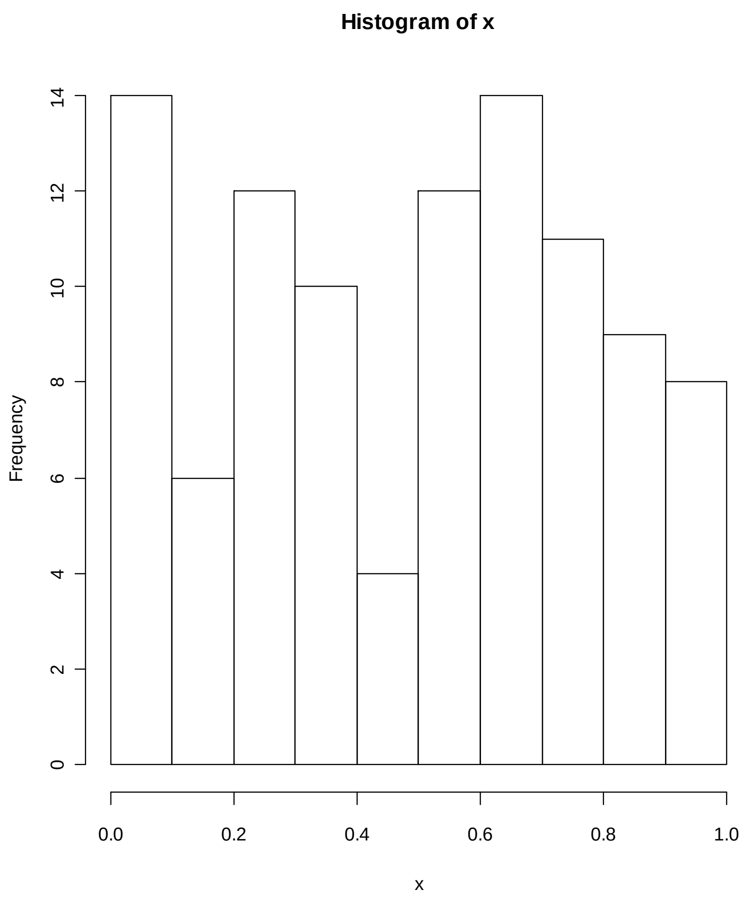
A user function: output

```
> z<-runif(100,0,1)
> fch20(z)
$mean.x
[1] 0.4947539

$med.x
[1] 0.5291341

$q.x
        0%        25%
   50%        75%       100%
0.01240262 0.24212404
   0.52913405 0.72482479
   0.98413912

Warning message:
In return(mean.x, med.x, q.x) :
   multi-argument returns are
   deprecated
>
```



Practical session

- Write a function which receive a numerical vector as an input and calculate the mean of the vector.

Chapter 5:
Application : the for loop

The bootstrap estimate of the standard error for the mean

The observed data

A sample of 10 observations:

```
> x <- c(11.201, 10.035, 11.118, 9.055, 9.434, 9.663, 10.403, 11.662, 9.285,8.84)
> mean(x)
[1] 10.0696
```

We wish to estimate the standard error of the sample mean

$$S.E(\bar{x}) = \frac{\sigma_F}{\sqrt{n}}$$

Parametric and nonparametric bootstrap

nonparametric bootstrap

$$F \rightarrow (x_1, x_2, \ldots, x_n)$$

We resample from the empirical distribution

parametric bootstrap

We assume a parametric model for F

$$F(\theta)$$

We resample from

$$F(\hat{\theta})$$

Nonparametric bootstrap

The data (x_i)

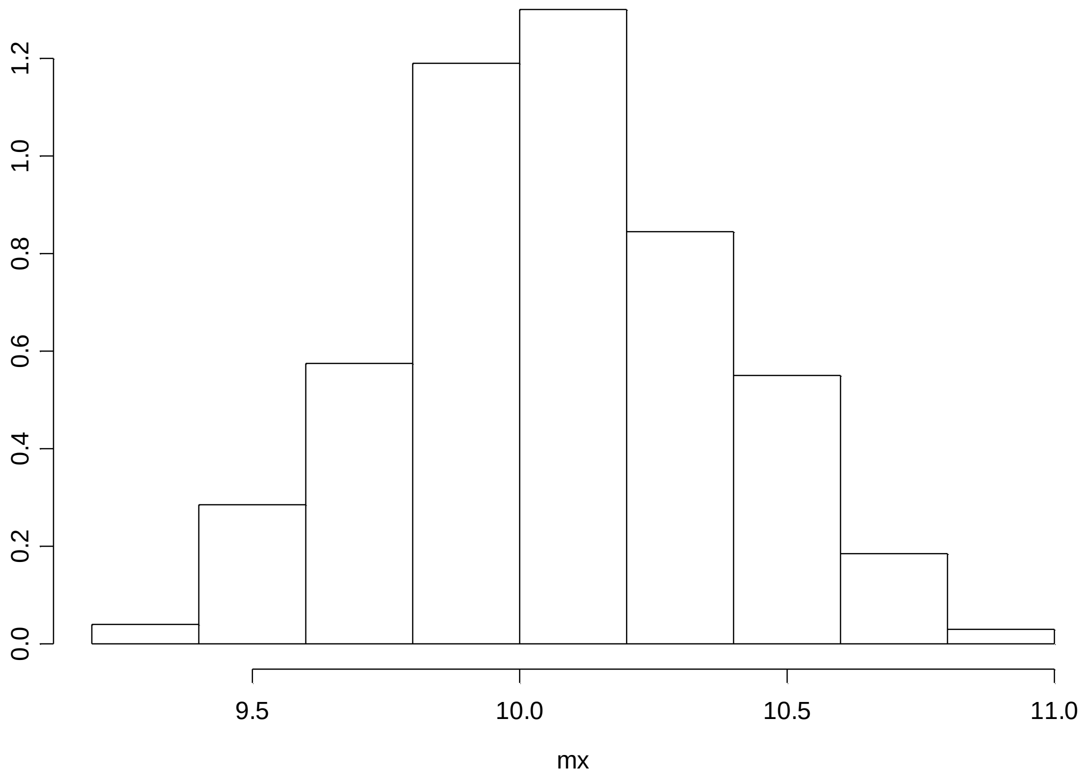
Bootstrap sample (x_i)*

Bootstrap sample (x_i)*

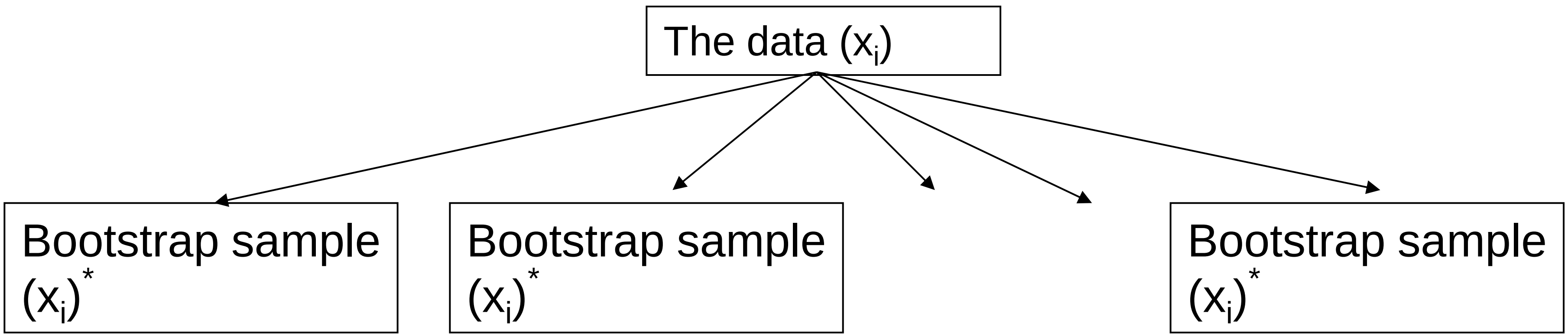
Bootstrap sample (x_i)*

$\hat{\mu}_1^*$ $\hat{\mu}_2^*$ $\hat{\mu}_B^*$

Nonparametric bootstrap



Nonparametric bootstrap



$\hat{\mu}_1^*$ $\hat{\mu}_2^*$ $\hat{\mu}_B^*$

$$S.E.(\hat{\mu}) = \left\{ \frac{1}{B-1} \sum_{b=1}^{B} \left(\hat{\mu}_b^* - \hat{\mu}^* \right)^2 \right\}^{0.5}$$

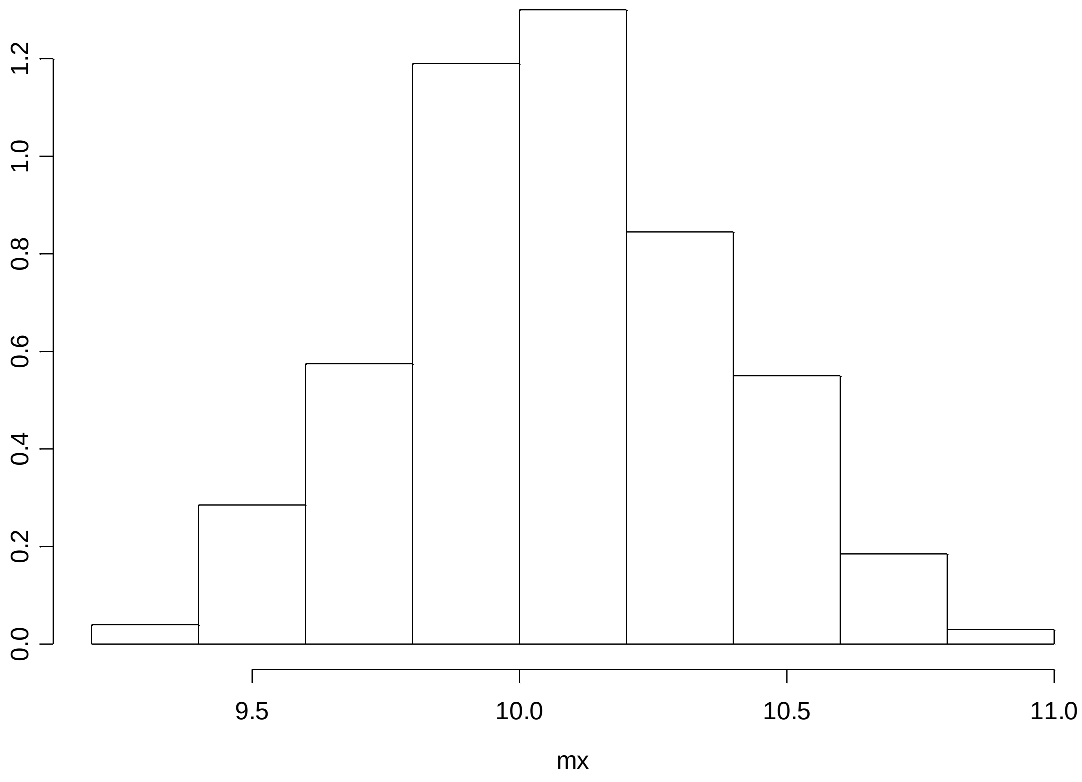
R code

```
> var(mx)
[1] 0.09357364
```

The estimated standard error 0.093

```
n<-length(x)
B<-1000
mx<-c(1:B)
for(i in 1:B){
cat(i)
boot.i<-sample(x,n,replace=T)
mx[i]<-mean(boot.i)
}
```

Nonparametric bootstrap



Parametric bootstrap

We assume a parametric model for F

$$F = N(\mu, \sigma^2)$$

We estimate F by

$$\hat{F} = N(\hat{\mu}, \hat{\sigma}^2)$$

We replace the unknown parameters in F with their plug-in estimates

Parametric bootstrap

The data (x_i)

Bootstrap sample (x_i)*

$N(\hat{\mu}, \hat{\sigma}^2)$

Bootstrap sample (x_i)*

$N(\hat{\mu}, \hat{\sigma}^2)$

Bootstrap sample (x_i)*

$N(\hat{\mu}, \hat{\sigma}^2)$

$\hat{\mu}_1^*$ $\hat{\mu}_2^*$ $\hat{\mu}_B^*$

$$S.E.(\hat{\mu}) = \left\{ \frac{1}{B+1} \sum_{b=1}^{B} \left(\hat{\mu}_b^* - \hat{\mu}^* \right)^2 \right\}^{0.5}$$

R code

```
> var(mx)
[1] 0.1007613
```

Bootstrap estimate for the standard error for the mean

```
B<-1000
MLx<-mean(x)
Varx<-var(x)
mx<-c(1:B)
for(i in 1:B){
cat(i)
boot.i<-rnorm(n,MLx,sqrt(Varx))
mx[i]<-mean(boot.i)
}
```

Parametric bootstrap