

Base R

Cheat Sheet

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

Vectors Functions

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

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'.
-------------------------	----------------------------

Programming

For Loop

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

Example

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

While Loop

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

Example

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

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/write.table</code> .
<code>load('file.RData')</code>	<code>save(df, file = 'file.RData')</code>	Read and write an 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>is.na(a)</code>	Is missing
<code>a != b</code>	Not equal	<code>a < b</code>	Less than	<code>a <= b</code>	Less than or equal to	<code>is.null(a)</code>	Is null

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.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.
(x, n)	Round to n decimal places.	rank(x)	Rank of elements.
(x, n)	Round to n significant figures.	var(x)	The variance.
(x, y)	Correlation.	sd(x)	The standard deviation.

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.
```

l[[2]]	l[1]	l\$x	l['y']
Second element of l.	New list with only the first element.	Element named x.	New list with only element named y.

Matrices

	m[2,] - Select a row	t(m)
	m[, 1] - Select a column	Transpose
	m[2, 3] - Select an element	m %*% n
		Matrix Multiplication
		solve(m, n)
		Find x in m*x = n

Data Frames

Also see the **dplyr** package.

```
df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))
A special case of a list where all elements are the same length.
```

x	y
1	a
2	b
3	c

List subsetting



Understanding a data frame

```
view(df) See the full data frame.
head(df) See the first 6 rows.
```

Matrix subsetting

df[, 2]		nrow(df)
		Number of rows.
df[2,]		ncol(df)
		Number of columns.
df[2, 2]		dim(df)
		Number of columns and rows.

cbind - Bind columns.



rbind - Bind rows.



Strings

Also see the **stringr** package.

paste(x, y, sep = '')	Join multiple vectors together.
paste(x, collapse = '')	Join elements of a vector together.
grep(pattern, x)	Find regular expression matches in x.
gsub(pattern, replace, x)	Replace matches in x with a string.
toupper(x)	Convert to uppercase.
tolower(x)	Convert to lowercase.
nchar(x)	Number of characters in a string.

Factors

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

Statistics

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

Distributions

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

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.
--	---------	-----------------------	--	------------	------------------------	--	---------	-----------------

Dates

See the **lubridate** package.