

Base R

Cheatsheet

Getting Help

Accessing the help files

`?mean`

Get help of a particular function.

`help.search('weighted mean')`

Search the help files for a word or phrase.

`help(package = 'dplyr')`

Find help for a package.

More about an object

`str(iris)`

Get a summary of an object's structure.

`class(iris)`

Find the class an object belongs to.

Using Packages

`install.packages('dplyr')`

Download and install a package from CRAN.

`library(dplyr)`

Load the package into the session, making all its functions available to use.

`dplyr::select`

Use a particular function from a package.

`data(iris)`

Working Directory

`getwd()`

Find the current working directory (where inputs are found and outputs are sent).

`setwd('C://file/path')`

Change the current working directory.

Use projects in RStudio to set the working directory to the folder you are working in.

Vectors

Creating Vectors

| | | |
|--------------------------------|--------------------------|-----------------------------|
| <code>c(2, 4, 6)</code> | <code>2 4 6</code> | Join elements into a vector |
| <code>2:6</code> | <code>2 3 4 5 6</code> | An integer sequence |
| <code>seq(2, 3, by=0.5)</code> | <code>2.0 2.5 3.0</code> | A complex sequence |
| <code>rep(1:2, times=3)</code> | <code>1 2 1 2 1 2</code> | Repeat a vector |
| <code>rep(1:2, each=3)</code> | <code>1 1 1 2 2 2</code> | Repeat elements of a vector |

Vectors Functions

| | |
|-----------------------|------------------------|
| <code>sort(x)</code> | <code>rev(x)</code> |
| Return x sorted. | Return x reversed. |
| <code>table(x)</code> | <code>unique(x)</code> |
| See counts of values. | See unique values. |

Selecting Vector Elements

By Position

| | |
|-------------------------|----------------------------------|
| <code>x[4]</code> | The fourth element. |
| <code>x[-4]</code> | All but the fourth. |
| <code>x[2:4]</code> | Elements two to four. |
| <code>x[-(2:4)]</code> | All elements except two to four. |
| <code>x[c(1, 5)]</code> | Elements one and five. |
| <code>head(x, 3)</code> | The first three elements. |
| <code>tail(x, 3)</code> | The last three elements. |

By Value

| | |
|-----------------------------------|--------------------------------|
| <code>x[x == 10]</code> | Element which are equal to 10. |
| <code>x[which(x == 10)]</code> | Element which are equal to 10. |
| <code>x[x < 0]</code> | All elements less than zero. |
| <code>x[x %in% c(1, 2, 5)]</code> | Elements in the set {1, 2, 5}. |

Named Vectors

| | |
|-------------------------|----------------------------|
| <code>x['apple']</code> | Element with name 'apple'. |
|-------------------------|----------------------------|

Testing Vector Elements

| | |
|-----------------------------|------------------------------|
| <code>all(x == 5)</code> | All elements are equal to 5? |
| <code>any(x < 10)</code> | An element is less than 10? |

Programming

While Loop

```
while (condition) {  
  Do something  
}
```

Example

```
while (i < 5) {  
  print(i)  
  i <- i + 1  
}
```

For Loop

```
for (variable in sequence) {  
  Do something  
}
```

Example

```
for (i in 1:4) {  
  j <- i + 10  
  print(j)  
}
```

If Statement

```
if (condition) {  
  Do something  
} else {  
  Do something  
}
```

Example

```
if (i > 3) {  
  print('Yes')  
} else {  
  print('No')  
}
```

Functions

```
func_name <- function(var) {  
  Do something  
  
  return(new_variable)  
}
```

Example

```
square <- function(x) {  
  squared <- x*x  
  
  return(squared)  
}
```

Reading and Writing Data

Also see the **readr** package.

| Input | Output | Description |
|--|--|---|
| <code>df <- read.table('file.txt')</code> | <code>write.table(df, 'file.txt')</code> | Read and write a delimited text file. |
| <code>df <- read.csv('file.csv')</code> | <code>write.csv(df, 'file.csv')</code> | Read and write a comma separated value file. This is a special case of <code>read.table</code> / <code>write.table</code> . |
| <code>load('file.RData')</code> | <code>save(df, file = 'file.RData')</code> | Read and write a n R data file, a file type special for R. |

| | | | | | | |
|------------|-----------------------|------------|-----------------------------|--------------|------------------------|--------------------------|
| Conditions | <code>a == b</code> | Are equal | <code>a > b</code> | Greater than | <code>a >= b</code> | Greater than or equal to |
| | <code>a != b</code> | Not equal | <code>a < b</code> | Less than | <code>a <= b</code> | Less than or equal to |
| | <code>is.na(a)</code> | Is missing | <code>is.null(a)</code> | Is null | | |
| | <code>c e</code> | c or e | <code>c && y</code> | c and y | | |

Types

Converting between common data types in R.
Can always go from a higher value in the table to a lower value.

| | | |
|--------------|-----------------------------------|--|
| as.logical | TRUE, FALSE, TRUE | Boolean values (TRUE or FALSE) |
| as.numeric | 1, 0, 1 | Integer or floating point numbers |
| as.integer | 1, 0, 1 | Integers |
| as.character | '1', '0', '1' | Character strings. Generally preferred to factors |
| as.factor | '1', '0', '1' levels: '1', '0' | Character strings with preset levels. Needed for some statistical models |

Maths Functions

| | | | |
|--------------|---------------------------------|-------------|-------------------------|
| log(x) | Natural log. | sum(x) | Sum. |
| exp(x) | Exponential. | mean(x) | Mean. |
| max(x) | Largest element. | median(x) | Median. |
| min(x) | Smallest element. | quantile(x) | Percentage quantiles. |
| ceiling(x) | Round to the next integer. | sd(x) | The standard deviation. |
| floor(x) | Round to the previous integer. | var(x) | The variance. |
| round(x, n) | Round to n decimal places. | rank(x) | Rank of elements. |
| signif(x, n) | Round to n significant figures. | cor(x, y) | Correlation. |

Variable Assignment

```
> a <- 'apple'
> a
[1] 'apple'
```

The Environment

| | |
|-----------------|--|
| ls() | List all variables in the environment. |
| rm(x) | Remove x from the environment. |
| rm(list = ls()) | Remove all variables from the environment. |

You can use the environment panel in RStudio to browse variables in your environment.

Lists

```
l <-list(x = 1:5, y = c('a', 'b'))
```

A list is a collection of elements which can be of different types.

| | | | |
|----------------------|---------------------------------------|-------------------|-------------------------------------|
| <code>l[[2]]</code> | <code>l[1]</code> | <code>l\$x</code> | <code>l['y']</code> |
| Second element of l. | New list with only the first element. | Element named x. | New list with only element named y. |

Miscellaneous

Arithmetics

```
16 = 3*5 + 1
16 %/% 3      Quotient (result 5).
16 %% 3      Remainder (result 1).
```

Permutations

```
sample(x, size, replace = F)  Give a sample of the specified size from the elements of x.
sample(c(1:5), 10, replace = TRUE)
sort(x, decreasing = F)      Sort a vector or factor.
sort(c(5, 1, 7, 3)) (Result 1 3 5 7)
order(x, decreasing = F)     Returns a permutation rearranging x.
order(c(5, 1, 7, 3)) (Result 3 1 4 2)
```

Function tests

```
replicate(n, f(x))           Execute n times the function f(x).
replicate(10, exp(1000000))
system.time(f(x))            Return the CPU times used to compute f(x).
system.time(exp(1000000))
```

Memo

Strings

Also see the **stringr** package.

| | |
|--|---|
| <code>cat(x, y, sep = '')</code> | Join and print multiple vectors together. |
| <code>cat(x, collapse = '')</code> | Join and print elements of a vector together. |
| <code>grep(pattern, x)</code> | Find regular expression matches in x. |
| <code>gsub(pattern, replace, x)</code> | Replace matches in x with a string. |
| <code>toupper(x)</code> | Convert to uppercase. |
| <code>tolower(x)</code> | Convert to lowercase. |
| <code>nchar(x)</code> | Number of characters in a string. |
| <code>utf8ToInt(x)</code> | ASCII code of character x. |
| <code>intToUtf8(x)</code> | Character with ASCII code x. |

Factors

| | |
|--|---|
| <code>factor(x)</code> | <code>cut(x, breaks = 4)</code> |
| Turn a vector into a factor. Can set the levels of the factor and the order. | Turn a numeric vector into a factor by 'cutting' into sections. |

Statistics

| | | |
|--|---|--|
| <code>lm(y ~ x, data=df)</code> Linear model. | <code>t.test(x, y)</code> Perform a t-test for difference between means. | <code>prop.test</code> Test for a difference between proportions. |
| <code>glm(y ~ x, data=df)</code> Generalized linear model. | <code>pairwise.t.test</code> Perform a t-test for paired data. | <code>aov</code> Analysis of variance. |
| <code>summary</code> or <code>fivenum</code> Get more detailed information out a model. | | |

Distributions

| | Random Variates | Density Function | Cumulative Distribution | Quantile |
|----------|---------------------|---------------------|-------------------------|---------------------|
| Normal | <code>rnorm</code> | <code>dnorm</code> | <code>pnorm</code> | <code>qnorm</code> |
| Poisson | <code>rpois</code> | <code>dpois</code> | <code>ppois</code> | <code>qpois</code> |
| Binomial | <code>rbinom</code> | <code>dbinom</code> | <code>pbinom</code> | <code>qbinom</code> |
| Uniform | <code>runif</code> | <code>dunif</code> | <code>punif</code> | <code>qunif</code> |

Plotting

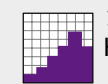
Also see the **ggplot2** package.



`plot(x)`
Values of x in order.



`plot(x, y)`
Values of x against y.



`hist(x)`
Histogram of x.