Cheatography

R Cheat Sheet

by [deleted] via cheatography.com/3687/cs/2469/

General

Get Help

? <Object/Function>

Find the working directory

getwd()

Setting Working directroy

setwd("~/specdata")

List files in working dir

dir()

Load code file into workspace

source("file.R")

Find the type of an object

class(my_vector)

List objects in workspace

ls()

Change page width

options(width = 160)

Operators

Assignement	var <- <New value>
Compare two objects	identical(obj1, obj2)
Equality	var1 == var2

Special Values

NA	value is **Not Available**
NaN	Not a Number
Inf	Infinity
T	True
F	False

Debugging

`traceback`

Prints function call stack

`debug( <fn> )`

Flags a function for "debug" mofr which allows you to step through execution of a function one line at a time

`browser`

Suspends the execution of a function, and outs the function in debug mode. n- next

`trace`

Allows you to insert debugging code into a function

`recover`

Allows you to modify the error behaviour so that you can browse the function call statck

Subsetting Vectors

First 10 elements	x[1:10]
Vector of all NAs	x[is.na(x)]
Vector of real values	x[!is.na(x)]
Values greater than 0	y[y > 0]
Combine conditions	x[!is.na(x) & x > 0]
3rd, 5th, 7th elements of x	x[c(3,5,7)]
All but the 2nd and 10th (neg)	x[c(-2, -10)] or x[-c(2,10)]
Access element by label	vect["bar"] or vect[c-("foo", "bar")]

Index vectors come in four different flavors -- logical vectors, vectors of positive integers, vectors of negative integers, and vectors of character strings

Vectors

Concatinate function

patients <- c("Bill","Gina","Kelly","Sean")

Matrices

Help on matrix type

? matrix

Add dimensions to vector to make a matrix

dim(my_vector) <- c(4, 5)

View dimesions of a matrix

dim(my_matrix)

View dimesions of a matrix

attributes(my_matrix)

Create a matrix. (4x5 containing 1-> 20)

my_matrix2 <- matrix(1:20,4,5)

+ Matrices can only contain a **single class** of data.
+ The first number is the number of rows and the second is the number of columns.

Data Frames

Create a data frame from avector and matrix

```
my_data <- data.frame(pati-
ents, my_matrix)
```

Add columns name to data frame

```
colnames(my_data) <- cnames-
_vector
```

Select rows based on column value

```
frame[ frame$col == "val", ]
```

Select columns by position

`frame[, 1:4]` cols 1 to 4

http://www.r-tutor.com/r-introduction/data-frame/data-frame-row-slice

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Cheatography

R Cheat Sheet

by [deleted] via cheatography.com/3687/cs/2469/

Conversion

To number	as.number(x)
To boolean (logical)	as.logical(x)
To comples number	as.complex()

Reading Data

```
# Create empty data frame
  data <- data.frame()

  #Readfiles id is vector of
integers
  for ( i in id ){
    infile = sprintf("%s/%-
03d.csv", directory, i)
    data <- rbind(data,read.csv(
infile ))
  }
  head(data)
```

IF statement

```
if(<condition>) {
    ## do something
} else {
    ## do something else
}
if(<condition1>) {
    ## do something
} else if(<condition2>) {
    ## do something different
} else {
    ## do something different
}
```

For Statement

```
for(i in 1:10) {
    print(i)
}
```

While Statemement

```
count <- 0
while(count < 10) {
    print(count)
    count <- count + 1
}
```

Repeat Statement

```
x0 <- 1
tol <- 1e-8
repeat {
    x1 <- computeEstimate()
    if(abs(x1 - x0) < tol) {
        break
    } else {
        x0 <- x1
    }
}
```

next,return

```
for(i in 1:100) {
    if(i <= 20) {
        ## Skip the first 20
iterations
        next
    }
    ## Do something here
}
```

next is used to skip an iteration of a loop

Loop functions

lapply	Loop over a list and evaluate a function on each element
sapply	Same as lapply but try to simplify the result
tapply	Apply a function over the margins of an array

Loop functions (cont)

mapply	Multivariate version of lapply
apply	Used to evaluate a function (often an anonymous one) over the margins of an array.
rowSums	apply(x,1,sum)
rowMeans	apply(x,1,mean)
colSums	apply(x,2,sum)
colMeans	apply(x,2,mean)

```
x<- list( a = 1:5, b= rnorm(10))
lapply(x,mean)
```

An annonymous fn for extracting the 1st col of each matrix

```
>lapply(x,function(elt) elt[,1])
```


Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com