

Introduction to R

February 7, 2018

R and RStudio

The image shows the RStudio interface with four key components highlighted by numbered callouts:

- 1) Script:** The editor window on the left containing R code. The code includes comments and functions to load data, create a scatterplot, and add a linear regression line. The console output shows the execution of these commands.
- 2) Console:** The bottom-left pane showing the output of the R commands. It displays the results of the `read.csv` function, the `plot` function, and the `lines` function.
- 3) Workspace:** The top-right pane showing the objects in the R environment. It lists the `google` data frame (51 observations, 9 variables) and the `reg` object (a linear model).
- 4) Results/Plots:** The bottom-right pane displaying a scatterplot titled "Interest in Data Visualization Searches by Percent of Population with College Degrees". The plot shows a positive correlation between the percentage of the population with college degrees and the number of searches for data visualization. A red line represents the linear regression fit.

What R can do

What R can do

Everything.^{1,2}

1 Except think about your science

2 Occasionally in a non efficient way

What R can do

Everything.^{1,2}

1 Except think about your science

2 Occasionally in a non efficient way

What about RStudio?

Make your life easier

Many handy tricks.

- 1 The mean
- 2 Working with 2D objects
- 3 T-test
- 4 Open problem

Calculating a mean: Arithmetic and assignment

```
(2 + 3 + 5 + 1) / 4
```

```
[1] 2.75
```

Calculating a mean: Arithmetic and assignment

```
(2 + 3 + 5 + 1) / 4
```

```
[1] 2.75
```

```
a <- 2
```

```
b <- 3
```

```
c <- 5
```

```
d <- 1
```

```
(a + b + c + d) / 4
```

```
[1] 2.75
```


Calculating a mean: Arithmetic and assignment

```
(2 + 3 + 5 + 1) / 4
```

```
[1] 2.75
```

```
a <- 2
```

```
b <- 3
```

```
c <- 5
```

```
d <- 1
```

```
(a + b + c + d) / 4
```

```
[1] 2.75
```

```
a <- 45
```

```
(a + b + c + d) / 4
```

```
[1] 13.5
```

Calculating a mean: using vectors

```
c(2,3,5,1) # c is for concatenate
```

```
[1] 2 3 5 1
```

Calculating a mean: using vectors

```
c(2,3,5,1) # c is for concatenate
```

```
[1] 2 3 5 1
```

```
mydata <- c(2,3,5,1) # save the vector
```

Calculating a mean: using vectors

```
c(2,3,5,1) # c is for concatenate
```

```
[1] 2 3 5 1
```

```
mydata <- c(2,3,5,1) # save the vector
```

```
mydata <- (2,3,5,1) # c is missing => error!
```

```
Error: <text>:1:14: unexpected ','  
1: mydata <- (2,  
               ^
```

Calculating a mean: using vectors

```
c(2,3,5,1) # c is for concatenate
```

```
[1] 2 3 5 1
```

```
mydata <- c(2,3,5,1) # save the vector
```

```
mydata <- (2,3,5,1) # c is missing => error!
```

```
Error: <text>:1:14: unexpected ',',  
1: mydata <- (2,  
  ^
```

Why bother with vectors?

```
mydata[2] <- 4  
mydata
```

```
[1] 2 4 5 1
```

Calculating a mean: using functions

How to use a function?

```
?mean
```

Calculating a mean: using functions

How to use a function?

```
?mean
```

```
mean(c(2,4,5,1))
```

```
[1] 3
```

```
mean(mydata)
```

```
[1] 3
```

```
mean(x = mydata)
```

```
[1] 3
```

- 1 The mean
- 2 Working with 2D objects
- 3 T-test
- 4 Open problem

Loading data

```
data("trees")
```

Loading data

```
data("trees")
```

```
str(trees)
```

```
'data.frame': 31 obs. of 3 variables:
```

```
$ Girth : num 8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
```

```
$ Height: num 70 65 63 72 81 83 66 75 80 75 ...
```

```
$ Volume: num 10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 ...
```

Try also `summary`, `class`, `head`, `tail`

- 1 The mean
- 2 Working with 2D objects
- 3 T-test**
- 4 Open problem

Student's T.test introduction

```
?t.test
```

Student's T.test introduction

```
?t.test
```

```
t.test(1:10, y = c(7:20))
```

Welch Two Sample t-test

data: 1:10 and c(7:20)

t = -5.4349, df = 21.982, p-value = 1.855e-05

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-11.052802 -4.947198

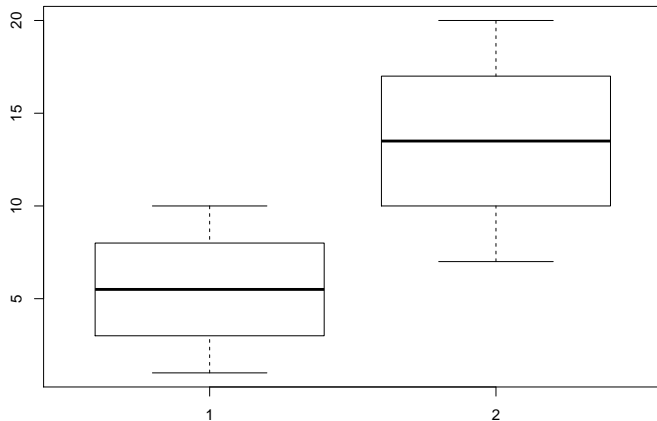
sample estimates:

mean of x mean of y

5.5 13.5

T.test introduction

```
boxplot(c(1:10, 7:20) ~ c(rep(1,10), rep(2, 14)))
```



- 1 The mean
- 2 Working with 2D objects
- 3 T-test
- 4 Open problem

The t.test problem

```
t.test(1:10, y = c(2:20,-9), var.equal = FALSE)
```

Welch Two Sample t-test

data: 1:10 and c(2:20, -9)

t = -2.4345, df = 27.642, p-value = 0.02163

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-8.2885317 -0.7114683

sample estimates:

mean of x mean of y

5.5 10.0

The t.test problem

```
t.test(1:10, y = c(2:20,-9), var.equal = TRUE)
```

Two Sample t-test

data: 1:10 and c(2:20, -9)

t = -1.9134, df = 28, p-value = 0.06597

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-9.3175688 0.3175688

sample estimates:

mean of x mean of y

5.5 10.0

The t.test problem

Random sample from a Gaussian distribution with variance 1

```
set.seed(seed = 179)
x1 <- rnorm(n = 20, mean = 0, sd = 1)
x2 <- rnorm(n = 20, mean = 0, sd = 1)
```

```
var(x1)
```

```
[1] 0.7040416
```

```
var(x2)
```

```
[1] 1.810404
```

The t.test problem

```
var.test(x = x1, y = x2)
```

F test to compare two variances

data: x1 and x2

F = 0.38889, num df = 19, denom df = 19, p-value = 0.04593

alternative hypothesis: true ratio of variances is not equal to 1

95 percent confidence interval:

0.1539260 0.9825028

sample estimates:

ratio of variances

0.3888866

Should we use var.equal = TRUE or FALSE ?

When var.test significant/not?

Bonus open problems if you get bored