R Reference Card

by Tom Short, EPRI PEAC, tshort@epri-peac.com 2004-11-07

Granted to the public domain. See www.Rpad.org for the source and latest version. Includes material from R for Beginners by Emmanuel Paradis (with permission).

Getting help
Most R functions have online documentation.

help(topic) documentation on topic
?topic id.
help.search("topic") search the help system
apropos("topic") the names of all objects in the search list matching the regular expression "topic"

help.start() start the HTML version of help
str(x) display the internal "str"ucture of an R object
summary(a) gives a "summary" of a, usually a statistical summary but it is
generic: meaning it has different operations for different classes of a
ls() shows objects in the search path; specify pat="*pat" to search on a pattern
ls.str() str() for each variable in the search path
dir() show files in the current directory
methods(a) shows S3 methods of class a
methods(class=class(a)) lists all the methods to handle objects of class a

data.frame(...) create a data frame of the named or unnamed arguments; data.frame(..., row.names=TRUE, col.names=TRUE) gives a "summary" of a
summary(a)

Variable conversion
as.array(x), as.data.frame(x), as.numeric(x), as.logical(x), as.character(x), ... convert type; for a complete list, use methods(as)

Variable information
is.na(x), is.null(x), is.array(x), is.data.frame(x), is.numeric(x), is.complex(x), is.character(x), ... test for type; for a complete list, use methods(is)

length(x) number of elements in x
dim(x) Retrieve or set the dimension of an object; dim(x) <- c(3,2)
dimnames(x) Retrieve or set the dimension names of an object
ncol(x) number of rows; NROW(x) is the same but treats a vector as a one-row matrix
ncol() and NCOL() id. for columns
class(x) get or set the class of x; class(x) <- "myclass" unclass(x) remove the class attribute of x
attr(x,which) get or set the attribute which of x
attributes(obj) get or set the list of attributes of obj

Data selection and manipulation
which.max(x) returns the index of the greatest element of x
which.min(x) returns the index of the smallest element of x
rev(x) reverses the elements of x
sort(x) sorts the elements of x in increasing order; to sort in decreasing order: rev(sort(x))
cut(x,breaks) divides x into intervals (factors); breaks is the number of cut intervals or a vector of cut points
match(x, y) returns a vector of the same length than x with the elements of x which are in y (NA otherwise)

which(x == a) returns a vector of the indices of x if the comparison operation is true (TRUE), in this example the values of x for which x[1] == a (the argument of this function must be a variable of mode logical)
choose(n, k) computes the combinations of k events among n repetitions = n!/((n-k)!k!)
na.omit(x) suppresses the observations with missing data (NA) (supersedes the corresponding line if x is a matrix or a data frame)
na.fail(x) returns an error message if x contains at least one NA

Indexing lists
x[n] list with elements n
x[[n]] nth element of the list
x[["name"]][] element of the list named "name"
x$name id.

Indexing matrices
x[i,j] element at row i, column j
x[i] row i
x[j] column j
x[i,j][] columns 1 and 3
x["name",] row named "name"

Indexing data frames (matrix indexing plus the following)
x["name"] column named "name"
x$name id.

Slicing and extracting data
Indexing vectors
x[n] nth element
all but the nth element
first n elements
elements from 1 to n
all elements
elements named "name"
elements greater than 3
elements between 3 and 5

elements in the given set

Slicing and extracting data
Indexing vectors
x[n] nth element
all but the nth element
first n elements
elements from 1 to n
all elements
elements named "name"
elements greater than 3
elements between 3 and 5

elements in the given set
unique(x) if x is a vector or a data frame, returns a similar object but with the duplicate elements suppressed

table(x) returns a table with the numbers of the different values of x (typically for integers or factors)

subset(x, ...) returns a selection of x with respect to criteria (..., typically comparisons: x$V1 < 10); if x is a data frame, the option select gives the variables to be kept or dropped using a minus sign

sample(x, size) resamples randomly and without replacement size elements in the vector x, the option replace = TRUE allows to resample with replacement

prop.table(x, margin=) table entries as fraction of marginal table

diff() lagged and iterated differences of vector x

convolve(x,y) applies linear filtering to a univariate time series or to each series separately of a multivariate time series

Many math functions have a logical parameter na.rm=FALSE to specify missing data (NA) removal.

Matrices

t(x) transpose

diag(x) diagonal

%*% matrix multiplication

solve(a,b) solves a \( a x = b \) for x

%\-1\% matrix inverse of a

rowsum(x) sum of rows for a matrix-like object; rowSums(x) is a faster version

colsum(x), colSums(x) id. for columns

rowMeans(x) fast version of row means

colMeans(x) id. for columns

Advanced data processing

apply(X, INDEX, FUN=) a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X

lapply(X, FUN) apply FUN to each element of the list X

tapply(X, INDEX, FUN=) apply FUN to each cell of a ragged array given by x with indexes INDEX

by(data, INDEX, FUN) apply FUN to data frame data subsetted by INDEX

merge(a,b) merge two data frames by common columns or row names

xtabs(a,b, data=x) a contingency table from cross-classifying factors

aggy(x,y,by=FUN) splits the data frame x into subsets, computes summary statistics for each, and returns the result in a convenient form; by is a list of grouping elements, each as long as the variables in x

stack(x, ..., simplify=) transform data available as separate columns in a data frame or list into a single column

unstack(x, ...) inverse of stack()

reshape(x, ...) reshapes a data frame between 'wide' format with repeated measurements in separate columns of the same record and 'long' format with the repeated measurements in separate records; use (direction="wide") or (direction="long")

Strings

paste(...) concatenate vectors after converting to character; sep= is the string to separate terms (a single space is the default); collapse= is an optional string to separate "collapsed" results

substr(x,start,stop) substrings in a character vector; can also assign, assign(substr(x, start, stop) \(<-\) value

strsplit(x,split) split x according to the substring split

grep(pattern,x) searches for matches to pattern within x; see ?regex

gsub(pattern, replacement, x) replacement of matches determined by regular expression matching sub() is the same but only replaces the first occurrence.

tolower(x) convert to lowercase
toupper(x) convert to uppercase

match(x,table) a vector of the positions of first matches for the elements of x among table

% in% table id. but returns a logical vector

pmatch(x,table) partial matches for the elements of x among table

nchar(x) number of characters

Dates and Times

The class Date has dates without times. POSIXct has dates and times, including time zones. Comparisons (e.g. >, seq(), and diffftime) are useful. Date also allows + and - .Date TimeClasses gives more information. See also package chron.

as.Date(s) and as.POSIXct(s) convert to the respective class; format(dt) converts to a string representation. The default string format is "2001-02-21". These accept a second argument to specify a format for conversion. Some common formats are:

%a, %A Abbreviated and full weekday name.

%b, %B Abbreviated and full month name.

%d Day of the month (01-31).

%m Hours (00-23).

%M Minutes (00-59).

%Y Year without century. Don’t use.

%y Year with century.

%Z Time zone as a character string (empty if not available).

Where leading zeros are shown they will be used on output but are optional on input. See strftime.

Plotting

plot(x) plot of the values of x (on the y-axis) ordered on the x-axis

plot(x, y) bivariate plot of x (on the x-axis) and y (on the y-axis)

hist(x) histogram of the frequencies of x

barplot(x) histogram of the values of x; use horiz=FALSE for horizontal bars

dotchart(x) if x is a data frame, plots a Cleveland dot plot (stacked plots line-by-line and column-by-column)

pie(x) circular pie-chart

scatterplot(x, y, ...) id. than plot() but the points with similar coordinates are drawn as flowers which petal number represents the number of points

stripchart(x) plot of the values of x on a line (an alternative to boxplot() for small sample sizes)

coplot(x’y | z) bivariate plot of x and y for each value or interval of values of z

interaction.plot (f1, f2, y) if f1 and f2 are factors, plots the means of y (on the y-axis) with respect to the values of f1 (on the x-axis) and of f2 (different curves); the option fun allows to choose the summary statistic of y (by default fun=mean)
In the normal Lattice formula, \( y \times |g1 \times g2 \) has combinations of optional conditioning variables \( g1 \) and \( g2 \) plotted on separate panels. Lattice functions take many of the same arguments as base graphics plus also data - the data frame for the formula variables and subset - for subsetting. Use panel to define a custom panel function (see apropos("panel") and ??lines). Lattice functions return an object of class trellis and have to be print-ed to produce the graph. Use print(xyplot(...)) inside functions where automatic printing doesn’t work. Use lattice.theme and lset to change Lattice defaults.

Optimization and model fitting

**optim(par, fn, method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN") general-purpose optimization; par is initial values, fn is function to optimize (normally minimize)**

\( \text{nlm}(f,p) \) minimize function \( f \) using a Newton-type algorithm with starting values \( p \)

\( \text{lm(formula)} \) fit linear models; formula is typically of the form \( \text{response} + \text{termA} + \text{termB} + \ldots \); use \( I(x*y) + I(x^2) \) for terms made of nonlinear components

\( \text{glm(formula,family=)} \) fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution; family is a description of the error distribution and link function to be used in the model; see ??family

\( \text{nls(formula)} \) fit nonlinear least-squares estimates of the nonlinear model parameters

\( \text{approx(x,y=)} \) linearly interpolate given data points; \( x \) can be an xy plotting structure

\( \text{spline(x,y=)} \) cubic spline interpolation

\( \text{loess(formula)} \) fit a polynomial surface using local fitting Many of the formula-based modeling functions have several common arguments: data - the data frame for the formula variables, subset - a subset of variables used in the fit, na.action - action for missing values: "na.fail", "na.omit", or a function. The following generics often apply to model fitting functions:

\( \text{predict} \) \( \{ \ldots \} \) predictions from \( \text{fit} \) based on input data

\( \text{df.residual} \) \( \{ \text{fit} \} \) returns the number of residual degrees of freedom

\( \text{coef} \) \( \{ \text{fit} \} \) returns the estimated coefficients (sometimes with their standard-errors)

\( \text{residuals} \) \( \{ \text{fit} \} \) returns the residuals

\( \text{deviance} \) \( \{ \text{fit} \} \) returns the deviance

\( \text{fitted} \) \( \{ \text{fit} \} \) returns the fitted values

\( \text{logLik} \) \( \{ \text{fit} \} \) computes the logarithm of the likelihood and the number of parameters

\( \text{AIC} \) \( \{ \text{fit} \} \) computes the Akaike information criterion or AIC

**Statistics**

\( \text{aov(formula)} \) analysis of variance model

\( \text{anova} \) \( \{ \ldots \} \) analysis of variance (or deviance) tables for one or more fitted model objects

\( \text{density} \) \( \{ x \} \) kernel density estimates of \( x \)

\( \text{binom.test()}, \text{pairwise.t.test()}, \text{power.t.test()}, \text{prop.test()}, \text{t.test()}, \ldots \) use help.search("*test")

**Distributions**

\( \text{rnorm(n, mean=0, sd=1)} \) Gaussian (normal)

\( \text{rexp(n, rate=1)} \) exponential

\( \text{rgamma(n, shape, scale=1)} \) gamma

\( \text{rpois(n, lambda)} \) Poisson

\( \text{rweibull(n, shape, scale=1)} \) Weibull

\( \text{rcauchy(n, location=0, scale=1)} \) Cauchy

\( \text{rbeta(n, shape1, shape2)} \) beta

\( \text{rt(n, df) } \) 'Student' \( \text{t} \)

\( \text{rf(n, df1, df2)} \) Fisher–Snedecor \( \text{F} \) \( \chi^2 \)

\( \text{rchisq(n, df)} \) Pearson

\( \text{rbinom(n, size, prob)} \) binomial

\( \text{rgeom(n, prob)} \) geometric

\( \text{rhyper(nn, m, n, k)} \) hypergeometric

\( \text{rlogis(n, location=0, scale=1)} \) logistic

\( \text{rnorm(n, meanlog=0, sdlog=1)} \) lognormal

\( \text{rnbinom(n, size, prob)} \) negative binomial

\( \text{runif(n, min=0, max=1)} \) uniform

\( \text{rwilcox(nn, m, n)} \), \( \text{rsignrank(nn, n)} \) Wilcoxon’s statistics

All these functions can be used by replacing the letter \( r \) with \( d \), \( p \) or \( q \) to get, respectively, the probability density (\( dfunc(x, \ldots) \)), the cumulative probability density (\( qfunc(x, \ldots) \)), and the value of quantile (\( qfunc(p, \ldots) \), with \( 0 < p < 1 \)).

**Programming**

\( \text{function( arglist ) expr} \) function definition

\( \text{return(value)} \)

\( \text{if(cond) expr} \)

\( \text{if(cond) cons.expr else alt.expr} \)

\( \text{for(var in seq) expr} \)

\( \text{while(cond) expr} \)

\( \text{repeat expr} \)

\( \text{break} \)

\( \text{next} \)

Use braces {} around statements

\( \text{ifelse(test, yes, no)} \) a value with the same shape as \( test \) filled with elements from either \( yes \) or \( no \)

\( \text{do.call(funnname, args)} \) executes a function call from the name of the function and a list of arguments to be passed to it