Day 3, Session 1: Accessing R help

Jessica Williams-Nguyen and Brian D. Williamson

EPI/BIOST Bootcamp 2017

26 September 2017

Accessing help

R can be tricky at times — but one of the most important things to remember (and easiest things to forget) is that if you are having a problem with R programming, someone else has almost surely had the same problem.

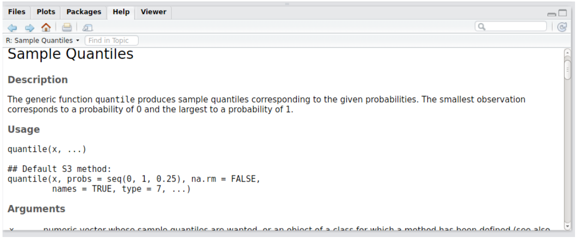
Thankfully, there are a wide variety of resources for getting help with R. These fall into two general categories: built-in help files for functions and packages, written by the package maintainers; and online forums where users can post questions and answers related to programming and/or R.

In addition to being accessed in different ways, parsing the relevant information is somewhat different between these two general categories.

Help files within R

To access the help file for a function or package from within R, type `?` followed by the name of the function you want to learn more about.

For example, if we want to learn about the `quantile()` function, we type `?quantile` in the console. This brings up a help page in the file/plot viewer:



Sample Quantiles

Description

The generic function `quantile` produces sample quantiles corresponding to the given probabilities. The smallest observation corresponds to a probability of 0 and the largest to a probability of 1.

Usage

```
quantile(x, ...)

## Default S3 method:
quantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE,
         names = TRUE, type = 7, ...)
```

Arguments

x numeric vector whose sample quantiles are wanted, or an object of a class for which a method has been defined (see also

Reading R help files

Help files can be somewhat daunting at first, so learning how to parse out the necessary information is valuable.

Standard R help files consist of a few parts, but the most useful are: the description, usage, arguments, details, value, and examples.

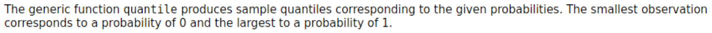
Reading R help files: the description

The first place you should go in an R help file is the description.

This tells you what the function is supposed to do, and often gives some detail about what you should expect the output to be.

The description is most useful if you see a new function and want to know what it does, or download a new package and want to learn about the functions inside it.

Example: `?quantile`



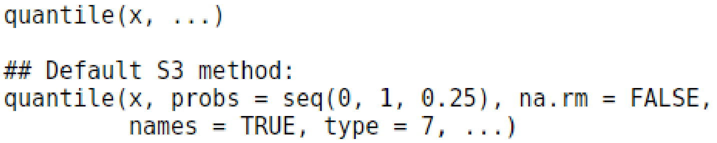
The generic function `quantile` produces sample quantiles corresponding to the given probabilities. The smallest observation corresponds to a probability of 0 and the largest to a probability of 1.

Reading R help files: usage

The usage section starts the meat of the help file. Here we learn how to call the function, and get an introduction to the arguments that it takes (these are described fully in the arguments section).

Often, this includes the bare minimum required by the function (the first line), as well as all of the arguments specific to that function (usually called `Default S3 method`). The default shows you what will happen if you only specify the argument(s) in the first line.

Example: `?quantile`



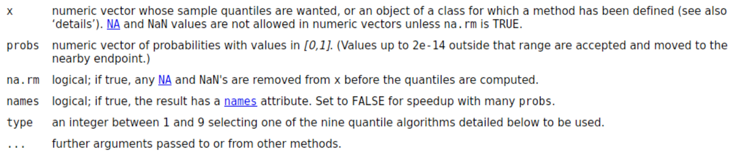
```
quantile(x, ...)

## Default S3 method:
quantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE,
         names = TRUE, type = 7, ...)
```

Reading R help files: arguments

Arguments control behavior of the function. The most basic argument to a function is the data that it operates on. Other arguments control things like *x*- and *y*-axis labels for plots, or which sample quantile to display, among many other options.

The arguments section tells you how to customize the output of the function to your specifications.

Example: `?quantile`



`x`	numeric vector whose sample quantiles are wanted, or an object of a class for which a method has been defined (see also 'details'). NA and NaN values are not allowed in numeric vectors unless `na.rm` is TRUE.
`probs`	numeric vector of probabilities with values in *[0,1]*. (Values up to `2e-14` outside that range are accepted and moved to the nearby endpoint.)
`na.rm`	logical; if true, any NA and NaN's are removed from `x` before the quantiles are computed.
`names`	logical; if true, the result has a `names` attribute. Set to FALSE for speedup with many `probs`.
`type`	an integer between 1 and 9 selecting one of the nine quantile algorithms detailed below to be used.
`...`	further arguments passed to or from other methods.

Special argument: ellipsis

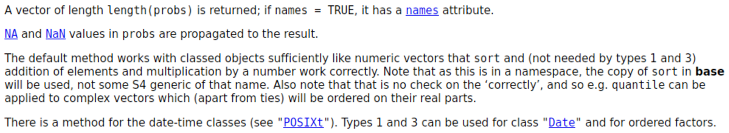
The ellipsis argument `...` allows arguments to be passed to other functions. This allows the programmer to use functions within their own function, but not specify how the user should customize those other functions.

In the `quantile()` function, the ellipsis isn't very helpful. However, in the `hist()` function (which plots a histogram), the ellipsis lets us specify how to set up the plot: we can control things like size of label text relative to the plot and the amount of white space in the margins, among other things.

Reading R help files: details

The details section often tells us what certain arguments lead to when calling the function. This is particularly helpful when you encounter unexpected behavior.

Example: `?quantile`



A vector of length `length(probs)` is returned; if `names = TRUE`, it has a names attribute.

NA and NaN values in `probs` are propagated to the result.

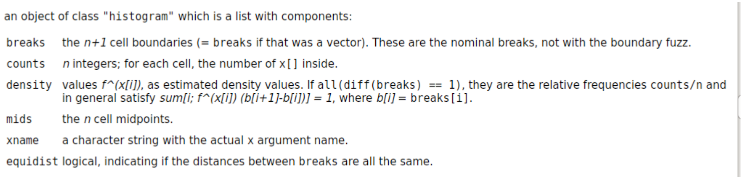
The default method works with classed objects sufficiently like numeric vectors that `sort` and (not needed by types 1 and 3) addition of elements and multiplication by a number work correctly. Note that as this is in a namespace, the copy of `sort` in **base** will be used, not some S4 generic of that name. Also note that that is no check on the ‘correctly’, and so e.g. `quantile` can be applied to complex vectors which (apart from ties) will be ordered on their real parts.

There is a method for the date-time classes (see "POSIXt"). Types 1 and 3 can be used for class "Date" and for ordered factors.

Reading R help files: value

The value section tells us what the function returns. This is especially important when you will be taking results from one function and putting them as an argument into a new function!

Example: `?hist`



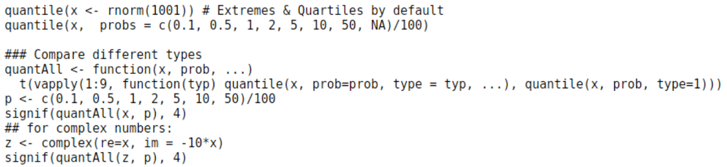
an object of class "histogram" which is a list with components:

`breaks`	the *n+1* cell boundaries (= `breaks` if that was a vector). These are the nominal breaks, not with the boundary fuzz.
`counts`	*n* integers; for each cell, the number of `x[]` inside.
`density`	values *f^(x[i])*, as estimated density values. If `all(diff(breaks) == 1)`, they are the relative frequencies `counts/n` and in general satisfy *sum[i; f^(x[i]) (b[i+1]-b[i])] = 1*, where *b[i]* = `breaks[i]`.
`mids`	the *n* cell midpoints.
`xname`	a character string with the actual `x` argument name.
`equidist`	logical, indicating if the distances between `breaks` are all the same.

Reading R help files: examples

Look here for how to use the function. Often the examples will be simple, but are good enough to get a rough idea of how the function works.

Example: `?quantile`



```
quantile(x <- rnorm(1001)) # Extremes & Quartiles by default
quantile(x,  probs = c(0.1, 0.5, 1, 2, 5, 10, 50, NA)/100)

### Compare different types
quantAll <- function(x, prob, ...)
  t(vapply(1:9, function(typ) quantile(x, prob=prob, type = typ, ...), quantile(x, prob, type=1)))
p <- c(0.1, 0.5, 1, 2, 5, 10, 50)/100
signif(quantAll(x, p), 4)
## for complex numbers:
z <- complex(re=x, im = -10*x)
signif(quantAll(z, p), 4)
```

Reading R help files: on CRAN

All R packages that are hosted on CRAN are required to also have a pdf containing all of the help files for the package.

I find this mostly useful to find out what a new package does; these documents can be easily accessed by searching for "some package CRAN" (replace "some package" with the package name).

Vignettes

Many R packages now come packaged with vignettes — these are essentially extended examples, which walk you through how to use a particular package or function.

Vignettes can be especially useful for complex functions, like the one for dplyr (a useful tool for reshaping data).

If you are new to a particular function, going through the vignette can help you get on your feet quickly!

Useful online resources

Looking up a help file from within R presumes that you know the name of the function, which sometimes is too much to ask.

A wide variety of problems can lead you to online resources. Often, I go looking for things that I should be able to do in R, but I don't know the exact function name. I also typically research error messages that pop up unexpectedly.

The first resource I always use is Google. Here I will type something like "do something in R", which usually brings me to a place like stackoverflow or stackexchange.

These two sites have a wide variety of useful exchanges between people who post questions and answers, and often have exactly the information you need.

Using online resources

Online resources are great in two respects: typically someone has already asked the question that you have, and typically there are a wide variety of answers to solve the problem; and users can upvote good answers, so that you typically only see the “best” answer to a given problem.

However, sometimes there is a difference of opinion on what “best” means in a given context — use your own judgment!

For the most part, you can’t go wrong with answers that have received a decent number of upvotes. But keep on scrolling until you find the answer that best matches your particular question!