The R Book

Chapter 2: Essentials of the R Language

Session 2

Functions in R

?function

Screen prompt
> log(42/7.3)

Sometimes more than one line
End each line with a trailing comma, operator, parenthesis

```
> 5+6+3+6+4+2+4+8+
+
    3+2+7
```

Multiple expressions on a single line
separated by semi-colons:

```
2+3; 5*7; 3-7
```

Built-in Functions

The log function
gives logs to the base e (e=2,718282)

log(10) → 2.302585

Inverse function
exp(1) → 2.718282

logs to the base 10
log10(6) → 0.7781513

log to base 3 of 9
log(9,3)

Built-in Functions

Function	Meaning
log(x)	log to base e of x
exp(x)	antilog of x (e^x)
log(x,n)	log to base n of x
log10(x)	log to base 10 of x
sqrt(x)	square root of x
factorial(x)	$x!$
choose(n,x)	binomial coefficients $n!/(x!\ (n-x)!)$
gamma(x)	$\Gamma(x)$, for real x $(x-1)!$, for integer x
lgamma(x)	natural log of $\Gamma(x)$
floor(x)	greatest integer $< x$
ceiling(x)	smallest integer $> x$
trunc(x)	closest integer to x between x and 0 trunc(1.5) $= 1$, trunc(-1.5) $= -1$ trunc is like floor for positive values and like ceiling for negative values
round(x, digits=0)	round the value of x to an integer
signif(x, digits=6)	give x to 6 digits in scientific notation
runif(n)	generates n random numbers between 0 and 1 from a uniform distribution
cos(x)	cosine of x in radians
sin(x)	sine of x in radians
tan(x)	tangent of x in radians
acos(x), asin(x), atan(x)	inverse trigonometric transformations of real or complex numbers
acosh(x), asinh(x), atanh(x)	inverse hyperbolic trigonometric transformations of real or complex numbers
abs(x)	the absolute value of x, ignoring the minus sign if there is one

x <- runif(500000);hist(x)

Numbers with Exponents

For very big numbers or very small numbers:

1.2e3 1200 'three decimal points to the right'
1.2e-2 0.012 'two decimal points to the left'
3.9+4.5i complex number

Modulo and Integer Quotients

Integer quotients %/%
119/13 → 9.153846
119%/%13 → 9

Remainders %% (modulo, whats left)
119 %% 13 → 2

test if one number is an exact multiple of some other number

15421 %% 7 == 0 → TRUE

Rounding

floor(5.7) → 5

ceiling(5.7) → 6

Build your own function rounded :
rounded<-function(x) floor(x+0.5)

Infinity and Things that Are Not a Number (NaN)

3/0 → Inf

-12/0 → -Inf

Calculations involving infinity can be evaluated: for instance,

exp(-Inf) → 0

0/Inf → 0

(0:3)^Inf → 0, 1, Inf, Inf

NaN (‘not a number’)
0/0 → NaN

Inf-Inf → NaN
Inf/Inf → NaN

built-in tests
is.finite(10) → TRUE

is.infinite(10) → FALSE

is.infinite(Inf) → TRUE

Missing values NA (not available)

x<-c(1:8,NA)

mean(x) → NA

na.rm=TRUE argument

mean(x, na.rm=TRUE) → 4.5

Locating missing values within a vector using seq or which

vmv<-c(1:6,NA,NA,9:12)
vmv → 1 2 3 4 5 6 NA NA 9 10 11 12

seq(along=vmv)[is.na(vmv)] → 7 8

which(is.na(vmv)) → 7 8

Turn NA into 0's

vmv[is.na(vmv)]<- 0
vmv → 1 2 3 4 5 6 0 0 9 10 11 12

vmw<-c(1:6,NA,NA,9:12)

ifelse(is.na(vmw),0,vmw) → 1 2 3 4 5 6 0 0 9 10 11 12
vmw → 1 2 3 4 5 6 NA NA 9 10 11 12

Assignment

x<-5 or x=5

x → 5

Operators

+ - */%% ^	arithmetic
> >= < <= == !=	relational
!&|	logical
~	model formulae
<- ->	assignment
$	list indexing (the ‘element name’ operator)
:	create a sequence

Creating a Vector

Vectors are variables with one or more values of the same type:

- logical true, false
- integer 1, 2, 3, ...
- real 1.23, 2.43, 3.56, ...
- complex 3+2*i, ...
- string Virginie, Alexandre, Juliette, …

In R : Variables are vectors with the length 1

y<- -4.3

Vectors can have the length 0

z<-y[-1]
length(z) → 0

Assigning values to a Vector

integer values 10 to 16 using : (colon)

```
y <- 10:16
```

using the concatenation function c

```
y <- c(10, 11, 12, 13, 14, 15, 16)
```

from the keyboard using scan:

```
y <- scan()
1: 10 [enter]
2: 11 [enter]
3: 12 [enter]
4: 13 [enter]
5: 14 [enter]
6: 15 [enter]
7: 16 [enter]
8: [enter]
```

from an external file, using read.table (later ...)

length: « the number of numbers of a vector »

When vectors are created by calculation from other vectors,

- the new vector will be as long as the longest vector
- the shorter variable is recycled

```
A<-1:10
B<-c(2,4,8)
C<-A*B
```

Message d'avis :
In A * B : **la taille d'un objet plus long n'est pas multiple de la taille d'un objet plus court**

C → 2 8 24 8 20 48 14 32 72 20

1x2, 4x2, 7x2, 10x2
2x4, 5x4, 8x4,
3x8, 6x8, 9x8,

Named Elements within Vectors

Labeling values in a vector

counts<-c(25,12,7,4,6,2,1,0,2) (nine entries)

names(counts)<-0:8

counts →

```
 0  1  2  3  4  5  6  7  8
25 12  7  4  6  2  1  0  2
```

Remove labels with as.vector

counts<-as.vector(counts)

counts →

```
25 12  7  4  6  2  1  0  2
```

Table 2.2 Vector Functions in R

Operation	Meaning
max(x)	maximum value in x
min(x)	minimum value in x
sum(x)	total of all the values in x
mean(x)	arithmetic average of the values in x
median(x)	median value in x
range(x)	vector of min(x) and max(x)
var(x)	sample variance of x
cor(x,y)	correlation between vectors x and y
sort(x)	a sorted version of x
rank(x)	vector of the ranks of the values in x
order(x)	an integer vector containing the permutation to sort x into ascending order
quantile(x)	vector containing the minimum, lower quartile, median, upper quartile, and maximum of x
cumsum(x)	vector containing the sum of all of the elements up to that point
cumprod(x)	vector containing the product of all of the elements up to that point
cummax(x)	vector of non-decreasing numbers which are the cumulative maxima of the values in x up to that point
cummin(x)	vector of non-increasing numbers which are the cumulative minima of the values in x up to that point
pmax(x,y,z)	vector, of length equal to the longest of x, y or z, containing the maximum of x, y or z for the ith position in each
pmin(x,y,z)	vector, of length equal to the longest of x, y or z, containing the minimum of x, y or z for the ith position in each
colMeans(x)	column means of dataframe or matrix x
colSums(x)	column totals of dataframe or matrix x
rowMeans(x)	row means of dataframe or matrix x
rowSums(x)	row totals of dataframe or matrix x

Summary Information from Vectors by Groups

In the terminal / Console :

data<-read.table("/home/michael/Documents/.../daphnia.txt",header=T)
data → ...
names(data) → "Growth.rate" "Water" "Detergent" "Daphnia"
tapply(Growth.rate,Detergent,mean) → error

attach(data) (you must « attach » the data to R in order to work with it)
tapply(Growth.rate,Detergent,mean) →

```
  BrandA   BrandB   BrandC   BrandD
3.884832 4.010044 3.954512 3.558231
```

the mean growth rate for each detergent

> tapply(Growth.rate,list(Detergent,Water),mean) →

```
            Tyne     Wear
BrandA 3.661807 4.107857
BrandB 3.911116 4.108972
BrandC 3.814321 4.094704
BrandD 3.356203 3.760259
```

the mean growth rate for each detergent and water type

Summary Information from Vectors by Groups

In the terminal / Console :

data<-read.table("/home/michael/Documents/.../daphnia.txt",header=T)
data → ...
names(data) → "Growth.rate" "Water" "Detergent" "Daphnia"
tapply(Growth.rate,Detergent,mean) → error

attach(data) (you must « attach » the data to R in order to work with it)

tapply(Growth.rate,Detergent,mean) →

```
  BrandA   BrandB   BrandC   BrandD
3.884832 4.010044 3.954512 3.558231
```

the mean growth rate for each detergent

> tapply(Growth.rate,list(Water,Daphnia),median) → `?????`

the median growth rate for water type and Daphnia clone

In RStudio :
Import Daphnia.txt (/home/michael/Documents/...daphnia.txt
attach(Daphnia)
...