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

Input and output
load() load datasets written with save
save(x) saves all objects
save(file,...) saves all objects
write.table(x,file=",row.names=TRUE,col.names=TRUE,
prints its arguments; generic, meaning it can have different operations for different classes of a
methods class a
ls() show 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() shows S3 methods of a
methods(class=class(a)) lists all the methods to handle objects of class a

Data creation

rep(x,times) replicate x times;
rep(c(1,2,3),2) is 1 2 3 1 2 3;
rep(c(1,2,3),each=2) is 1 2 3 1 2 3;
data.frame(...) create a data frame of the named or unnamed arguments:
data.frame(v=1:4,ch=c("a","B","c","d"),n=10)
shorter vectors are recycled to the length of the longest
list(...) create a list of the named or unnamed arguments:
list(a=c(1,2),b="hi",c=31);print()

Data selection and manipulation

cbind(...) combine arguments by rows for matrices, data frames, and others
rbind(...) combine arguments by rows for matrices, data frames, and others
Indexing vectors
x[n] n-th element
x[-n] all but the n-th element
x[:-n] elements from n to the end
x[-(1:n)] elements from n+1 to the end
x["name"] name of element
x[1:3] all elements greater than 3
x[3:5] all elements between 3 and 5
x[1:2 & 3] elements in the given set
x["name"] element of the list named "name"
x$sname id.

Variable conversion

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

Variable information

is.s4(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
nrow(x) and NCOL(x) 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.min(x) returns the index of the greatest element of x
which.max(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 l 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 elements among n repetitions
na.omit(x) suppresses the observations with missing data (NA) (suppresses 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
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>unique(x)</code></td>
<td>Returns a vector or a data frame, returns a similar object but with the duplicate elements suppressed</td>
</tr>
<tr>
<td><code>table(x)</code></td>
<td>Returns a table with the numbers of the different values of x (typically for integers or factors)</td>
</tr>
<tr>
<td><code>subset(x, ...)</code></td>
<td>Returns a selection of elements in the vector x, the option <code>select</code> gives the variables to be kept or dropped using a minus sign</td>
</tr>
<tr>
<td><code>sample(x, size)</code></td>
<td>Resamples randomly and without replacement size elements from the data frame x, the option <code>replace = TRUE</code> allows to resample with replacement</td>
</tr>
<tr>
<td><code>prop.table(x, margin=)</code></td>
<td>Table entries as fraction of marginal table</td>
</tr>
<tr>
<td><code>arg(x)</code></td>
<td>Angle in radians of the complex number</td>
</tr>
<tr>
<td><code>modulus; Im(x)</code></td>
<td>Real part of a complex number</td>
</tr>
<tr>
<td><code>a vector which id. for the maximum</code></td>
<td><code>pmax(x,y,...)</code> computes the logarithm of <code>log(x, base)</code></td>
</tr>
<tr>
<td><code>var(x)</code></td>
<td>Or <code>cor(x)</code> product of the elements of x</td>
</tr>
<tr>
<td><code>mean(x)</code></td>
<td>Mean of the elements of x</td>
</tr>
<tr>
<td><code>median(x)</code></td>
<td>Weighted mean of the elements of x</td>
</tr>
<tr>
<td><code>sd(x)</code></td>
<td>Standard deviation of x</td>
</tr>
<tr>
<td><code>cor(x)</code></td>
<td>Correlation matrix of x if it is a matrix or a data frame, the correlation-covariance matrix is calculated</td>
</tr>
<tr>
<td><code>quantile(x, probs=)</code></td>
<td>Sample quantiles corresponding to the given probabilities</td>
</tr>
<tr>
<td><code>weighted.mean(x, w)</code></td>
<td>weighted mean of x with weights w</td>
</tr>
<tr>
<td><code>rank(x)</code></td>
<td>Ranks of the elements of x</td>
</tr>
<tr>
<td><code>var(x)</code></td>
<td>Variance of the elements of x (calculated on n - 1); if x is a matrix or a data frame, the variance-covariance matrix is calculated</td>
</tr>
<tr>
<td><code>sd(x)</code></td>
<td>Standard deviation of x</td>
</tr>
<tr>
<td><code>cor(x)</code></td>
<td>Correlation matrix of x if it is a matrix or a data frame (if x is a vector)</td>
</tr>
<tr>
<td><code>var(x)</code></td>
<td>Variance of the columns of x and those of y if they are matrices or data frames</td>
</tr>
<tr>
<td><code>cor(x)</code></td>
<td>Linear correlation between x and y, or correlation matrix if they are matrices or data frames</td>
</tr>
<tr>
<td><code>round(x, n)</code></td>
<td>Rounds the elements of x to n decimals</td>
</tr>
<tr>
<td><code>log(x, base)</code></td>
<td>Computes the logarithm of x with base base</td>
</tr>
<tr>
<td><code>scale(x)</code></td>
<td>If x is a matrix, centers and reduces the data; to center only use the option <code>center=FALSE</code>, to reduce only <code>scale=FALSE</code> (by default <code>center=TRUE</code>, <code>scale=TRUE</code>)</td>
</tr>
<tr>
<td><code>pmin(x, y,...)</code></td>
<td>A vector which id. than the minimum of x1, y1, ...</td>
</tr>
<tr>
<td><code>pmax(x, y,...)</code></td>
<td>A vector whose element is the maximum of x1[1] to x1[i]</td>
</tr>
<tr>
<td><code>cumsum(x)</code></td>
<td>Cumulative sum of the elements of x</td>
</tr>
<tr>
<td><code>cumprod(x)</code></td>
<td>Cumulative product of the elements of x</td>
</tr>
<tr>
<td><code>cummin(x)</code></td>
<td>Cumulative minimum of the elements of x</td>
</tr>
<tr>
<td><code>cummax(x)</code></td>
<td>Cumulative maximum of the elements of x</td>
</tr>
<tr>
<td><code>union(x,y,...)</code></td>
<td>Unions the duplicate elements suppressed</td>
</tr>
<tr>
<td><code>intersect(x,y)</code></td>
<td>id. for columns</td>
</tr>
<tr>
<td><code>setdiff(x,y)</code></td>
<td>Returns a selection of elements in the vector x, the option <code>select</code> gives the variables to be kept or dropped using a minus sign</td>
</tr>
<tr>
<td><code>is.element(elset, &quot;set&quot;)&quot; set&quot; functions</code></td>
<td>Real part of a complex number</td>
</tr>
<tr>
<td><code>Im(x)</code></td>
<td>Imaginary part</td>
</tr>
<tr>
<td><code>Mod(x)</code></td>
<td>Modulus of abs(x) is the same</td>
</tr>
<tr>
<td><code>Arg(x)</code></td>
<td>Angle in radians of the complex number</td>
</tr>
<tr>
<td><code>Conj(x)</code></td>
<td>Complex conjugate</td>
</tr>
<tr>
<td><code>convolve(x, y)</code></td>
<td>Computes the several convolutions of two sequences</td>
</tr>
<tr>
<td><code>fft(x)</code></td>
<td>Fast Fourier Transform of an array</td>
</tr>
<tr>
<td><code>mvfft(x)</code></td>
<td>FFT of each column of a matrix</td>
</tr>
<tr>
<td><code>filter(x, filter)</code></td>
<td>Applies linear filtering to a univariate time series or to each series separately of a multivariate time series</td>
</tr>
<tr>
<td><code>apply(X, INDEX, FUN)</code></td>
<td>Applies linear filtering to a univariate time series</td>
</tr>
<tr>
<td><code>mean(x)</code></td>
<td>Mean of the elements of x</td>
</tr>
<tr>
<td><code>median(x)</code></td>
<td>Median of the elements of x</td>
</tr>
<tr>
<td><code>weighted.mean(x, w)</code></td>
<td>Weighted mean of the elements of x</td>
</tr>
<tr>
<td><code>median(x)</code></td>
<td>Median of the elements of x</td>
</tr>
<tr>
<td><code>rownames(x)</code></td>
<td>A vector which id. for columns</td>
</tr>
<tr>
<td><code>colnames(x)</code></td>
<td>Id. for the maximum</td>
</tr>
<tr>
<td><code>rowMeans(x)</code></td>
<td>Fast version of row means</td>
</tr>
<tr>
<td><code>colMeans(x)</code></td>
<td>Id. for columns</td>
</tr>
<tr>
<td><code>Advanced data processing</code></td>
<td><code>apply(X, INDEX, FUN=)</code> a vector or array or list of values obtained by applying a function FUN to margins (INDEX) of X</td>
</tr>
<tr>
<td><code>lapply(X, INDEX, FUN=)</code></td>
<td>Apply FUN to each cell of a ragged array given by x with indexes INDEX</td>
</tr>
<tr>
<td><code>by(data, INDEX, FUN)</code></td>
<td>Apply FUN to data frame data subsetted by INDEX</td>
</tr>
<tr>
<td><code>merge(a,b)</code></td>
<td>Merge a and b, merge two data frames by common columns or row names</td>
</tr>
<tr>
<td><code>xtabs(a, b, data=x)</code></td>
<td>Contingency table from cross-classifying factors</td>
</tr>
<tr>
<td><code>aggregate(x, by, FUN)</code></td>
<td>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</td>
</tr>
<tr>
<td><code>stack(x, ...)</code></td>
<td>Transform data available as separate columns in a data frame or list into a single column</td>
</tr>
<tr>
<td><code>unstack(x, ...)</code></td>
<td>Inverse of stack()</td>
</tr>
<tr>
<td><code>reshape(x, ...)</code></td>
<td>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=&quot;wide&quot;) or (direction=&quot;long&quot;)</td>
</tr>
<tr>
<td><code>Strings</code></td>
<td><code>paste(., ...)</code> 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 &quot;collapsed&quot; results</td>
</tr>
<tr>
<td><code>strsplit(x, split)</code></td>
<td>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</td>
</tr>
<tr>
<td><code>substr(x, start, stop) &lt;- value</code></td>
<td>String to separate terms (a single space is the default); collapse= is an optional string to separate &quot;collapsed&quot; results</td>
</tr>
<tr>
<td><code>plot(x)</code></td>
<td>Plot of the values of x on a line (an alternative to boxplot() for small sample sizes)</td>
</tr>
<tr>
<td><code>interaction.plot(f1, f2, y)</code></td>
<td>Bivariate plot of x and y for each value or interval of values of z</td>
</tr>
</tbody>
</table>

## Dates and Times

The class Date has dates without times. POSIXct has dates and times, including time zones. Comparisons (e.g. >), seq(), and difftime() are useful. Date also allows + and - .DateTimeClasses 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).
- `%h` Hours (00-23).
- `%H` Hours (01-12).
- `%j` Day of year (001-366).
- `%m` Month (01-12).
- `%m` Month (01-12).
- `%M` Minute (00-59).
- `%p` AM/PM indicator.
- `%s` Second as decimal number (00-61).
- `%z` UTC offset (z is the number of characters in the offset). Offset from Greenwich; -0800 is 8 hours west of UTC.
- `%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
- `boxplot(x)` “box-and-whiskers” plot
- `sunflowerplot(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
matplot(x, y) bivariate plot of the first column of x vs. the first one of y, the second one of x vs. the second one of y, etc.

fourfoldplot(x) visualizes, with quarters of circles, the association between two dichotomous variables for different populations (x must be an array with dim=c(2, 2, k), or a matrix with dim=c(2, 2) if k = 1)
assessplot(x) Cohen–Fridy graph showing the deviations from independence of rows and columns in a two dimensional contingency table
mosaicplot(x) ‘mosaic’ graph of the residuals from a log-linear regression of a contingency table
pairs(x) if x is a matrix or a data frame, draws all possible bivariate plots between the columns of x
plot.ts(x) if x is an object of class "ts", plot of x with respect to time, x may be multivariate but the series must have the same frequency and dates
ts.plot(x) id. but if x is multivariate the series may have different dates and must have the same frequency
qqnorm(x) quantiles of x with respect to the values expected under a normal law
qqplot(x, y) quantiles of y with respect to the quantiles of x
contour(x, y, z) contour plot (data are interpolated to draw the quantiles of x)
image(x, y, z) adds points (the option points(x, y) is used) but with colours (actual data are plotted)
contour.plot (data are interpolated to draw the quantiles of x)
contour.plot(x,y,z) draws a line of slope b and intercept a
abline(h=y) draws a horizontal line at ordinate y
abline(v=x) draws a vertical line at abscissa x
abline(lm.obj) draws the regression line given by lm.obj
rect(x, y, x2, y2) draws a rectangle which left, right, bottom, and top limits are x, y, x2, and y2, respectively
polygon(x, y) draws a polygon linking the points with coordinates given by x and y
legend(x, y, legend) adds the legend at the point (x,y) with the symbols given by legend

Graphical parameters
These can be set globally with par( . . . ); many can be passed as parameters to plotting commands.

Low-level plotting commands
points(x, y) adds points (the option type= can be used)
lines(x, y) id. but with lines
text(x, y, labels, ...) adds text given by labels at coordinates (x,y); a typical use is: plot(x, y, type="n"); text(x, y, names)

Lattice (Trellis) graphics
xyplot(y~x) bivariate plots (with many functionalities)
barchart(y~x) histogram of the values of y with respect to those of x
dotplot(y~x) Cleveland dot plot (stacked plots line-by-line and column-by-column)
densityplot(y~x) density functions plot
densityplot(y~x) density functions plot
histogram(x) histogram of the frequencies of x
bwplot(y~x) “box-and-whiskers” plot
qqmath(x) quantiles of x with respect to the values expected under a theoretical distribution
stripplot(y~x) single dimension plot, x must be numeric, y may be a factor
qq(y~x) quantiles to compare two distributions, x must be numeric, y may be numeric, character, or factor but must have two ‘levels’
splo姆(x) matrix of bivariate plots
parallel(x) parallel coordinates plot
levelplot(z~x+y|gl1+g2) coloured plot of the values of z at the coordinates given by x and y (x, y and z are all of the same length)
wireframe(z~x+y|gl1+g2) 3d surface plot
cloud(z~x+y|gl1+g2) 3d scatter plot
In the normal Lattice formula, $y \times g1^g2$ 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 ?llines). 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) nlm(f,p) minimize function f using a Newton-type algorithm with starting values p

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

**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

**nls(formula)** nonlinear least-squares estimates of the nonlinear model parameters

**approx(x,y=)** linearly interpolate given data points; x can be an xy plotting structure

**spline(x,y=)** cubic spline interpolation

**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:

**predict(fit,...)** predictions from fit based on input data

**df.residual(fit)** returns the number of residual degrees of freedom

**coef(fit)** returns the estimated coefficients (sometimes with their standard-errors)

**residuals(fit)** returns the residuals

**deviance(fit)** returns the deviance

**fitted(fit)** returns the fitted values

**logLik(fit)** computes the log-likelihood of the model and the number of parameters

**AIC(fit)** computes the Akaike information criterion or AIC

**Statistics**

**aov(formula)** analysis of variance model

**anova(fit,...)** analysis of variance (or deviance) tables for one or more fitted model objects

**density(x)** kernel density estimates of x

**binom.test()**, **pairwise.t.test()**, **power.t.test()**, **prop.test()**, **t.test()**, ... use help.search("*test")

**Distributions**

**rnorm(n, mean=0, sd=1)** Gaussian (normal)

**rexp(n, rate=1)** exponential

**rgamma(n, shape, scale=1)** gamma

**rpois(n, lambda)** Poisson

**rweibull(n, shape, scale=1)** Weibull

**rcauchy(n, location=0, scale=1)** Cauchy

**rbeta(n, shape1, shape2)** beta

**rt(n, df) 'Student' (t)**

**rf(n, df1, df2)** Fisher–Snedecor (F) ($\chi^2$)

**rchisq(n, df)** Pearson

**rbinom(n, size, prob)** binomial

**rgeom(n, prob)** geometric

**rhyper(nn, m, n, k)** hypergeometric

**rlogis(n, location=0, scale=1)** logistic

**rnorm(n, meanlog=0, sdlog=1)** lognormal

**rnbinom(n, size, prob)** negative binomial

**runif(n, min=0, max=1)** uniform

**rwilcox(nn, m, n), 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, ...)), the cumulative probability density (qfunc(x, ...)), and the value of quantile (qfunc(p, ...), with 0 < p < 1).

**Programming**

**function( arglist ) expr** function definition return(value)

**if(cond) expr**

**if(cond) cons.expr else alt.expr**

**for(var in seq) expr**

**while(cond) expr**

**repeat expr**

**break**

**next**

Use braces { } around statements

**ifelse(test, yes, no)** a value with the same shape as test filled with elements from either yes or no

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