R4

Getting help

Covered in R4

- Generic help with R and RStudio
- Specific help with functions
  - Finding the arguments of a function
  - Finding functions starting with a phrase
  - Finding functions including a phrase
  - Finding help for a function

1 Generic help

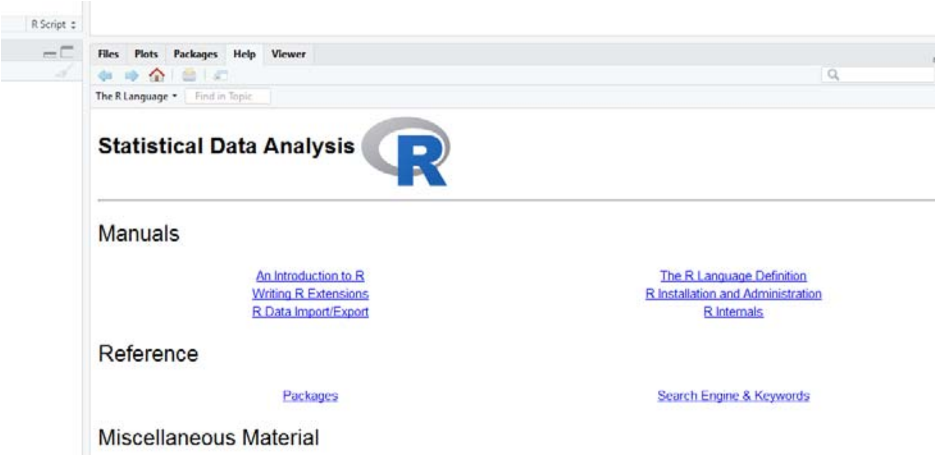
In the last chapter we introduced some basic mathematical functions in R. However because R is a command based programming language – if you don't know how the commands work you're going to have trouble getting R to do what you want.

General help

When you start up R or RStudio it suggests you could type

```
help.start( )
```

In RStudio, this will display, in the bottom-right window, general help html page which links to all the manuals, reference documents and other miscellaneous material:



Statistical Data Analysis

Manuals

An Introduction to R
Writing R Extensions
R Data Import/Export
The R Language Definition
R Installation and Administration
R Internals

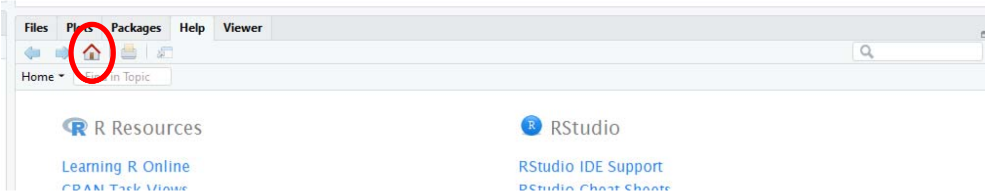
Reference

Packages
Search Engine & Keywords

Miscellaneous Material

The most useful general item is "An Introduction to R" which is the starting R manual. However, the manuals and FAQ documents are clearly intended for experts and so aren't going to help you much until you get much more proficient.

RStudio has its own help page which you can access by clicking the home button in the Help window:



R Resources
Learning R Online
CRAN Task Views

RStudio
RStudio IDE Support
RStudio Cheat Sheets

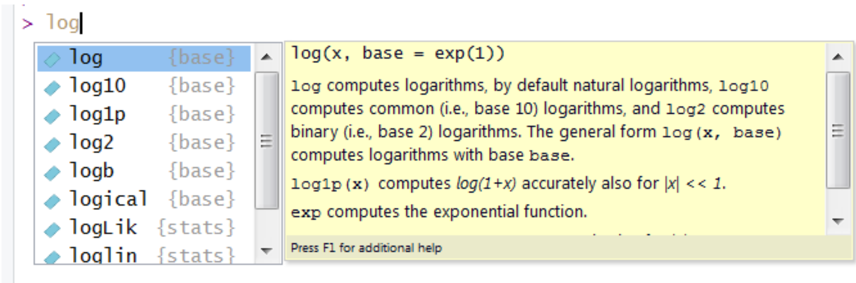
This links to lots of useful information and you might wish to try a few of the links yourself.

In particular, the first link, *Learning R Online*, recommends lots of free resources that you could use, instead of this one if you wish, to get to grips with R.

2 More specific help

Finding the arguments of a known function

Supposing I was working with the log function and had forgotten its arguments. Working in RStudio, we can just type in the function in the Console and it will display some useful information, including the possible arguments and options on the first line of the yellow box:



```
> log
```

`log(x, base = exp(1))`

`log` computes logarithms, by default natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `log(x, base)` computes logarithms with base `base`.

`log1p(x)` computes *log(1+x)* accurately also for |*x*| << *1*.

`exp` computes the exponential function.

(Alternatively use the function args, *eg* try typing args(log) in the Console.)

The arguments listed are:

- the input value, *x* (*ie* the value we want to calculate the log of)
- the option for the base. Note it says base = exp(1). That tells us that the default base is e (*ie* what the base is if we miss out this argument). So just writing log(10) will gives us ln(10).

Getting help on a known function

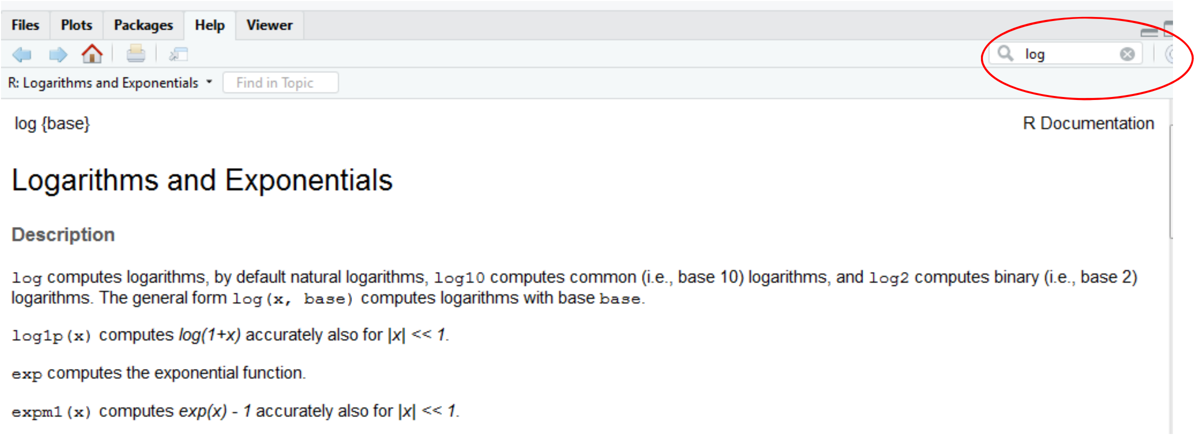
If you want help on a particular function, you can use the *help* function, or just use a question mark followed by the function name (without any brackets). For example:

help(log)

or

?log

Either will open up the relevant help page in the help window of RStudio:



log {base} R Documentation

Logarithms and Exponentials

Description

`log` computes logarithms, by default natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `log(x, base)` computes logarithms with base `base`.

`log1p(x)` computes *log(1+x)* accurately also for |*x*| << *1*.

`exp` computes the exponential function.

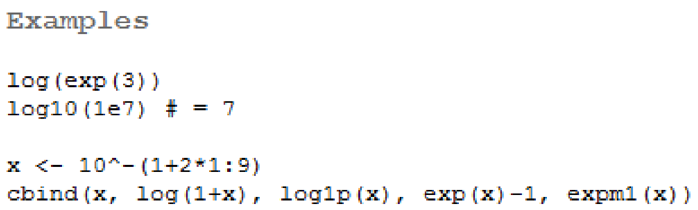
`expm1(x)` computes *exp(x) - 1* accurately also for |*x*| << *1*.

The help page gives us lots of information, probably beyond what we were looking for, about the function.

Another way of reaching the same page is by using the search box in the help window – circled in red above.

Getting examples for known functions

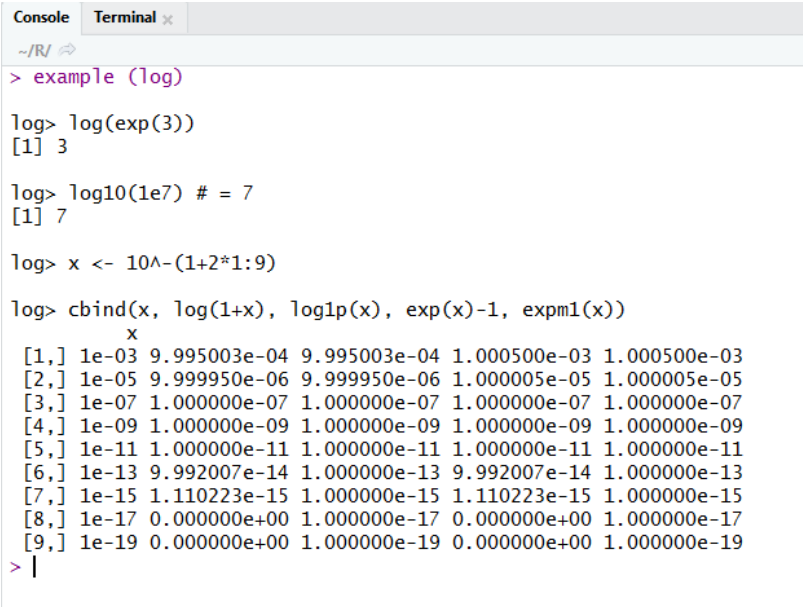
At the bottom of the help page there are some examples:



```
Examples

log(exp(3))
log10(1e7) # = 7

x <- 10^-(1+2*1:9)
cbind(x, log(1+x), log1p(x), exp(x)-1, expm1(x))
```

Rather than reading about them you can run them in R, either by copying and pasting or by using the example function:

example(<function name>)



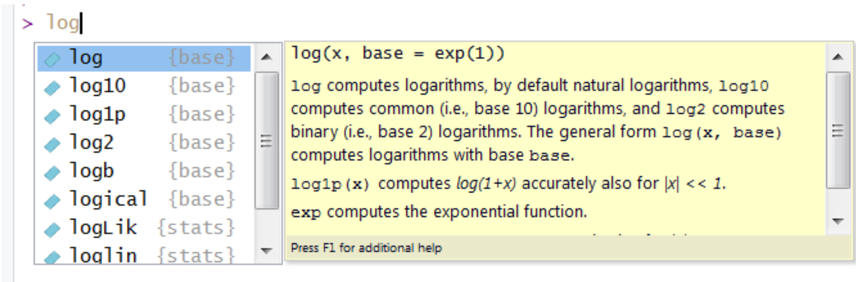
```
> example (log)

log> log(exp(3))
[1] 3

log> log10(1e7) # = 7
[1] 7

log> x <- 10^-(1+2*1:9)

log> cbind(x, log(1+x), log1p(x), exp(x)-1, expm1(x))
          x
 [1,] 1e-03 9.995003e-04 9.995003e-04 1.000500e-03 1.000500e-03
 [2,] 1e-05 9.999950e-06 9.999950e-06 1.000005e-05 1.000005e-05
 [3,] 1e-07 1.000000e-07 1.000000e-07 1.000000e-07 1.000000e-07
 [4,] 1e-09 1.000000e-09 1.000000e-09 1.000000e-09 1.000000e-09
 [5,] 1e-11 1.000000e-11 1.000000e-11 1.000000e-11 1.000000e-11
 [6,] 1e-13 9.992007e-14 1.000000e-13 9.992007e-14 1.000000e-13
 [7,] 1e-15 1.110223e-15 1.000000e-15 1.110223e-15 1.000000e-15
 [8,] 1e-17 0.000000e+00 1.000000e-17 0.000000e+00 1.000000e-17
 [9,] 1e-19 0.000000e+00 1.000000e-19 0.000000e+00 1.000000e-19
> |
```

However, some of these examples can often be a bit obscure. So if in doubt you may just end up searching the internet (*eg* looking on YouTube) to see if someone can explain it more simply.

Finding an unknown function that starts with a given phrase

One of the main issues you'll face in R is finding the correct command for what you want R to do. That's sometimes a little bit trickier to solve.

If you know the function's name starts with a given phrase, say log, you can type that into the Console to reveal the list of functions, as mentioned previously:



```
> log
log       {base}
log10     {base}
log1p     {base}
log2      {base}
logb      {base}
logical   {base}
logLik    {stats}
loglin    {stats}
```

log(x, base = exp(1))

log computes logarithms, by default natural logarithms, log10 computes common (i.e., base 10) logarithms, and log2 computes binary (i.e., base 2) logarithms. The general form log(x, base) computes logarithms with base base.

log1p(x) computes *log(1+x)* accurately also for *|x| << 1*.

exp computes the exponential function.

Press F1 for additional help

This will list all the commands that start with log (with a brief explanation of each). In this case there are 8. You can scroll through the list and hit enter when you find the one you want.

If you're still not sure which of these it is then you could look up each of them using the help function as we did above.

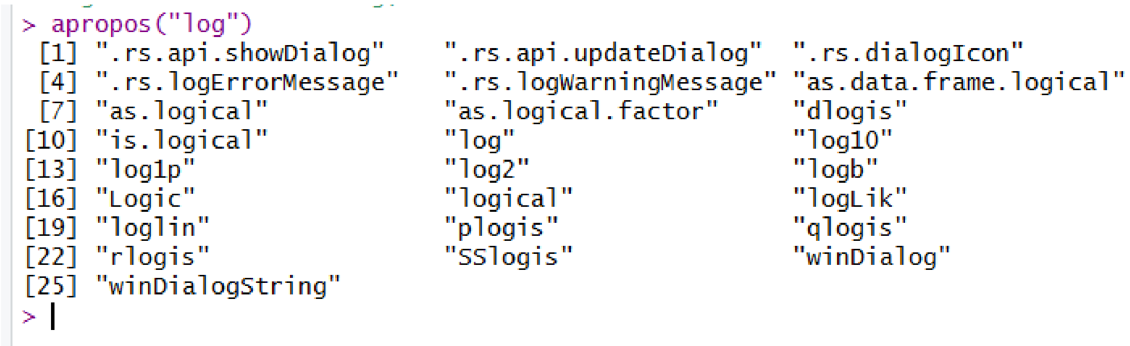
Finding an unknown function that contains a given phrase

Suppose you don't know what the function starts with but you do know that it contains a particular phrase, say "log". In which case you use the **apropos function**.

apropos("<phrase>")

This comes from the French *á propos* which means literally "to purpose", *ie* with reference or with regard to this purpose.

The only frustration is you have to put the letters that you're searching for in speech marks:



```
> apropos("log")
 [1] ".rs.api.showDialog"     ".rs.api.updateDialog"   ".rs.dialogIcon"
 [4] ".rs.logErrorMessage"    ".rs.logWarningMessage" "as.data.frame.logical"
 [7] "as.logical"             "as.logical.factor"      "dlogis"
[10] "is.logical"             "log"                    "log10"
[13] "log1p"                  "log2"                   "logb"
[16] "Logic"                  "logical"                "logLik"
[19] "loglin"                 "plogis"                 "qlogis"
[22] "rlogis"                 "SSlogis"                "winDialog"
[25] "winDialogString"
> 
```

You'll see we get all the 8 functions starting with log that we saw earlier and a further 15 functions. (You may see even more depending on versions/packages installed.)

Finding an unknown function on a particular topic

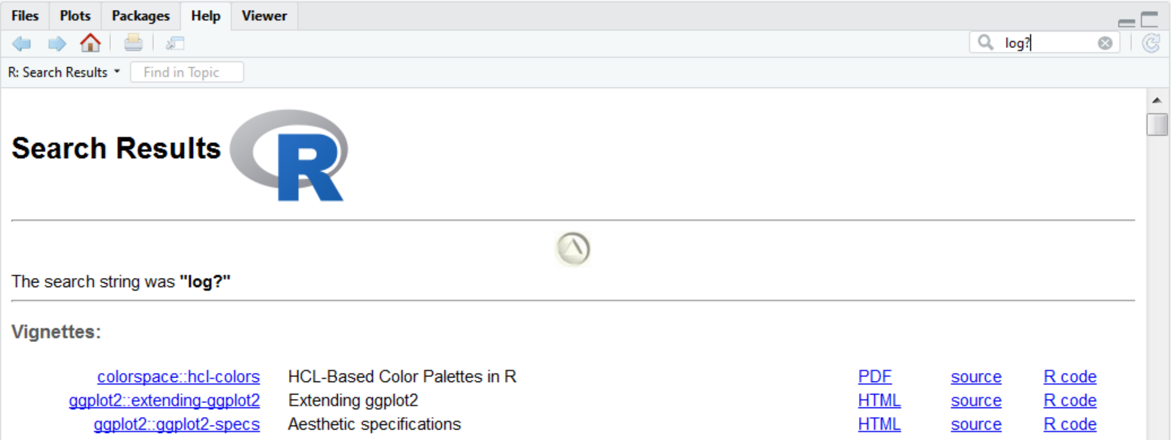
If none of the approaches above help you find the function you want then you'll have to do a more general search using:

help.search("<phrase>") (you need to put the name in speech marks)

or

??<phrase> (no need for speech marks)

or type the phrase followed by a question mark in the search box of the help window, which displays a slightly different set of results:



Search Results

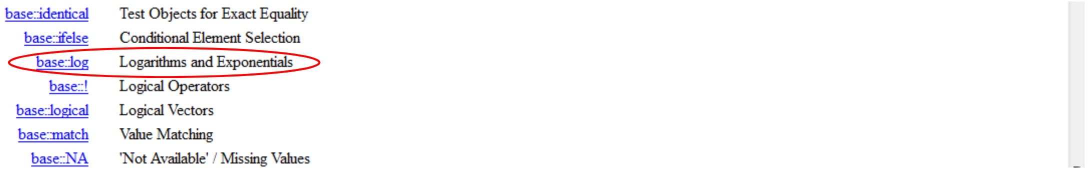
The search string was **"log?"**

Vignettes:

colorspace::hcl-colors	HCL-Based Color Palettes in R	PDF	source	R code
ggplot2::extending-ggplot2	Extending ggplot2	HTML	source	R code
ggplot2::ggplot2-specs	Aesthetic specifications	HTML	source	R code

However, you may just end up searching the internet for "How do I … in R?".

The beginning of each entry in the search results gives the name of the package where that function can be found:



base::identical	Test Objects for Exact Equality
base::ifelse	Conditional Element Selection
base::log	Logarithms and Exponentials
base::!	Logical Operators
base::logical	Logical Vectors
base::match	Value Matching
base::NA	'Not Available' / Missing Values

You can see above that the function *log* is in the base package – *ie* the standard package that is included in R. We'll talk more about packages later.

Clicking on the link takes us to the help(log) page we saw earlier.

Finally, you may wish to explore the other features that can be found in the Help menu of RStudio, including a list of keyboard shortcuts, which might help you save time as you become more proficient in R. For example, Ctrl 2 takes you to the Console.

3 Summary

Key commands

help.start( )	Takes you to the general html help page
args(<function>)	Lists the arguments of <function>
?<function>	Takes you to the html help page for <function>
example(<function>)	Runs the examples for <function> given at the bottom of its html help page
apropos(“abc”)	Lists the functions that contain the phrase abc in their name
??abc	Lists all functions that have something to do with abc

4 Have a go

You will only get proficient at R by practising.

1. Without referring to this chapter, use R to:

   - find the arguments of the choose function
   - find the help page for the choose function
   - run the examples for the choose function
   - find a list of all functions that start with "choose"
   - find a list of all functions which contain the word "choose"
   - find a list of all functions that have something to do with "choose".