Introduction to R for Biologists

Day 1 – Intro to R and the Tidyverse Ecosystem

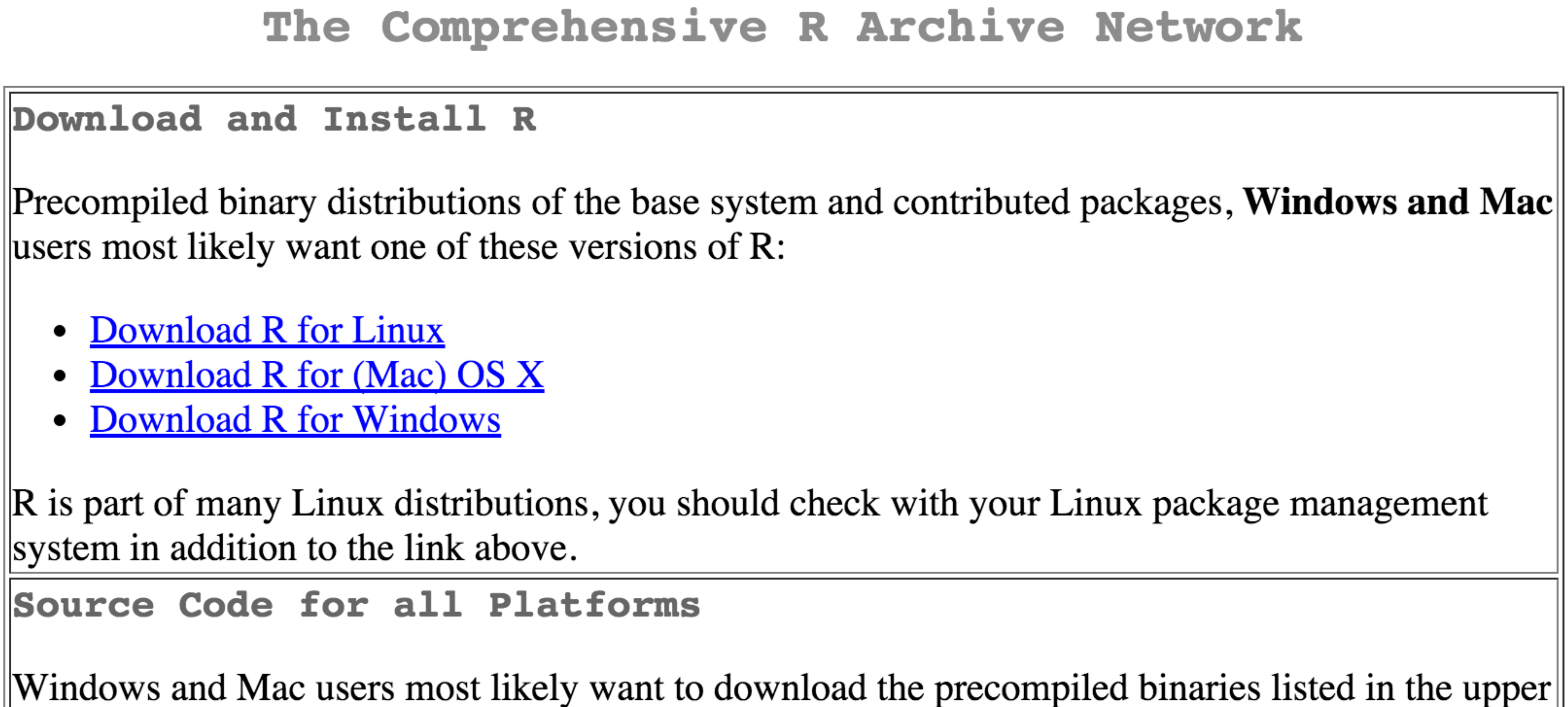
*Several slides have been contributed by or modified from slides by Dr. Claus Wilke

Day 1 Outline

1. How to get set up using R
2. How and why to use RStudio & R Markdown (.Rmd)
3. Basics of programming
   - Data types
   - Functions
   - Troubleshooting
4. Intro to the Tidyverse
   - Tidy vs untidy data
   - Tidyverse-specific functions

R: The premier data analysis and visualization platform

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



The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

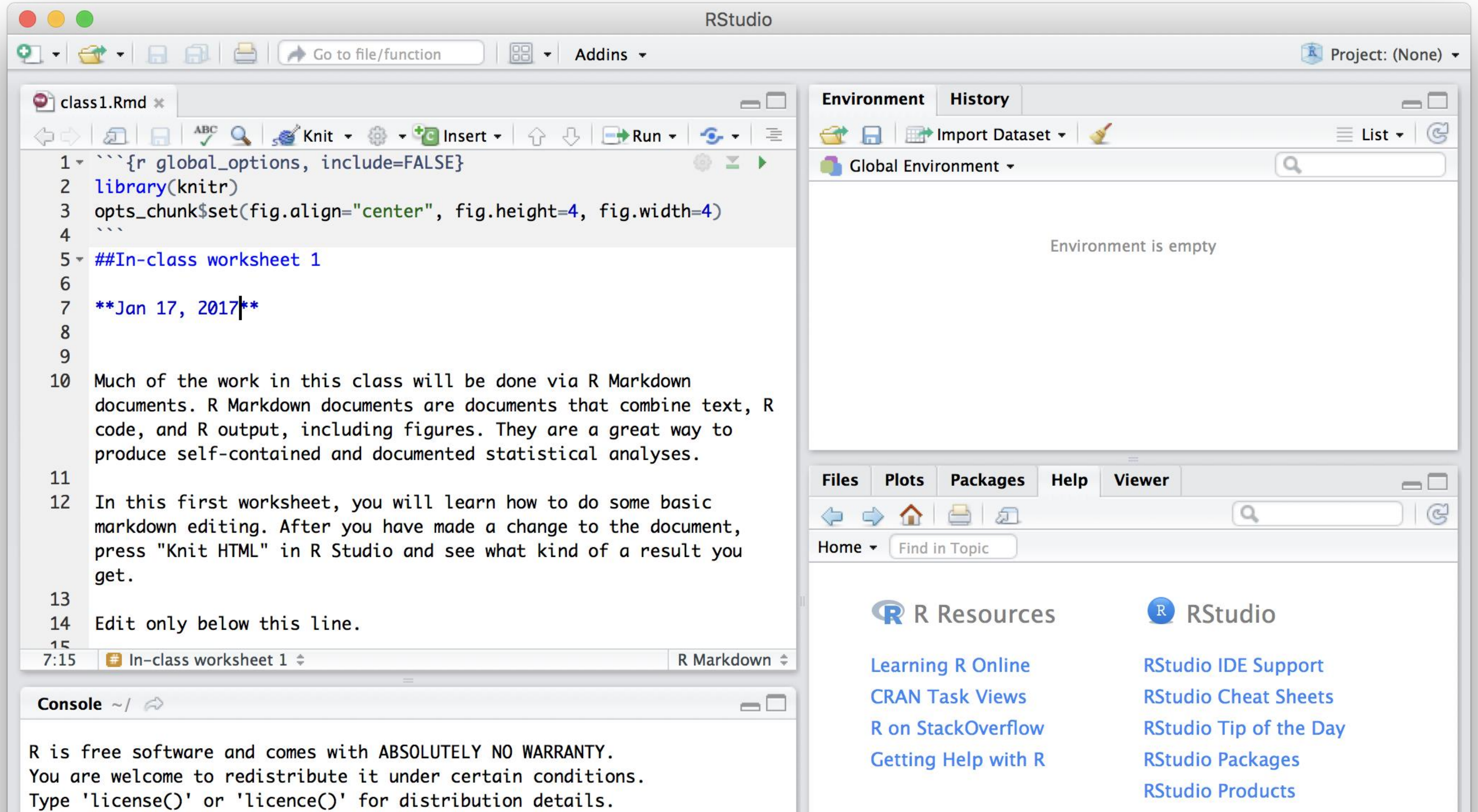
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper

R Studio:
A nice user interface for R

https://www.rstudio.com/products/rstudio/download/



~~~
```{r global_options, include=FALSE}
library(knitr)
opts_chunk$set(fig.align="center", fig.height=4, fig.width=4)
~~~
##In-class worksheet 1

Jan 17, 2017

Much of the work in this class will be done via R Markdown documents. R Markdown documents are documents that combine text, R code, and R output, including figures. They are a great way to produce self-contained and documented statistical analyses.

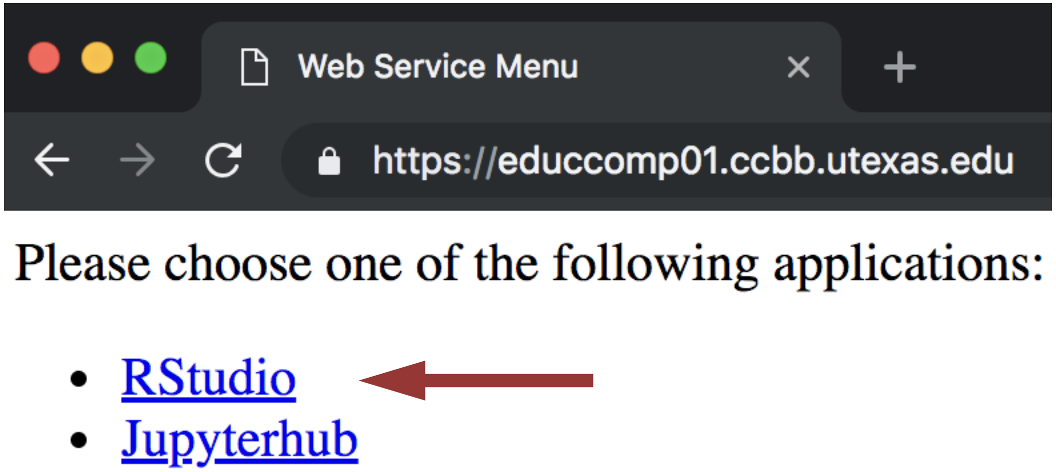
In this first worksheet, you will learn how to do some basic markdown editing. After you have made a change to the document, press "Knit HTML" in R Studio and see what kind of a result you get.

Edit only below this line.
```

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
```
```

# Access R Studio through your web browser

1. https://gsafcomp01.ccbb.utexas.edu/
2. https://gsafcomp02.ccbb.utexas.edu/

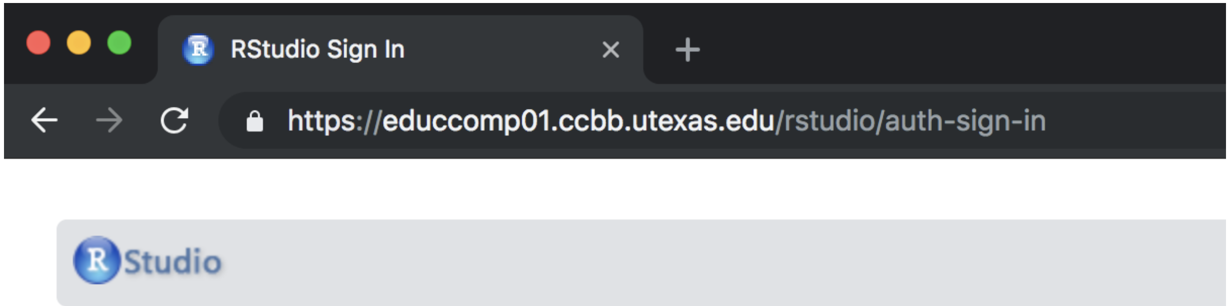
# Select RStudio



Please choose one of the following applications:

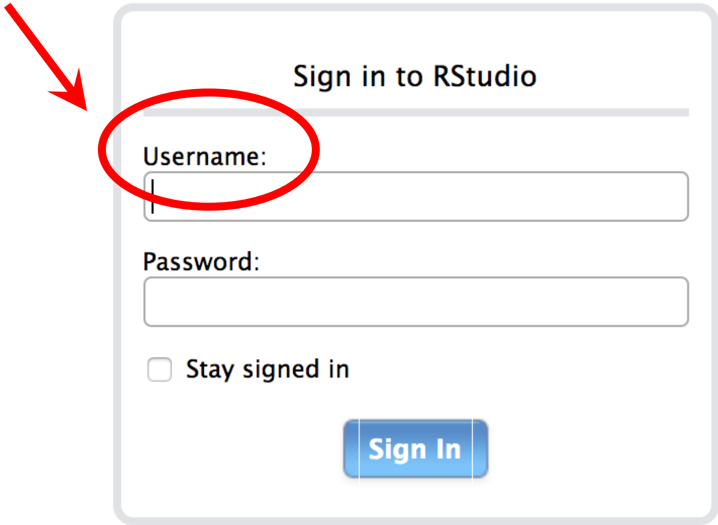
- RStudio
- Jupyterhub

# Sign in with your student# and password



https://educcomp01.ccbb.utexas.edu/rstudio/auth-sign-in

Refer to class email for your individual username


Sign in to RStudio

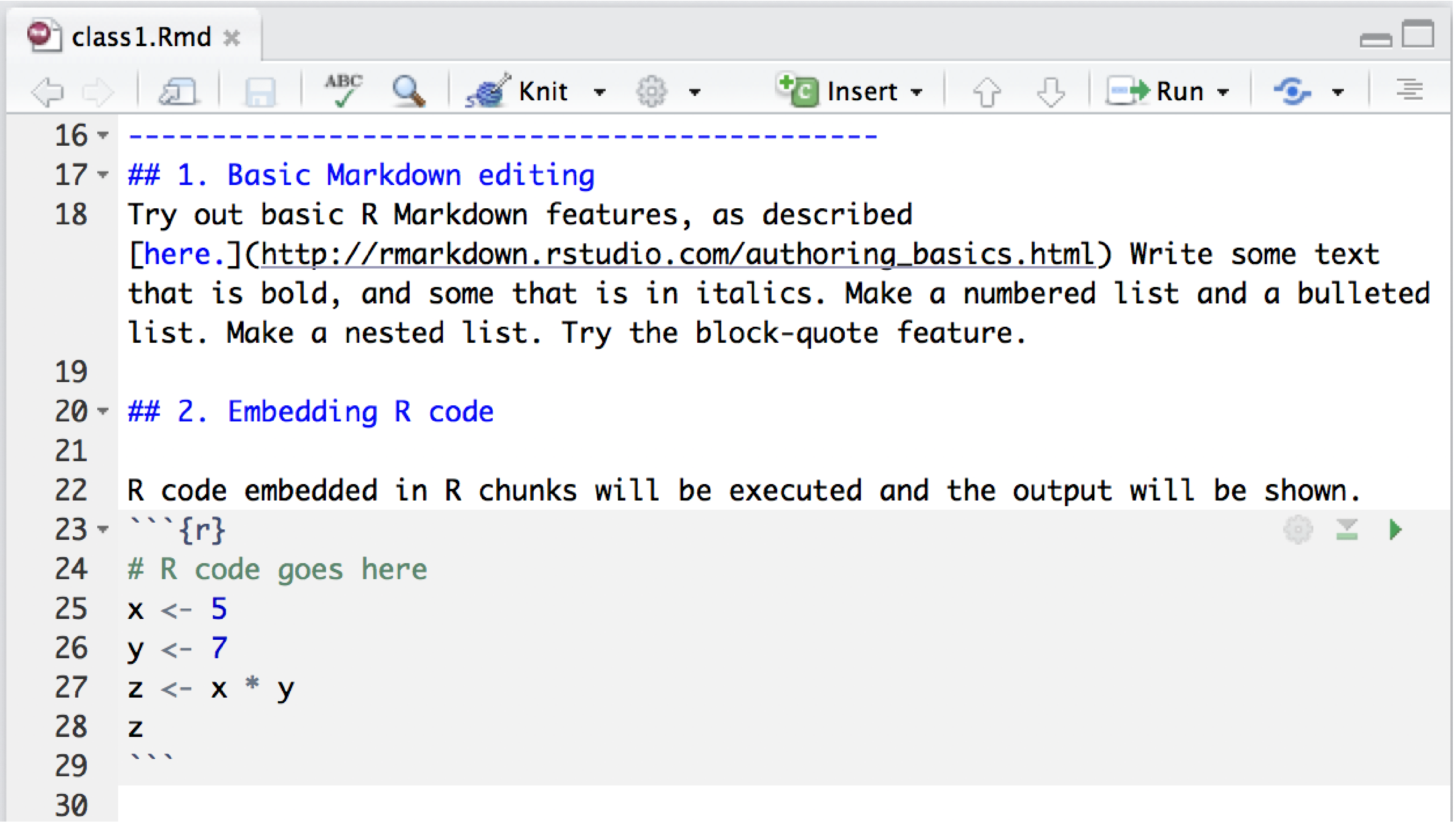
Username:

Password:

☐ Stay signed in

# R Markdown

# R Markdown:
# Writing documents with embedded R code



~~~
---------------------------------------------
## 1. Basic Markdown editing
Try out basic R Markdown features, as described [here.](http://rmarkdown.rstudio.com/authoring_basics.html) Write some text that is bold, and some that is in italics. Make a numbered list and a bulleted list. Make a nested list. Try the block-quote feature.

## 2. Embedding R code

R code embedded in R chunks will be executed and the output will be shown.
```{r}
# R code goes here
x <- 5
y <- 7
z <- x * y
z
~~~

```
```

R Markdown:
Writing documents with embedded R code

1. Basic Markdown editing

Try out basic R Markdown features, as described here. Write some text that is bold, and some that is in italics. Make a numbered list and a bulleted list. Make a nested list. Try the block-quote feature.

2. Embedding R code

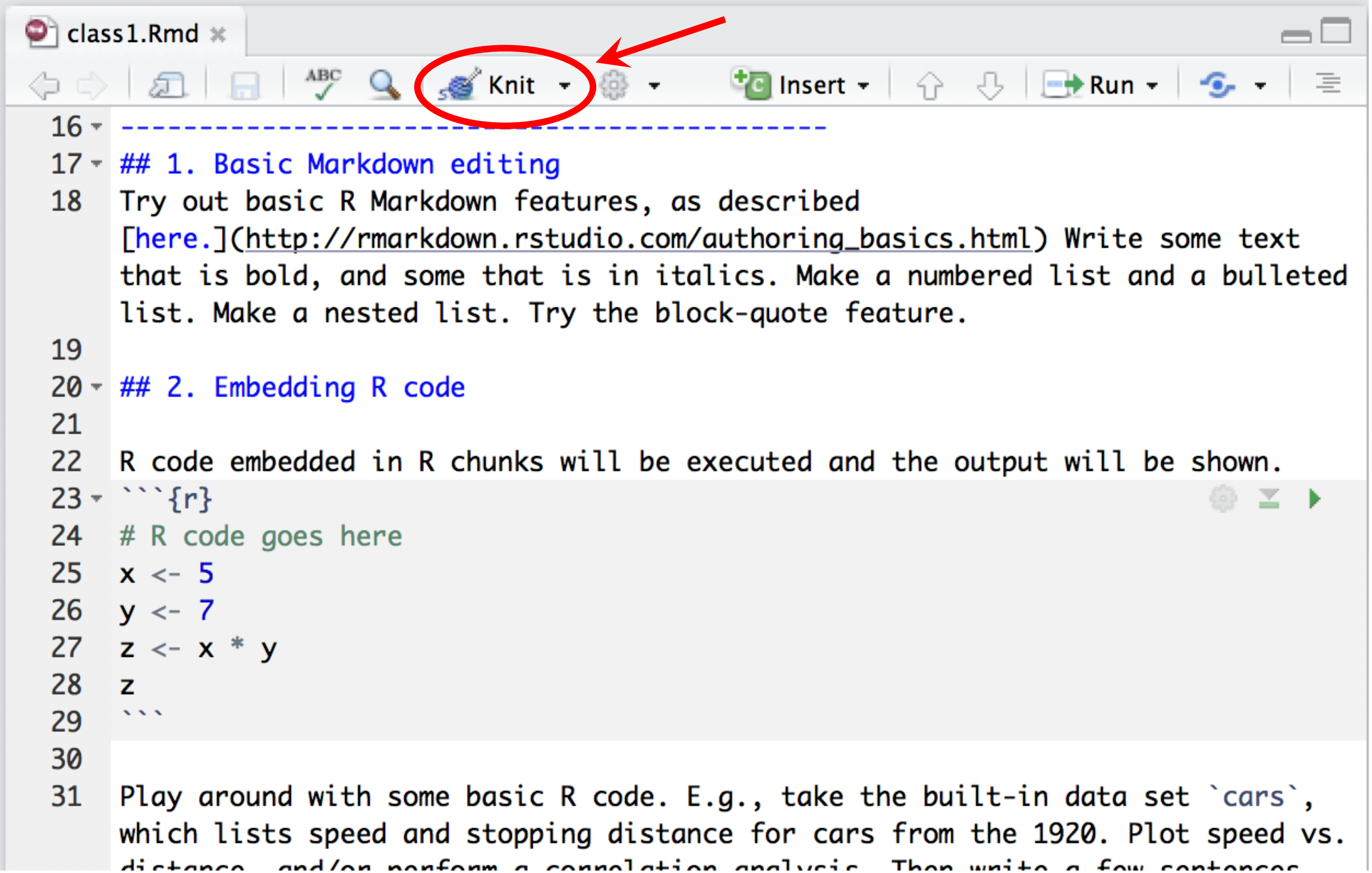
R code embedded in R chunks will be executed and the output will be shown.

```r
# R code goes here
x <- 5
y <- 7
z <- x * y
z
```

```
## [1] 35
```

Play around with some basic R code. E.g., take the built-in data set `cars`, which lists speed and stopping distance for cars from the 1920. Plot speed vs. distance, and/or perform a correlation analysis. Then write a few sentences describing what you see.

We convert R Markdown to HTML by "knitting" the Markdown file



~~~
----------------------------------------------
## 1. Basic Markdown editing
Try out basic R Markdown features, as described
[here.](http://rmarkdown.rstudio.com/authoring_basics.html) Write some text
that is bold, and some that is in italics. Make a numbered list and a bulleted
list. Make a nested list. Try the block-quote feature.

## 2. Embedding R code

R code embedded in R chunks will be executed and the output will be shown.
```{r}
# R code goes here
x <- 5
y <- 7
z <- x * y
z
~~~

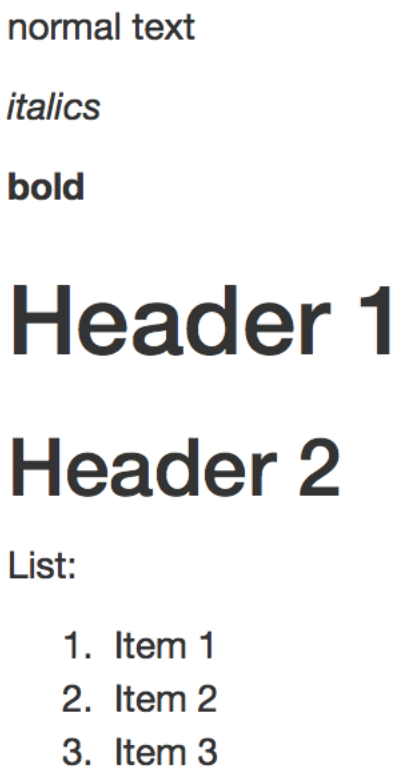
Play around with some basic R code. E.g., take the built-in data set `cars`,
which lists speed and stopping distance for cars from the 1920. Plot speed vs.
```
```

Markdown basics

http://rmarkdown.rstudio.com/authoring_basics.html

```
normal text

*italics*

**bold**

# Header 1
## Header 2

List:

1. Item 1
2. Item 2
3. Item 3
```

→



normal text

italics

bold

Header 1

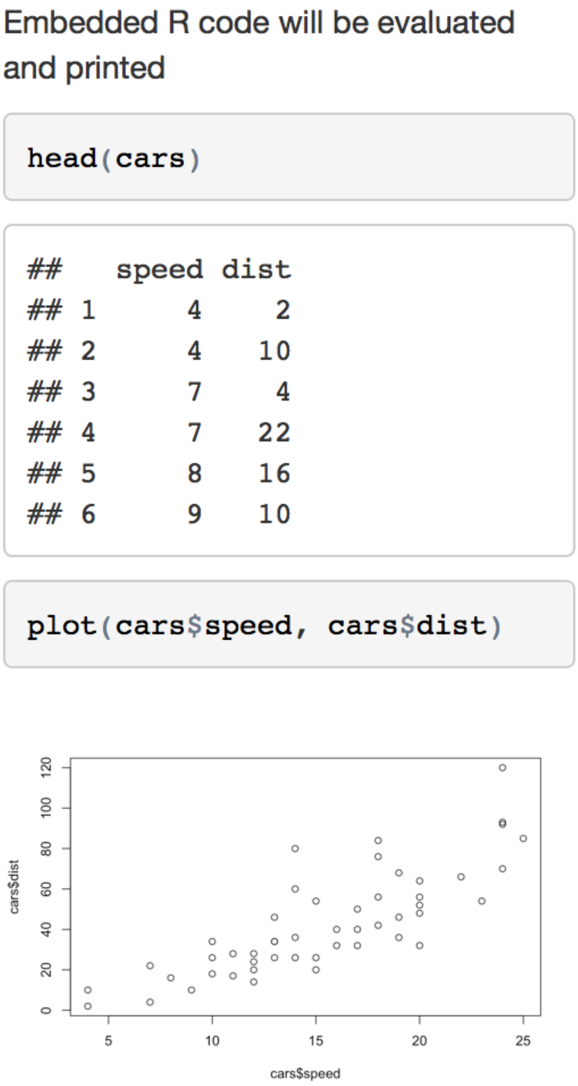
Header 2

List:

1. Item 1
2. Item 2
3. Item 3

Markdown basics

~~~
Embedded R code will be
evaluated and printed

```{r}
head(cars)
plot(cars$speed, cars$dist)
```
~~~



Embedded R code will be evaluated and printed

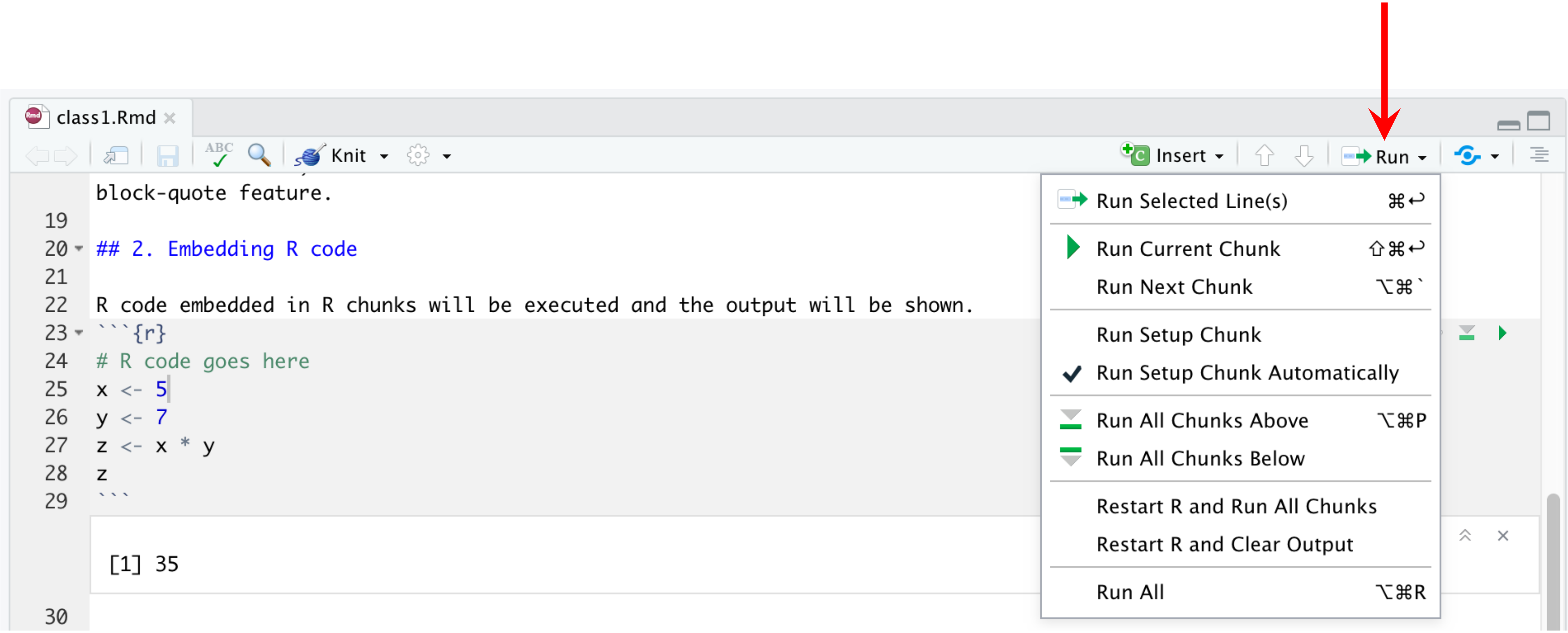
```
head(cars)
```

```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

```
plot(cars$speed, cars$dist)
```

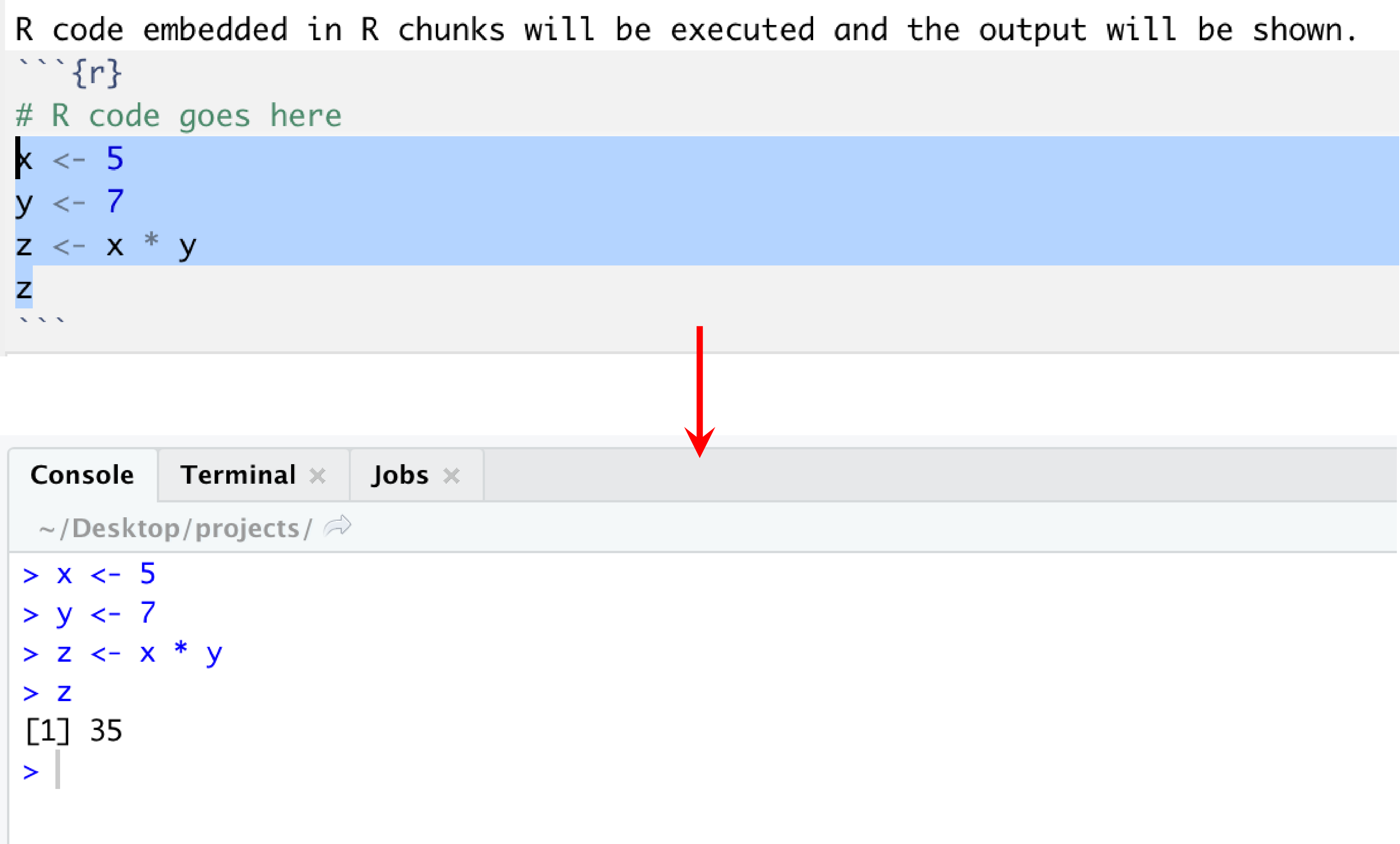
Different ways to execute code in RStudio

Press the “Run” button



~~~
block-quote feature.

## 2. Embedding R code

R code embedded in R chunks will be executed and the output will be shown.
```{r}
# R code goes here
x <- 5
y <- 7
z <- x * y
z
~~~
```

```
[1] 35
```
```

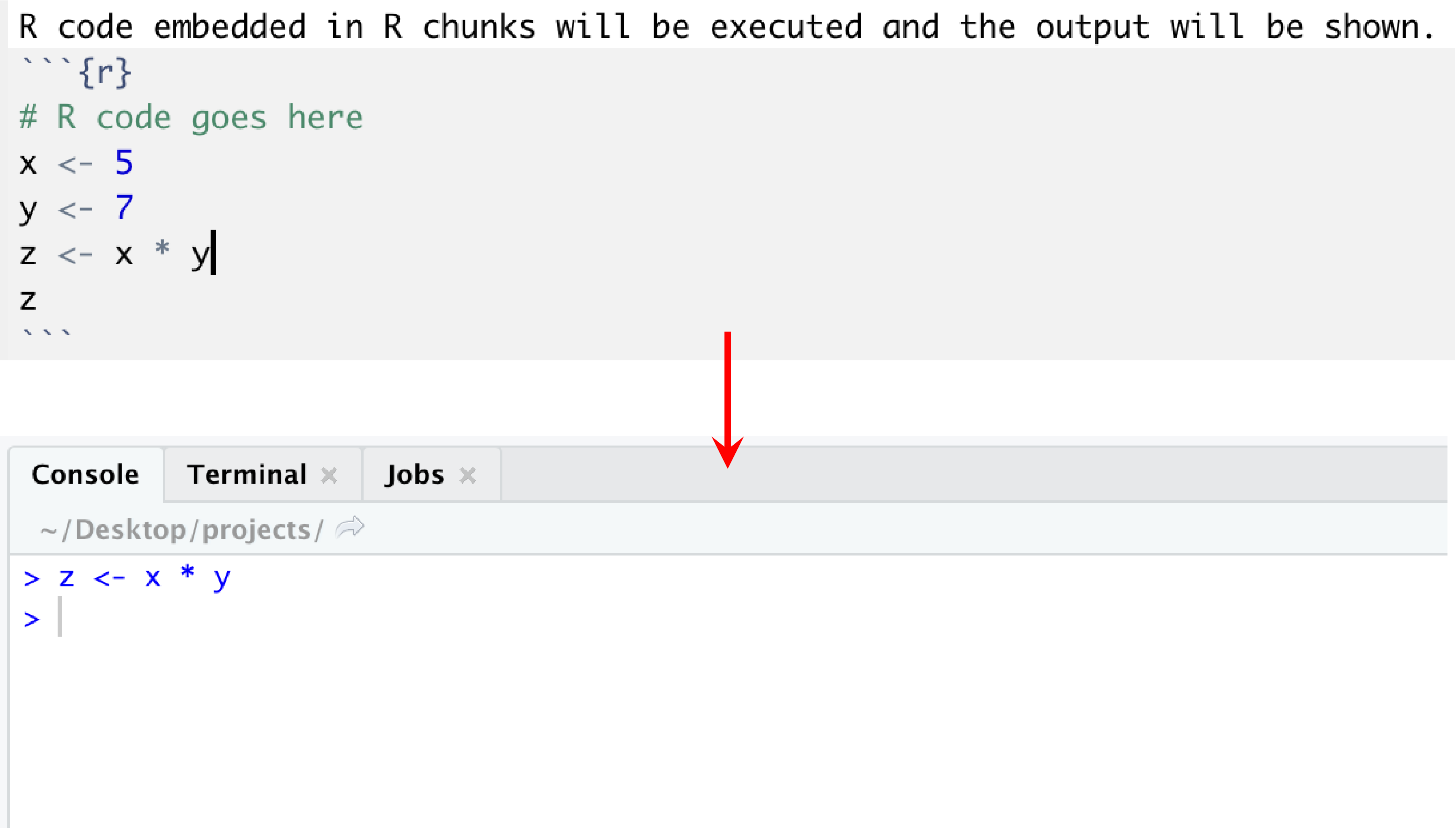
Highlight code you want to execute and press ctrl+Enter (cmd+Enter on Macs)



R code embedded in R chunks will be executed and the output will be shown.

~~~
```{r}
# R code goes here
x <- 5
y <- 7
z <- x * y
z
~~~
```

Console
Terminal
Jobs
~/Desktop/projects/

```
> x <- 5
> y <- 7
> z <- x * y
> z
[1] 35
>
```
```

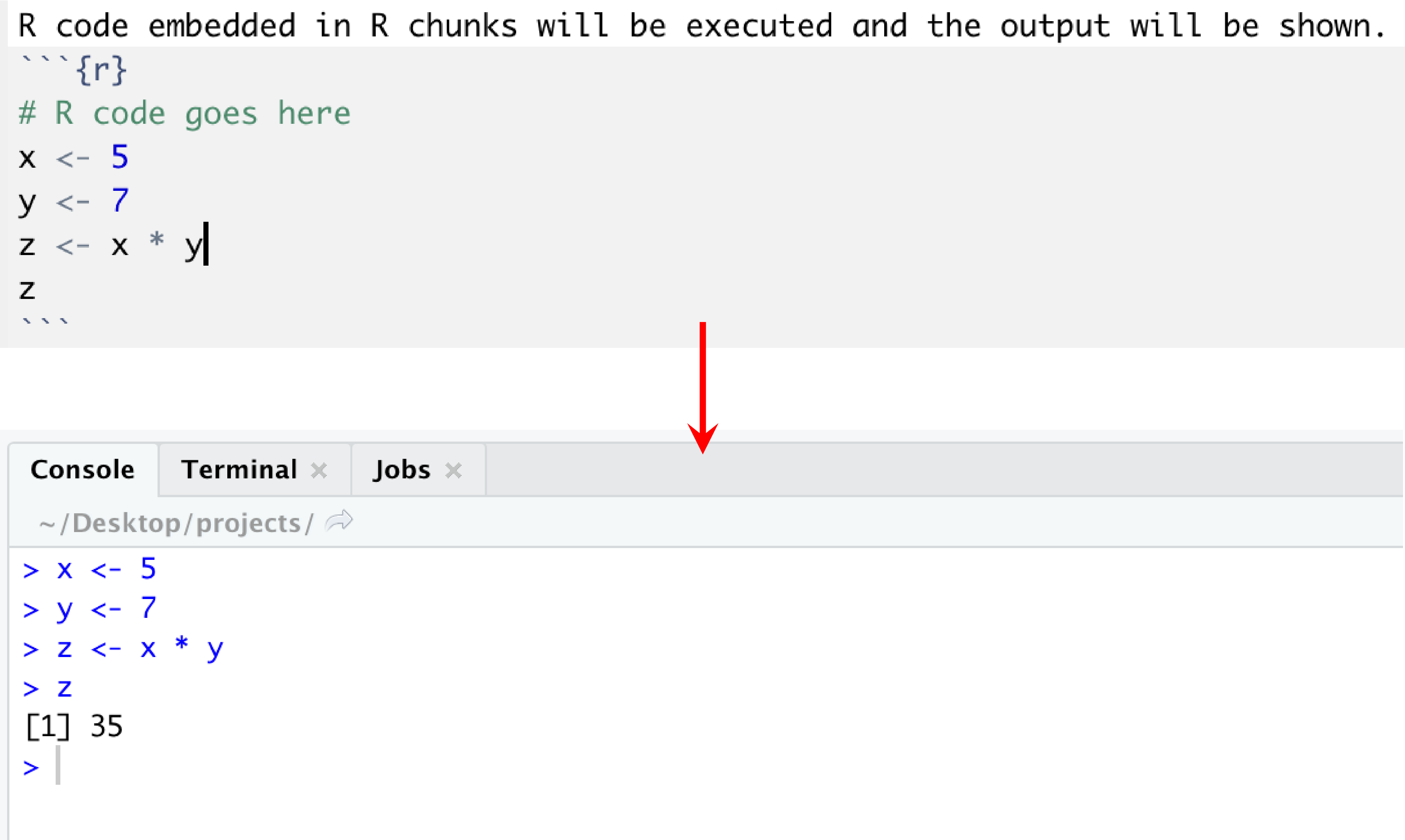
# Place pointer on line of code you want to execute, press ctrl+Enter (cmd+Enter on Macs)



R code embedded in R chunks will be executed and the output will be shown.

~~~
```{r}
# R code goes here
x <- 5
y <- 7
z <- x * y
z
~~~
```

Console
Terminal
Jobs
~/Desktop/projects/

```
> z <- x * y
> 
```
```

Use ctrl+Shift+Enter (cmd+Shift+Enter on Macs) to execute an entire code chunk



R code embedded in R chunks will be executed and the output will be shown.

~~~
```{r}
# R code goes here
x <- 5
y <- 7
z <- x * y
z
~~~
```

Console
Terminal
Jobs
~/Desktop/projects/

```
> x <- 5
> y <- 7
> z <- x * y
> z
[1] 35
> 
```
```

# Shortcuts for coding

- **Ctrl+Shift+M** (Cmd+Shift+M on Macs) produces a pipe operator `%>%` (will be used with the tidyverse)
- **Ctrl+Shift+C** (Cmd+Shift+C on Macs) will comment/uncomment a line or multiple lines
- **Tab** and **Shift+Tab** will indent and un-indent lines, respectively

# R Programming Basics

# Assignments, numbers, vectors

```
> x <- 5
> x
[1] 5
```

Assign number 5 to variable x

```
> 5*x^2+7
[1] 132
```

Calculate $5*x^2+7$

```
> y <- c(1, 2, 3, 4, 5)
> y
[1] 1 2 3 4 5
```

Create vector, assign to variable y

```
> x*y
[1]  5 10 15 20 25
```

Multiply each element in y with the number in x

Strings

A string contains text:

```
> name <- “Rachael Cox"
> name
[1] “Rachael Cox"
```

A vector of strings:

```
> animals <- c("cat", "mouse", "mouse",
"cat", "rabbit")
> animals
[1] "cat"     "mouse"   "mouse"   "cat"
"rabbit"
```

Factors

Factors keep track of distinct categories (levels) in a vector:

```
> animals
[1] "cat"     "mouse"  "mouse"  "cat"
"rabbit”

> factor(animals)
[1] cat    mouse  mouse  cat    rabbit
Levels: cat mouse rabbit
```

Data frames

We use data frames to store data sets with multiple variables:

```
> pets <- data.frame(
    family = c(1, 2, 3, 4, 5),
   pet = animals
 )

> pets
  family   pet
1      1   cat
2      2 mouse
3      3 mouse
4      4   cat
```

Data frames

We access individual columns in a data frame with $ + the column name:

```
> pets$family
[1] 1 2 3 4 5


> pets$pet
[1] cat    mouse  mouse  cat    rabbit
Levels: cat mouse rabbit
```

Data frames

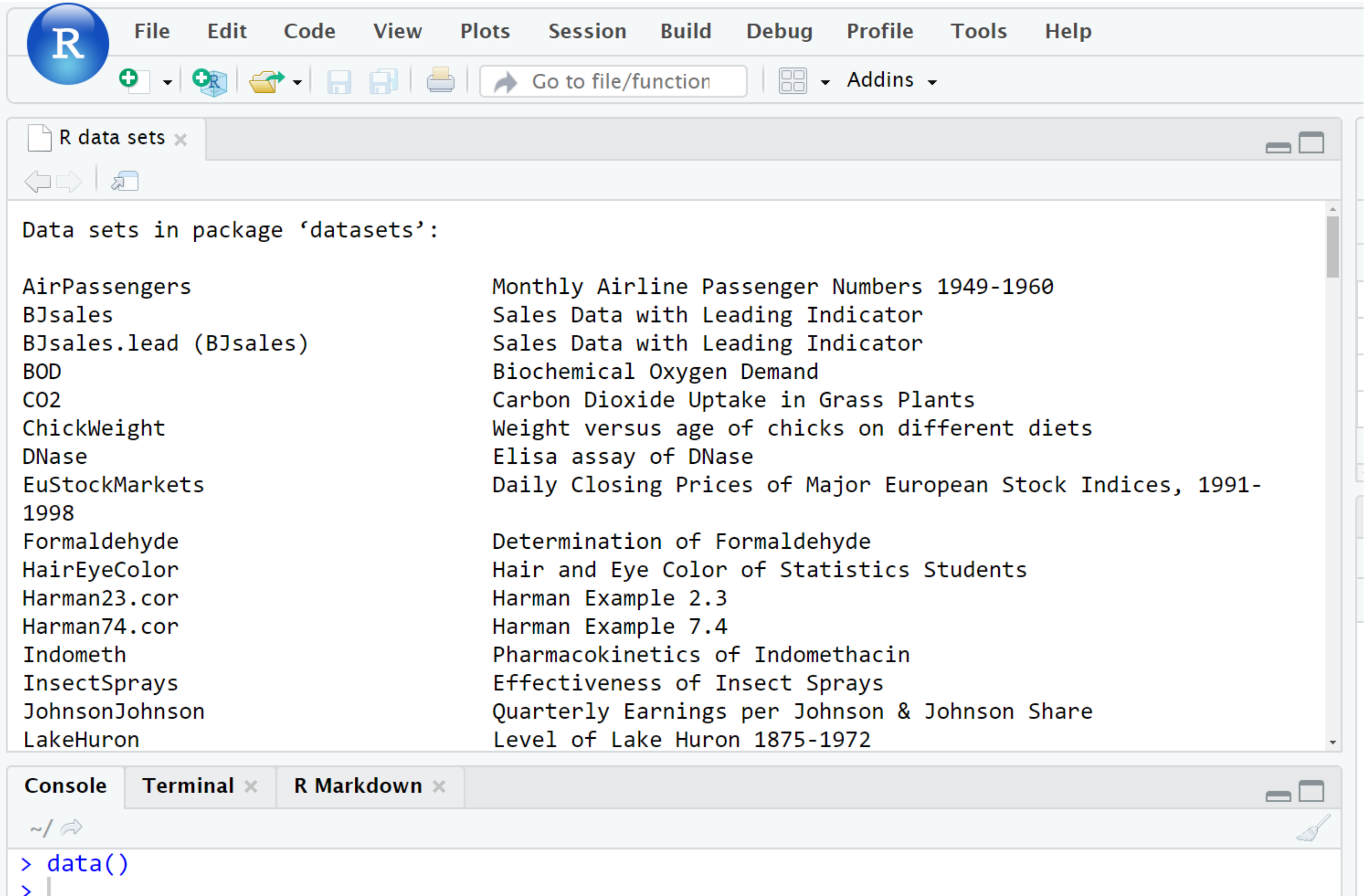
R has many built-in data frames:

```
> cars
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
7    10   18
8    10   26
9    10   34
```

```
> chickwts
   weight    feed
1     179 horsebean
2     160 horsebean
3     136 horsebean
4     227 horsebean
...   ...    ...
11    309 linseed
12    229 linseed
13    181 linseed
14    141 linseed
```

Data frames

Available built-in datasets can be accessed with `data()`

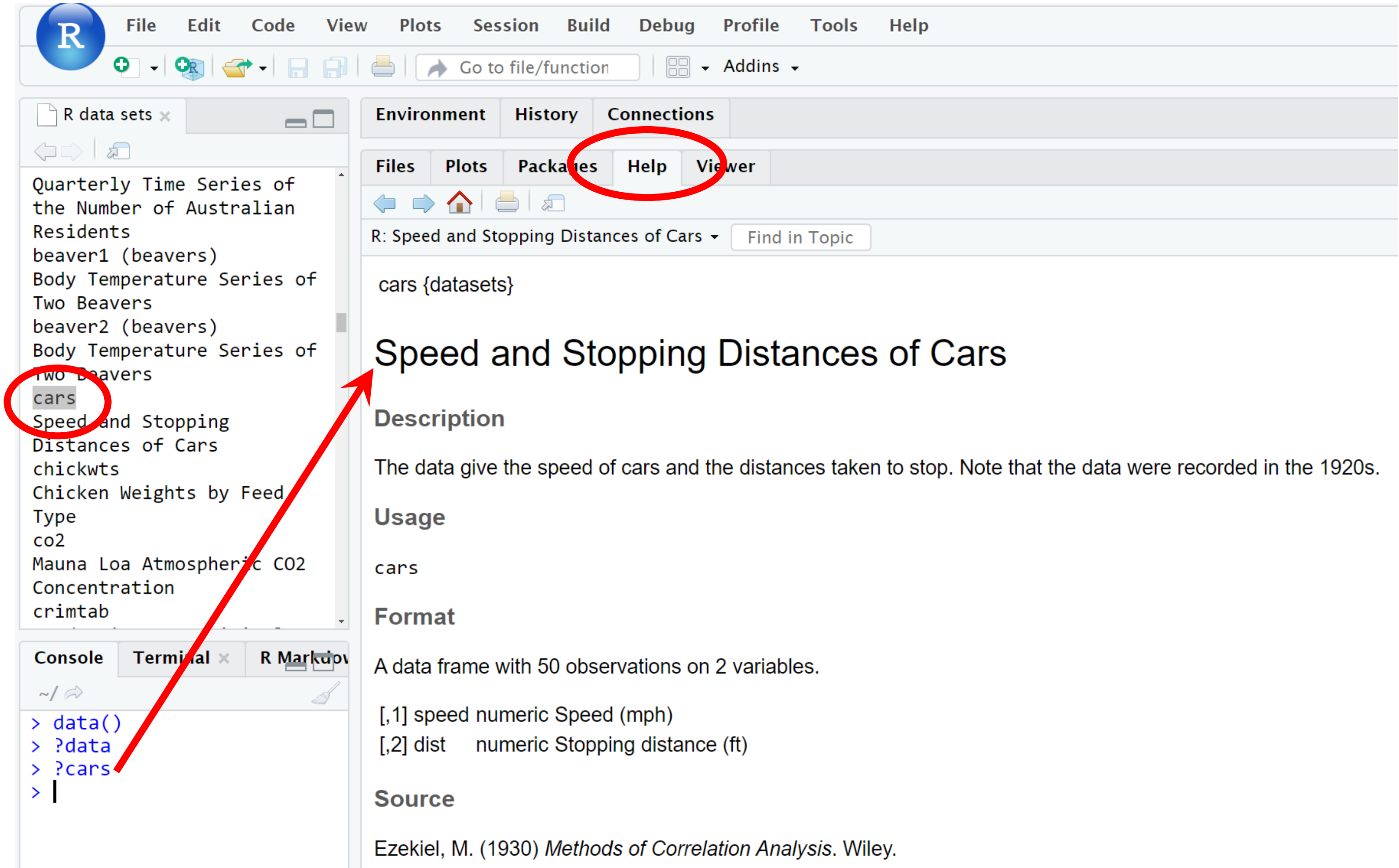


```
Data sets in package ‘datasets’:

AirPassengers                   Monthly Airline Passenger Numbers 1949-1960
BJsales                         Sales Data with Leading Indicator
BJsales.lead (BJsales)          Sales Data with Leading Indicator
BOD                             Biochemical Oxygen Demand
CO2                             Carbon Dioxide Uptake in Grass Plants
ChickWeight                     Weight versus age of chicks on different diets
DNase                           Elisa assay of DNase
EuStockMarkets                  Daily Closing Prices of Major European Stock Indices, 1991-
1998
Formaldehyde                    Determination of Formaldehyde
HairEyeColor                    Hair and Eye Color of Statistics Students
Harman23.cor                    Harman Example 2.3
Harman74.cor                    Harman Example 7.4
Indometh                        Pharmacokinetics of Indomethacin
InsectSprays                    Effectiveness of Insect Sprays
JohnsonJohnson                  Quarterly Earnings per Johnson & Johnson Share
LakeHuron                       Level of Lake Huron 1875-1972
```

```
> data()
>
```

Data frames

Data set information can be accessed with `?dataset`



```
Quarterly Time Series of
the Number of Australian
Residents
beaver1 (beavers)
Body Temperature Series of
Two Beavers
beaver2 (beavers)
Body Temperature Series of
Two Beavers
cars
Speed and Stopping
Distances of Cars
chickwts
Chicken Weights by Feed
Type
co2
Mauna Loa Atmospheric CO2
Concentration
crimtab
```

```
> data()
> ?data
> ?cars
>
```

cars {datasets}

Speed and Stopping Distances of Cars

Description

The data give the speed of cars and the distances taken to stop. Note that the data were recorded in the 1920s.

Usage

`cars`

Format

A data frame with 50 observations on 2 variables.

[,1] speed numeric Speed (mph)
[,2] dist numeric Stopping distance (ft)

Source

Ezekiel, M. (1930) *Methods of Correlation Analysis*. Wiley.

Data frames

The `head()` function shows the first few lines of a data frame:

```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
>
```

Functions

Functions are called in the format `function(argument)`

```
> head(cars)
```

Function name

First argument

Functions

Functions can have any number of required arguments or optional arguments

```
> head(cars, 8)
```

Second argument
(optional; default = 6)

Function name

First argument
(required)

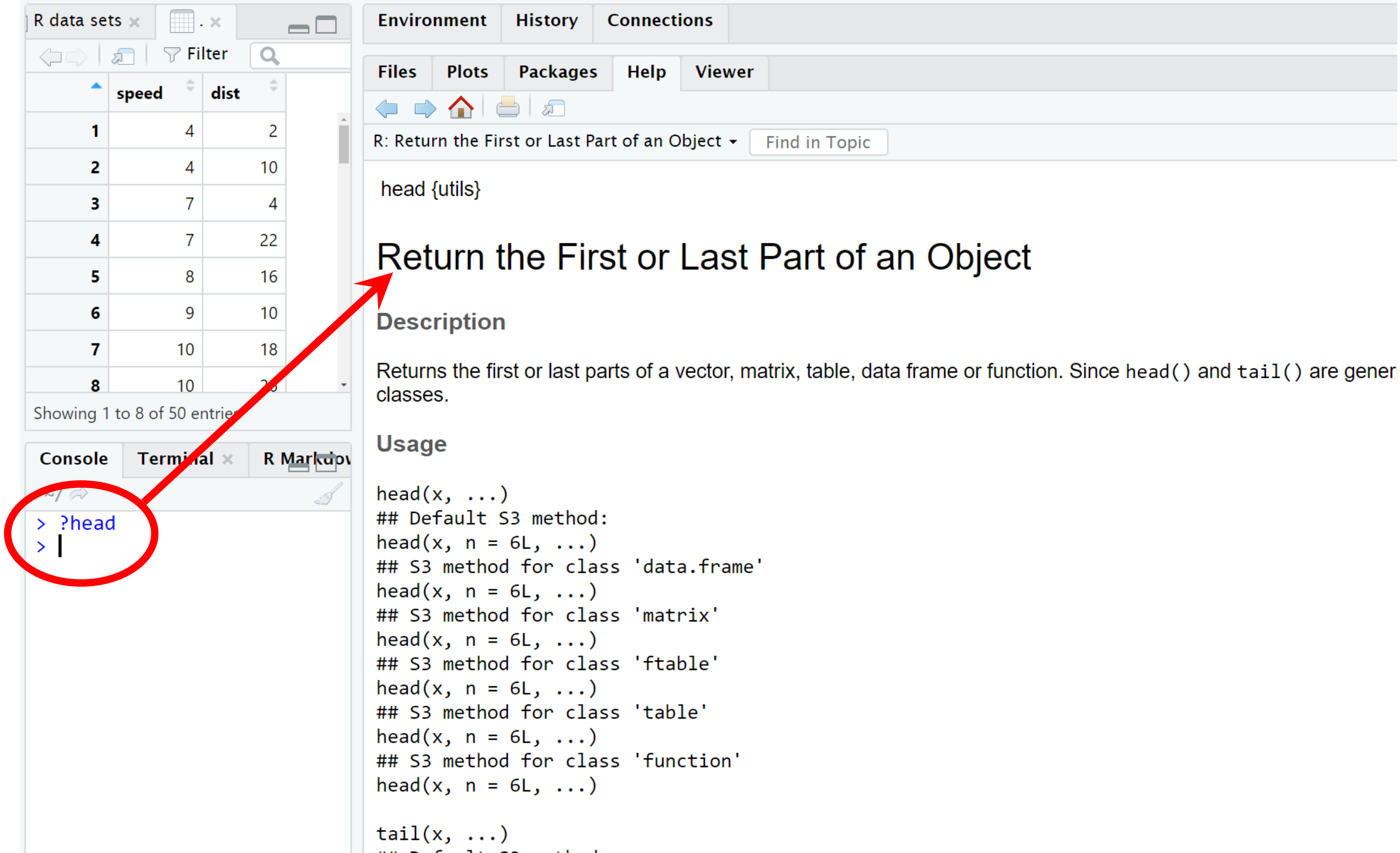
Functions

`head(cars, 8)` will show the first 8 lines of the data frame instead of the default 6:

```
> head(cars, 8)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
7    10   18
8    10   26
```

Functions

More information about what a function does and/or requires can be accessed with `?function`



	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10
7	10	18
8	10	26

Showing 1 to 8 of 50 entries

```
> ?head
> 
```

head {utils}

Return the First or Last Part of an Object

Description

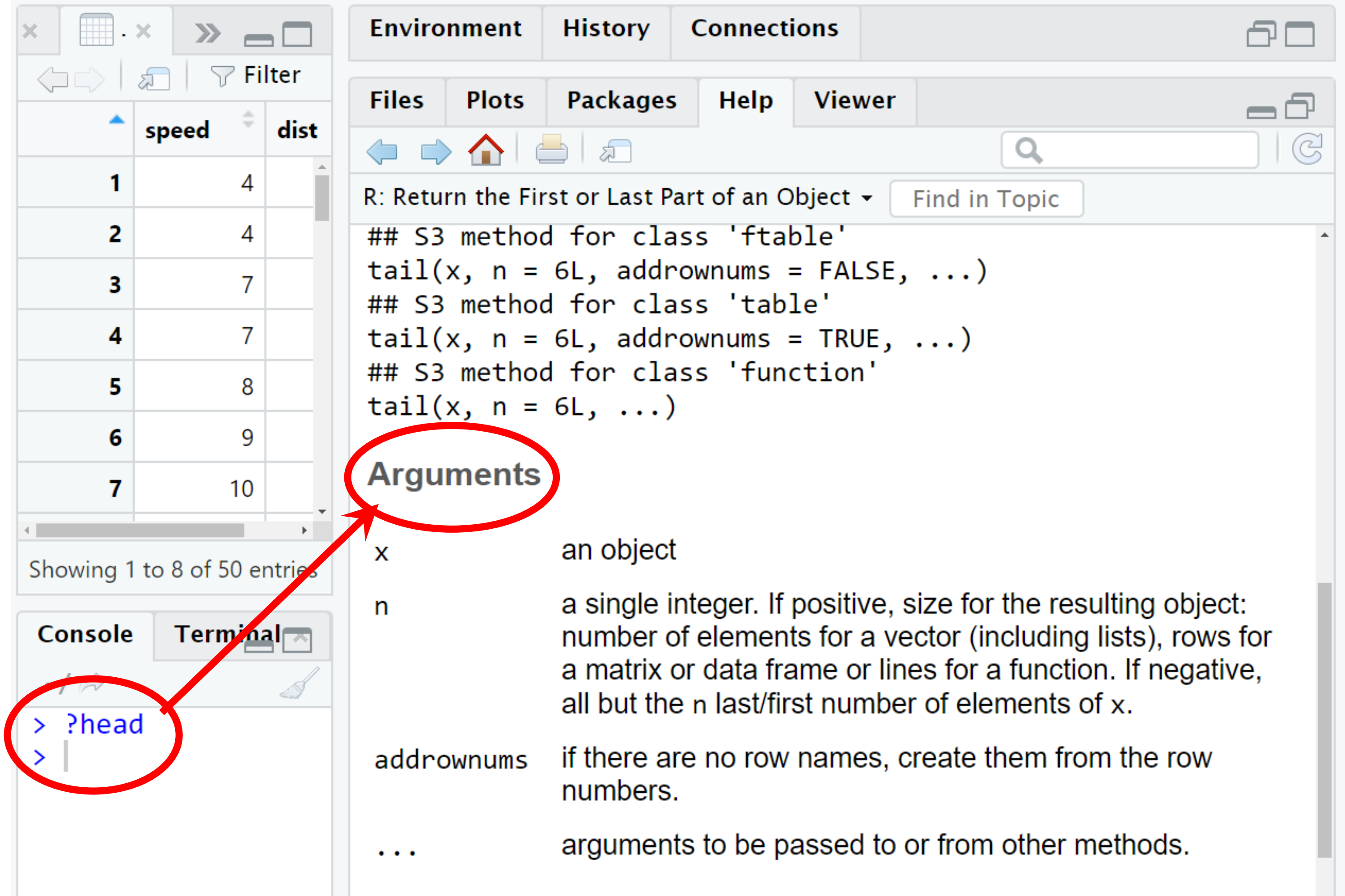
Returns the first or last parts of a vector, matrix, table, data frame or function. Since `head()` and `tail()` are gener classes.

Usage

```
head(x, ...)
## Default S3 method:
head(x, n = 6L, ...)
## S3 method for class 'data.frame'
head(x, n = 6L, ...)
## S3 method for class 'matrix'
head(x, n = 6L, ...)
## S3 method for class 'ftable'
head(x, n = 6L, ...)
## S3 method for class 'table'
head(x, n = 6L, ...)
## S3 method for class 'function'
head(x, n = 6L, ...)

tail(x, ...)
```

Functions

`?function` has argument information



	speed	dist
1	4	
2	4	
3	7	
4	7	
5	8	
6	9	
7	10	

Showing 1 to 8 of 50 entries

```
> ?head
>
```

R: Return the First or Last Part of an Object

```
## S3 method for class 'ftable'
tail(x, n = 6L, addrownums = FALSE, ...)
## S3 method for class 'table'
tail(x, n = 6L, addrownums = TRUE, ...)
## S3 method for class 'function'
tail(x, n = 6L, ...)
```

Arguments

`x`	an object
`n`	a single integer. If positive, size for the resulting object: number of elements for a vector (including lists), rows for a matrix or data frame or lines for a function. If negative, all but the `n` last/first number of elements of `x`.
`addrownums`	if there are no row names, create them from the row numbers.
`...`	arguments to be passed to or from other methods.

Functions

We can implicitly or explicitly pass arguments

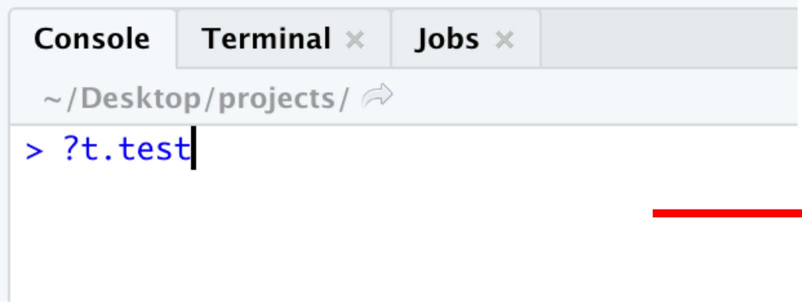
```
> head(cars, 8)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
7    10   18
8    10   26
>
```

```
> head(x=cars, n=8)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
7    10   18
8    10   26
>
```

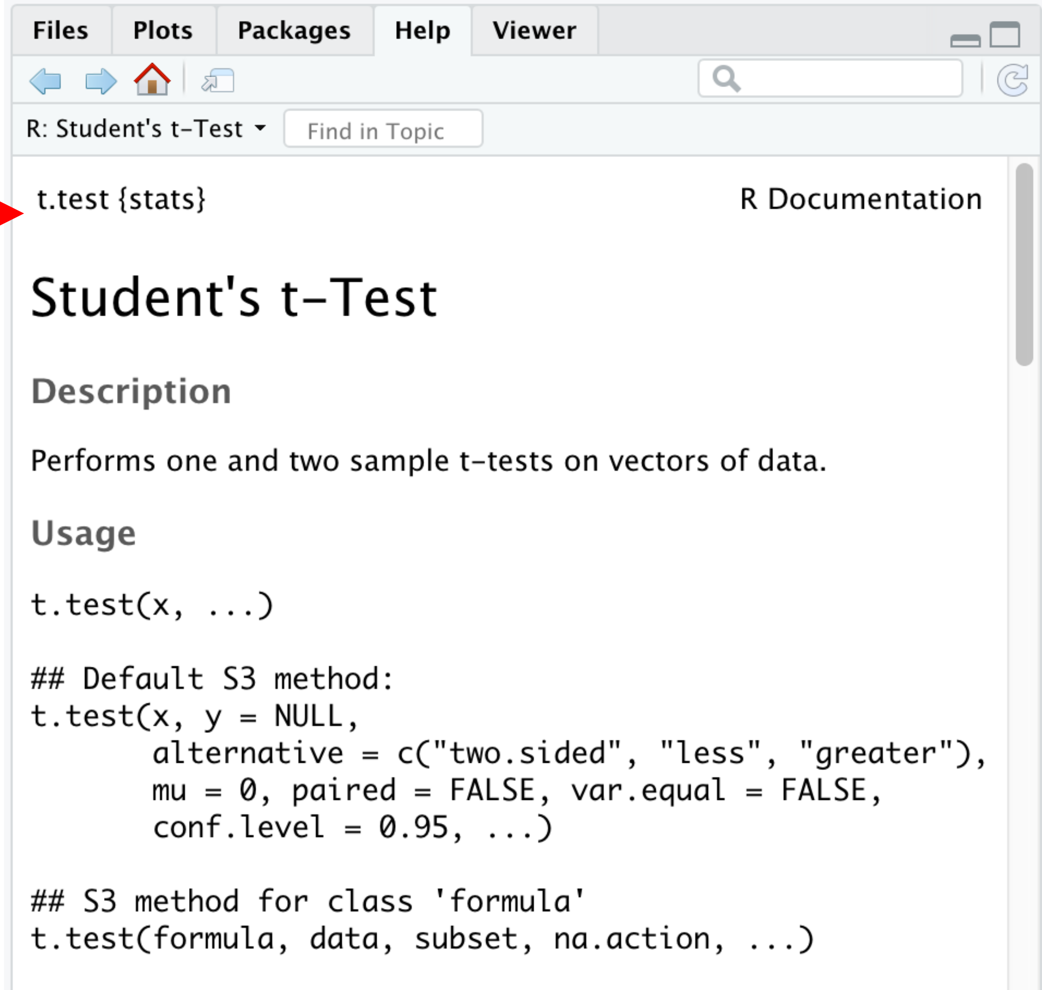
Troubleshooting

Ask RStudio for help

Type `?function` into console



```
> ?t.test
```



t.test {stats} R Documentation

Student's t-Test

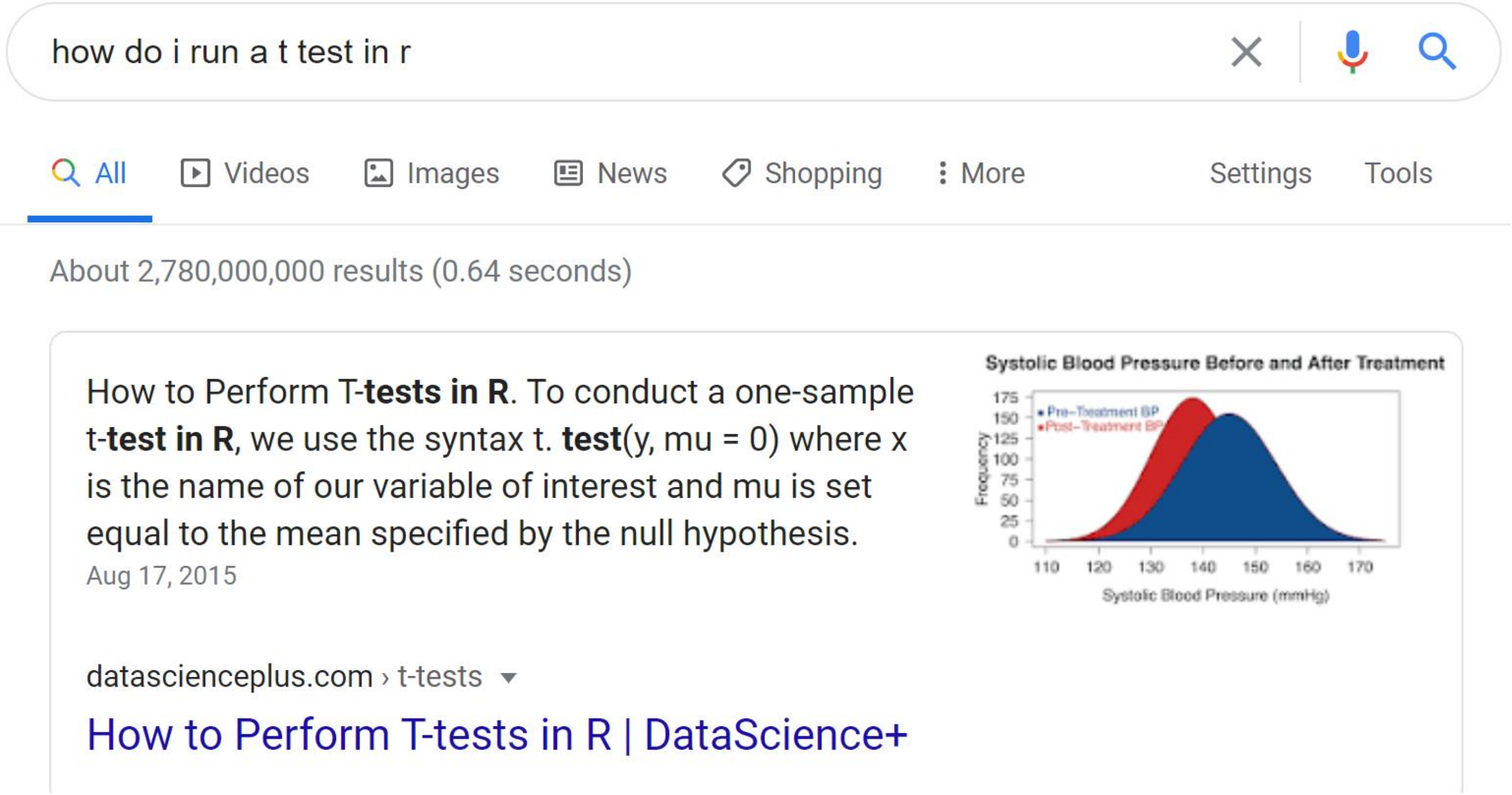
Description

Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

Ask Google for help



how do i run a t test in r

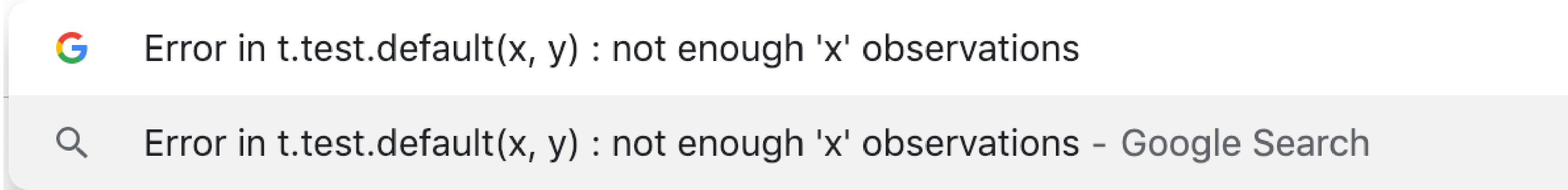
About 2,780,000,000 results (0.64 seconds)

How to Perform T-**tests in R**. To conduct a one-sample t-**test in R**, we use the syntax t. **test**(y, mu = 0) where x is the name of our variable of interest and mu is set equal to the mean specified by the null hypothesis.

Aug 17, 2015

datascienceplus.com › t-tests

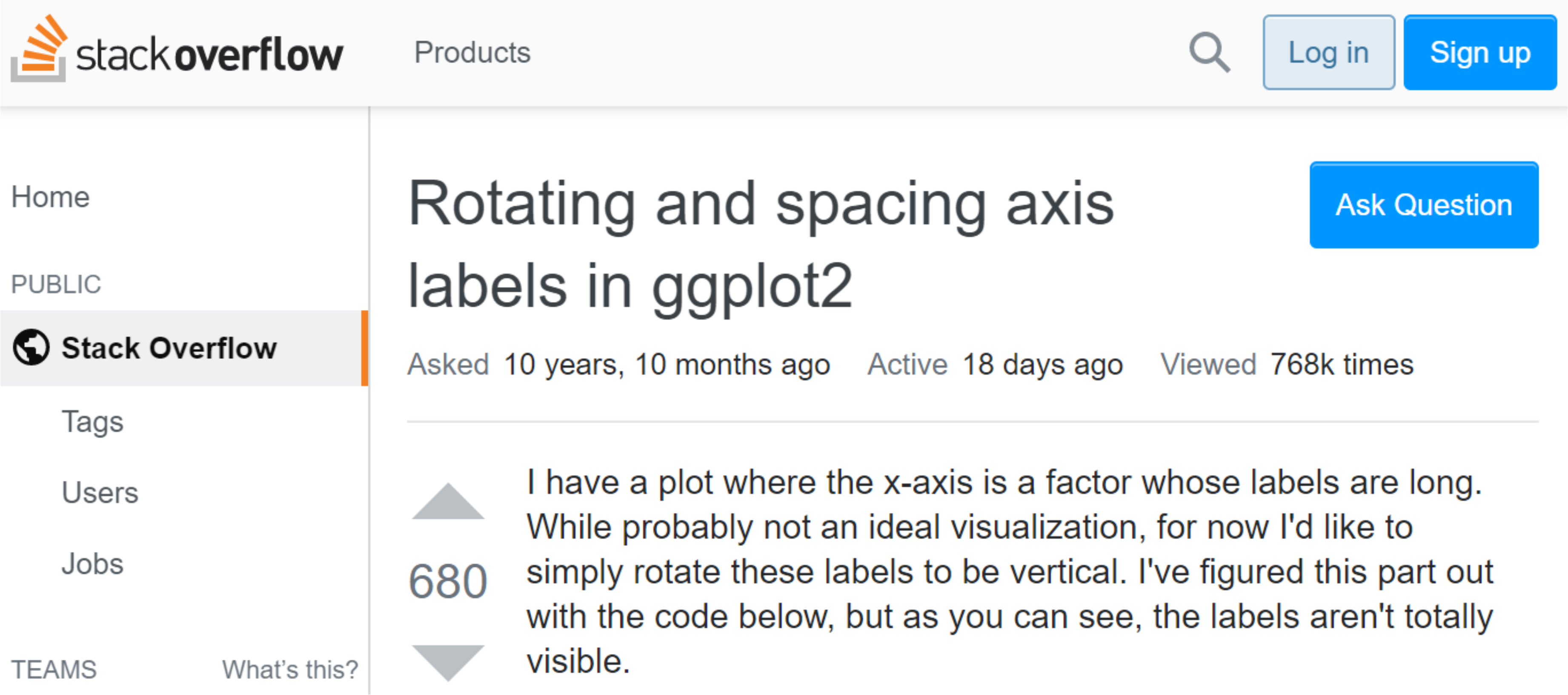
How to Perform T-tests in R | DataScience+



Error in t.test.default(x, y) : not enough 'x' observations

Error in t.test.default(x, y) : not enough 'x' observations - Google Search

Ask StackOverflow for help



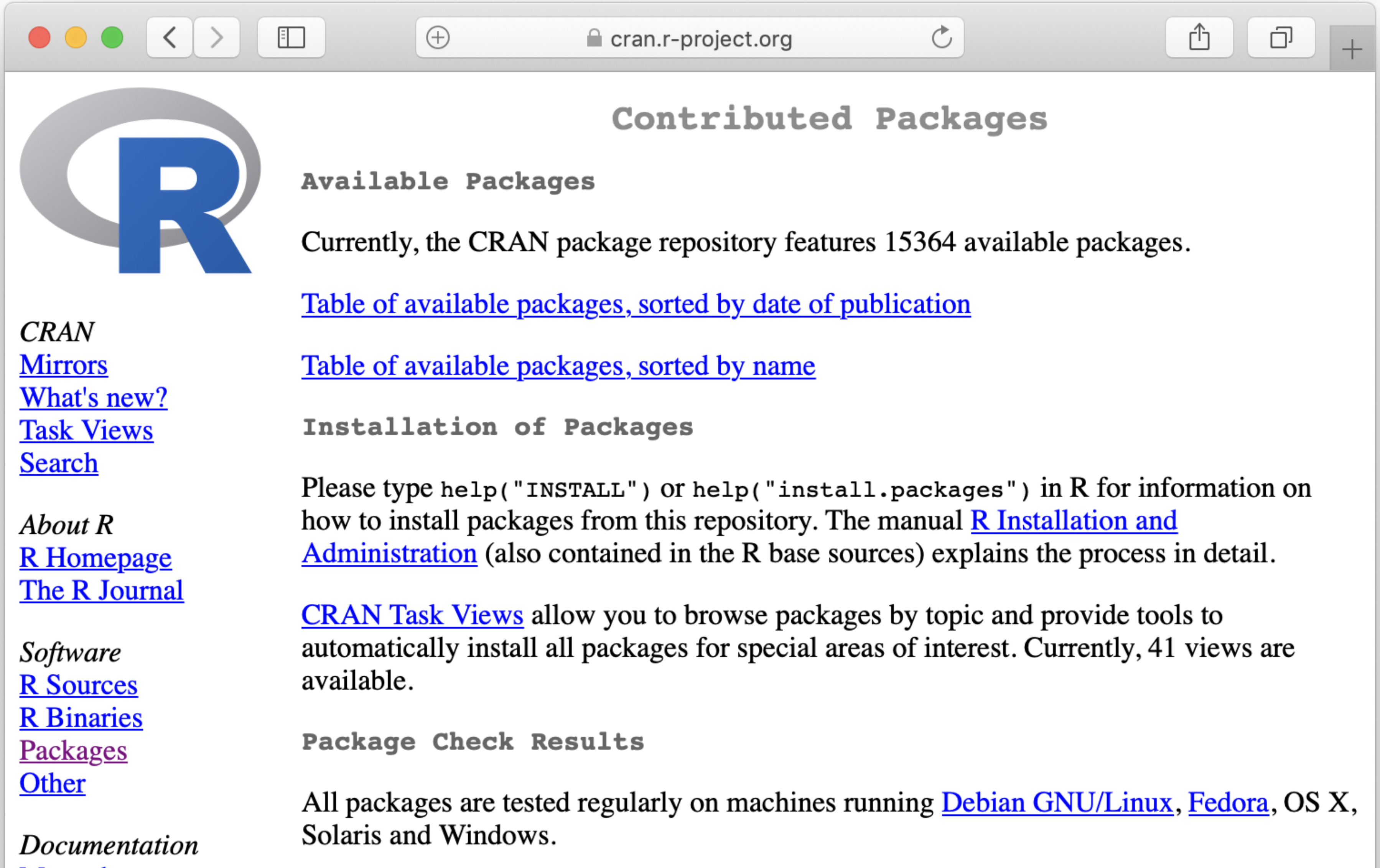
Rotating and spacing axis labels in ggplot2

Asked 10 years, 10 months ago Active 18 days ago Viewed 768k times

I have a plot where the x-axis is a factor whose labels are long. While probably not an ideal visualization, for now I'd like to simply rotate these labels to be vertical. I've figured this part out with the code below, but as you can see, the labels aren't totally visible.

Extending R through packages: There's a package for everything

R packages are available on CRAN (Comprehensive R Archive Network)



CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation

Contributed Packages

Available Packages

Currently, the CRAN package repository features 15364 available packages.

Table of available packages, sorted by date of publication

Table of available packages, sorted by name

Installation of Packages

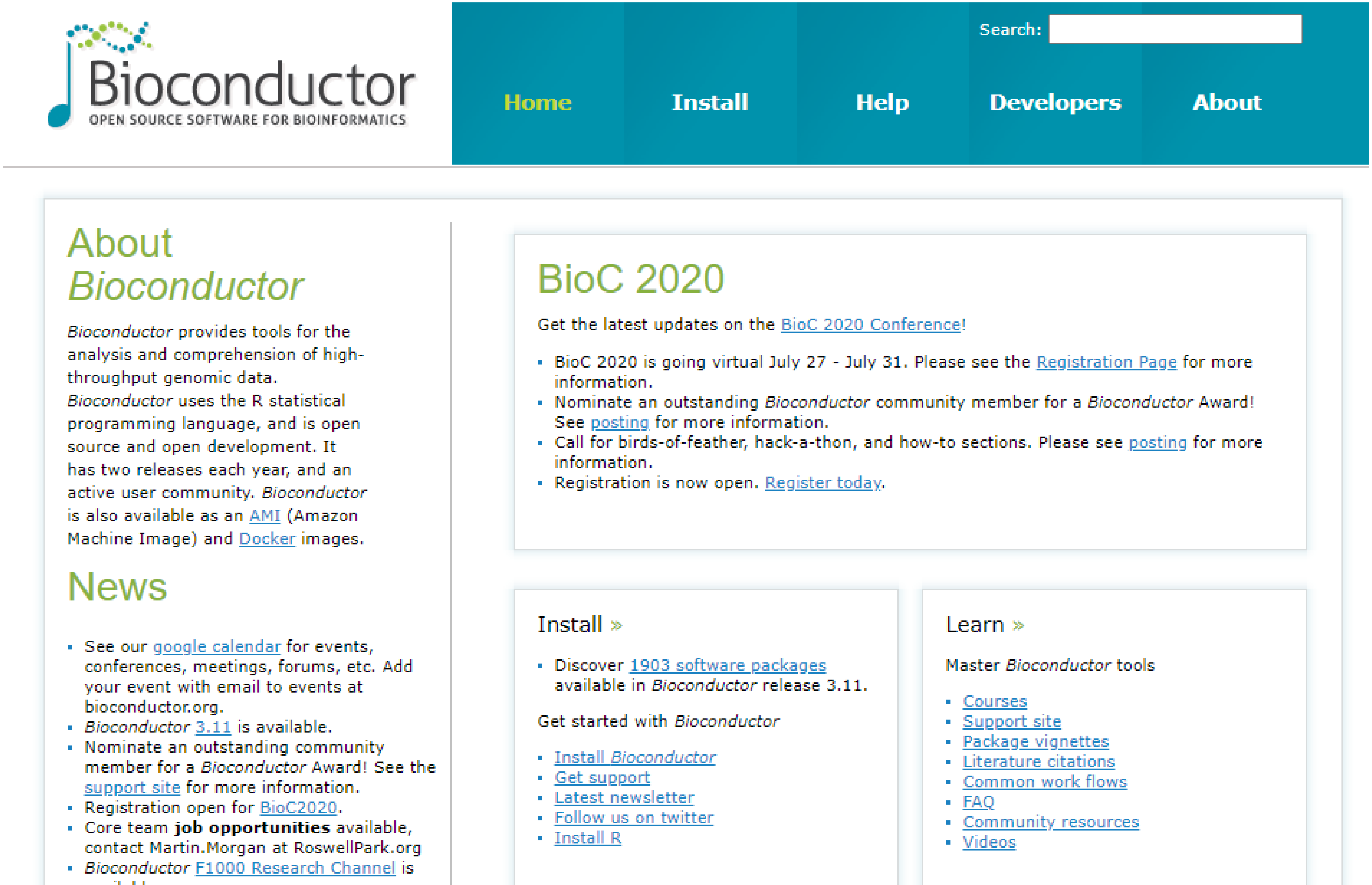
Please type `help("INSTALL")` or `help("install.packages")` in R for information on how to install packages from this repository. The manual R Installation and Administration (also contained in the R base sources) explains the process in detail.

CRAN Task Views allow you to browse packages by topic and provide tools to automatically install all packages for special areas of interest. Currently, 41 views are available.

Package Check Results

All packages are tested regularly on machines running Debian GNU/Linux, Fedora, OS X, Solaris and Windows.

Bio-specific R packages are available on Bioconductor



About *Bioconductor*

Bioconductor provides tools for the analysis and comprehension of high-throughput genomic data. *Bioconductor* uses the R statistical programming language, and is open source and open development. It has two releases each year, and an active user community. *Bioconductor* is also available as an AMI (Amazon Machine Image) and Docker images.

News

- See our google calendar for events, conferences, meetings, forums, etc. Add your event with email to events at bioconductor.org.
- *Bioconductor* 3.11 is available.
- Nominate an outstanding community member for a *Bioconductor* Award! See the support site for more information.
- Registration open for BioC2020.
- Core team **job opportunities** available, contact Martin.Morgan at RoswellPark.org
- *Bioconductor* F1000 Research Channel is

BioC 2020

Get the latest updates on the BioC 2020 Conference!

- BioC 2020 is going virtual July 27 - July 31. Please see the Registration Page for more information.
- Nominate an outstanding *Bioconductor* community member for a *Bioconductor* Award! See posting for more information.
- Call for birds-of-feather, hack-a-thon, and how-to sections. Please see posting for more information.
- Registration is now open. Register today.

Install »

- Discover 1903 software packages available in *Bioconductor* release 3.11.

Get started with *Bioconductor*

- Install *Bioconductor*
- Get support
- Latest newsletter
- Follow us on twitter
- Install R

Learn »

Master *Bioconductor* tools

- Courses
- Support site
- Package vignettes
- Literature citations
- Common work flows
- FAQ
- Community resources
- Videos