## Where to get help

### Use the built-in RStudio help interface to search for more information on R functions

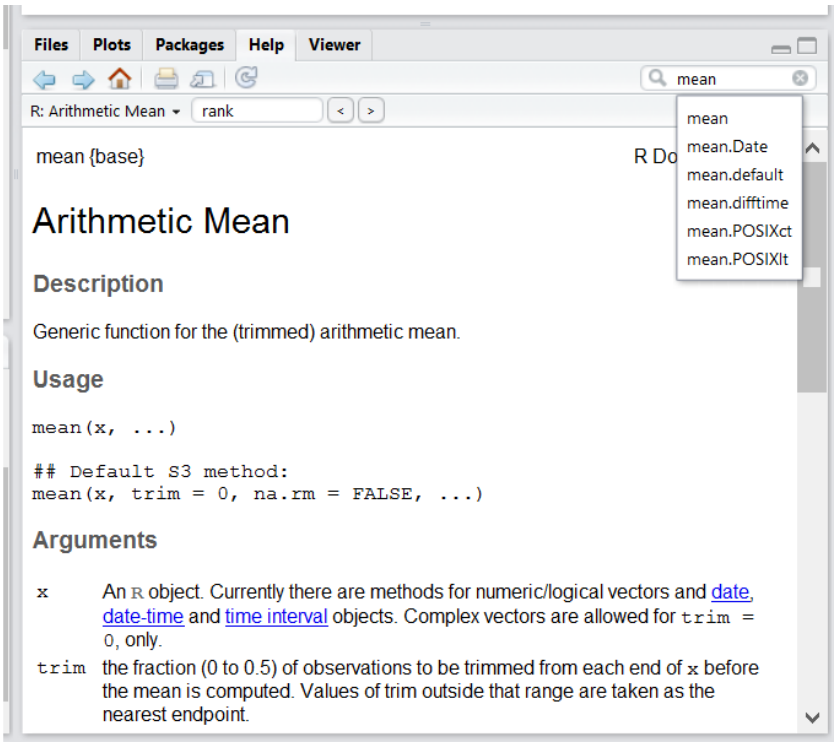


Figure 1: RStudio help interface.

One of the fastest ways to get help, is to use the RStudio help interface. This panel by default can be found at the lower right hand panel of RStudio. As seen in the screenshot, by typing the word "Mean", RStudio tries to also give a number of suggestions that you might be interested in. The description is then shown in the display window.

### I know the name of the function I want to use, but I'm not sure how to use it

If you need help with a specific function, let's say `barplot()`, you can type:

```
?barplot
```

If you just need to remind yourself of the names of the arguments, you can use:

```
args(lm)
```

### I want to use a function that does X, there must be a function for it but I don't know which one...

If you are looking for a function to do a particular task, you can use the `help.search()` function, which is called by the double question mark `??`. However, this only looks through the installed packages for help pages with a match to your search request

```
??kruskal
```

If you can't find what you are looking for, you can use the rdocumentation.org website that searches through the help files across all packages available.

Finally, a generic Google or internet search "R <task>" will often either send you to the appropriate package documentation or a helpful forum where someone else has already asked your question.

### I am stuck... I get an error message that I don't understand

Start by googling the error message. However, this doesn't always work very well because often, package developers rely on the error catching provided by R. You end up with general error messages that might not be very helpful to diagnose a problem (e.g. "subscript out of bounds"). If the message is very generic, you might also include the name of the function or package you're using in your query.

However, you should check Stack Overflow. Search using the `[r]` tag. Most questions have already been answered, but the challenge is to use the right words in the search to find the answers:

http://stackoverflow.com/questions/tagged/r

The Introduction to R can also be dense for people with little programming experience but it is a good place to understand the underpinnings of the R language.

The R FAQ is dense and technical but it is full of useful information.

### Asking for help

The key to receiving help from someone is for them to rapidly grasp your problem. You should make it as easy as possible to pinpoint where the issue might be.

Try to use the correct words to describe your problem. For instance, a package is not the same thing as a library. Most people will understand what you meant, but others have really strong feelings about the difference in meaning. The key point is that it can make things confusing for people trying to help you. Be as precise as possible when describing your problem.

If possible, try to reduce what doesn't work to a simple *reproducible example*. If you can reproduce the problem using a very small data frame instead of your 50000 rows and 10000 columns one, provide the small one with the description of your problem. When appropriate, try to generalise what you are doing so even people who are not in your field can understand the question. For instance instead of using a subset of your real dataset, create a small (3 columns, 5 rows) generic one. For more information on how to write a reproducible example see this article by Hadley Wickham.

To share an object with someone else, if it's relatively small, you can use the function `dput()`. It will output R code that can be used to recreate the exact same object as the one in memory:

```
## iris is an example data frame that comes with R and head() is a
## function that returns the first part of the data frame
dput(head(iris))
```

```
structure(list(Sepal.Length = c(5.1, 4.9, 4.7, 4.6, 5, 5.4),
    Sepal.Width = c(3.5, 3, 3.2, 3.1, 3.6, 3.9), Petal.Length = c(1.4,
    1.4, 1.3, 1.5, 1.4, 1.7), Petal.Width = c(0.2, 0.2, 0.2,
    0.2, 0.2, 0.4), Species = structure(c(1L, 1L, 1L, 1L, 1L,
    1L), levels = c("setosa", "versicolor", "virginica"), class = "factor")), row.names = c(1
6L), class = "data.frame")
```

If the object is larger, provide either the raw file (i.e., your CSV file) with your script up to the point of the error (and after removing everything that is not relevant to your issue). Alternatively, in particular if your question is not related to a data frame, you can save any R object to a file[^export]:

```
saveRDS(iris, file="/tmp/iris.rds")
```

The content of this file is however not human readable and cannot be posted directly on Stack Overflow. Instead, it can be sent to someone by email who can read it with the `readRDS()` command (here it is assumed that the downloaded file is in a `Downloads` folder in the user's home directory):

```
some_data <- readRDS(file="~/Downloads/iris.rds")
```

Last, but certainly not least, **always include the output of `sessionInfo()`** as it provides critical information about your platform, the versions of R and the packages that you are using, and other information that can be very helpful to understand your problem.

```
sessionInfo()
```

```
R version 4.3.1 (2023-06-16 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)

Matrix products: default


locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

loaded via a namespace (and not attached):
 [1] compiler_4.3.1    fastmap_1.1.1     cli_3.6.1         tools_4.3.1
 [5] htmltools_0.5.5   rstudioapi_0.15.0 rmarkdown_2.23    knitr_1.43
 [9] jsonlite_1.8.7    xfun_0.39         digest_0.6.33     rlang_1.1.1
[13] evaluate_0.21
```

**Where to ask for help?**

- The person sitting next to you. Don't hesitate to talk to your neighbour during the workshop, compare your answers, and ask for help.
- The instructors. We're here to help you.
- Your friendly colleagues: if you know someone with more experience than you, they might be able and willing to help you.
- Stack Overflow: if your question hasn't been answered before and is well crafted, chances are you will get an answer in less than 5 min. Remember to follow their guidelines on how to ask a good question.
- The R-help mailing list: it is read by a lot of people (including most of the R core team), a lot of people post to it, but the tone can be pretty dry, and it is not always very welcoming to new users. If your question is valid, you are likely to get an answer very fast but don't expect that it will come with smiley faces. Also, here more than anywhere else, be sure to use correct vocabulary (otherwise you might get an answer pointing to

the misuse of your words rather than answering your question). You will also have more success if your question is about a base function rather than a specific package.
- If your question is about a specific package, see if there is a mailing list for it. Usually it's included in the DESCRIPTION file of the package that can be accessed using `packageDescription("name-of-package")`. You may also want to try to email the author of the package directly, or open an issue on the code repository (e.g., GitHub).
- There are also some topic-specific mailing lists (GIS, phylogenetics, etc…), the complete list is here.

### More resources

- The Posting Guide for the R mailing lists.
- How to ask for R help useful guidelines.
- This blog post by Jon Skeet has quite comprehensive advice on how to ask programming questions.
- The reprex package is very helpful to create reproducible examples when asking for help. The rOpenSci community call "How to ask questions so they get answered" (Github link and video recording) includes a presentation of the reprex package and of its philosophy.

---