Getting Help for R

Writing R code can be challenging as well as irritating and frustrating. When you write R code, at some point you will feel like chucking your computer out a very high window. Even highly capable R users will frequently hit roadblocks and the most-efficient code writers often have become so by having a greater understanding of where to find help. This document is aimed to provide help in finding the solutions you need as you write your R code.

While I am more than happy to help folks with their code issues, I strongly recommend that you have read and utilized the first two sections of this document before dialing my phone number. I also recommend that you have prepared to ask help for help as outlined in the final section. I'd hate for you to have to call me back.

Helping Yourself

Folks on the internet can be a testy bunch. You will find your life much more pleasant if you attempt to find the answers on your own before asking others for help. While a great many places exist to go seek R assistance from others online, if you end up asking a simple or common question that can be easily found on your own you may be chastised quite severely. So, where does one look for answers to R problems or difficulties?

R Help Documentation:

Base R has an extensive help system with consistently formatted documentation. In addition, contributed packages on the cran repository must also have their own help pages, structured exactly the same way, that are loaded into the help system anytime you load the package into R. As you get used to the format used by the documentation, you will come to find many of the answers you may need. The most useful parts of the documentation page are often (examples are from the FSA package, wrAdd function):

1) The USAGE section which displays the functions syntax
    a. Arguments with just a name are required (i.e. "x")
    b. Arguments with an "=" sign are optional. The value listed after the sign indicate the default values which will be used if no alternative is given (i.e. "conf.level=0.95)
    c. Arguments with a vector show acceptable values (i.e. "alternative = c("two.sided", "less", "greater")

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

2) The VALUE section which displays a description for each argument

Value

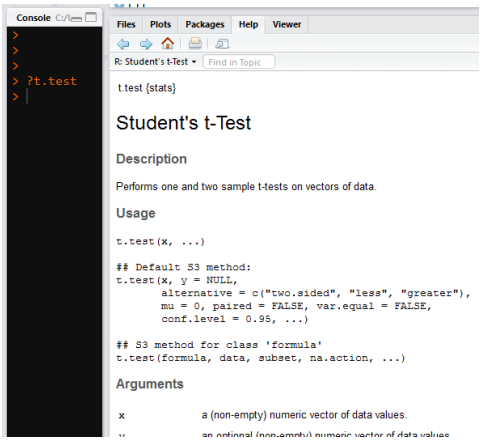
A list with class `"htest"` containing the following components:

`statistic`	the value of the t-statistic.
`parameter`	the degrees of freedom for the t-statistic.
`p.value`	the p-value for the test.
`conf.int`	a confidence interval for the mean appropriate to the specified alternative hypothesis.
`estimate`	the estimated mean or difference in means depending on whether it was a one-sample test or a two-sample test.
`null.value`	the specified hypothesized value of the mean or mean difference depending on whether it was a one-sample test or a two-sample test.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating what type of t-test was performed.
`data.name`	a character string giving the name(s) of the data.

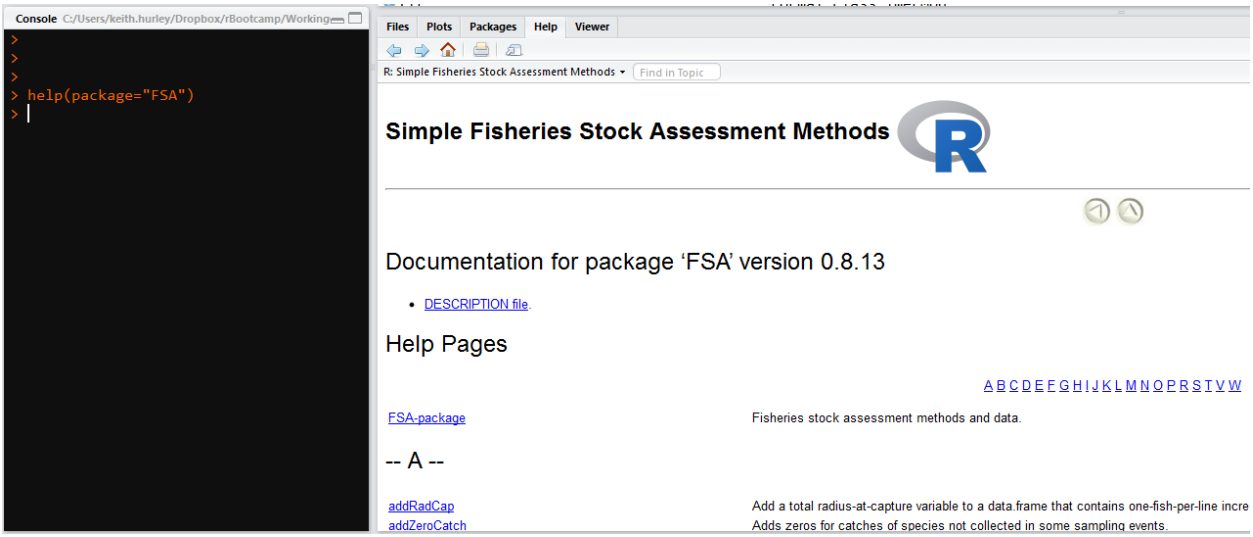
3) The EXAMPLES section which displays some actually working examples, these examples can be run from the console by typing example("function_name")

Searching R Help Documentation

There are a number of ways to search and access this documentation. RStudio has a help pane that allows a user to search or click through the loaded help documentation. However, most folks find it easier to use either the help function or the help operator (?). By typing the following code into either the console or a code window (and running the line), the help system will search for a help page for the specified function.



The help() function may also be used to access information about a package. This function will take you to an index for the help documentation for the package.



One major problem with thse steps, however, is that one must know how to spell the function or package name. The apropos() function searches the help documentation for objects and functions that are accessible in the current R session and that have names that include the specified string. This may be a character string or a *regular expression* to be used for pattern-matching. Matching by apropos() is case-insensitive. The results returned by apropos() can then be fed into the help operator (?) to get more information. For example, let's look at a search using a character string to find help about glm:

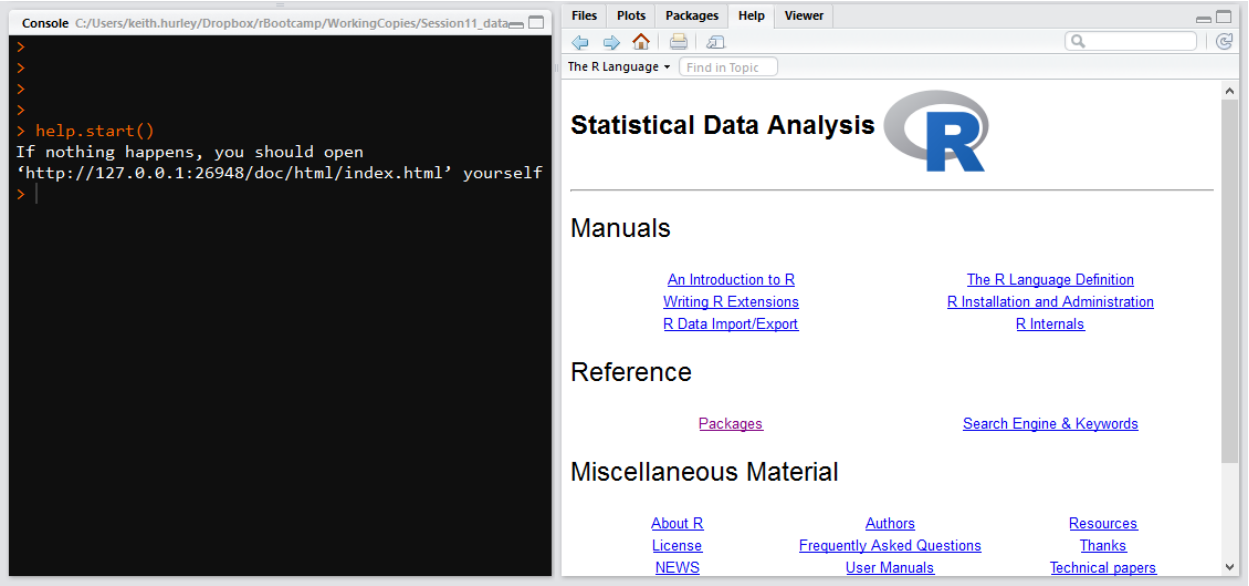
```
> apropos("glm")
 [1] ".__C__anova.glm"      ".__C__anova.glm.null" ".__C__glm"          ".__C__glm.null"       ".__C__glmerMod"
 [6] "glm"                  "glm.control"          "glm.convert"        "glm.fit"              "glm.nb"
[11] "glmer"                "glmer.nb"             "glmerControl"       "glmerLaplaceHandle"   "glmFamily"
[16] "glmmPQL"              "glmResp"              "isGLMM"             "loglm"                "loglm1"
[21] "mkGlmerDevfun"        "optimizeGlmer"        "predict.glm"        "residuals.glm"        "summary.glm"
[26] "updateGlmerDevfun"
```

And now let's use a regular expression to find help about functions and objects that START with glm (that's what the ^ means):

```
> apropos("^glm")
 [1] "glm"               "glm.control"       "glm.convert"       "glm.fit"           "glm.nb"
 [6] "glmer"             "glmer.nb"          "glmerControl"      "glmerLaplaceHandle" "glmFamily"
[11] "glmmPQL"           "glmResp"
```

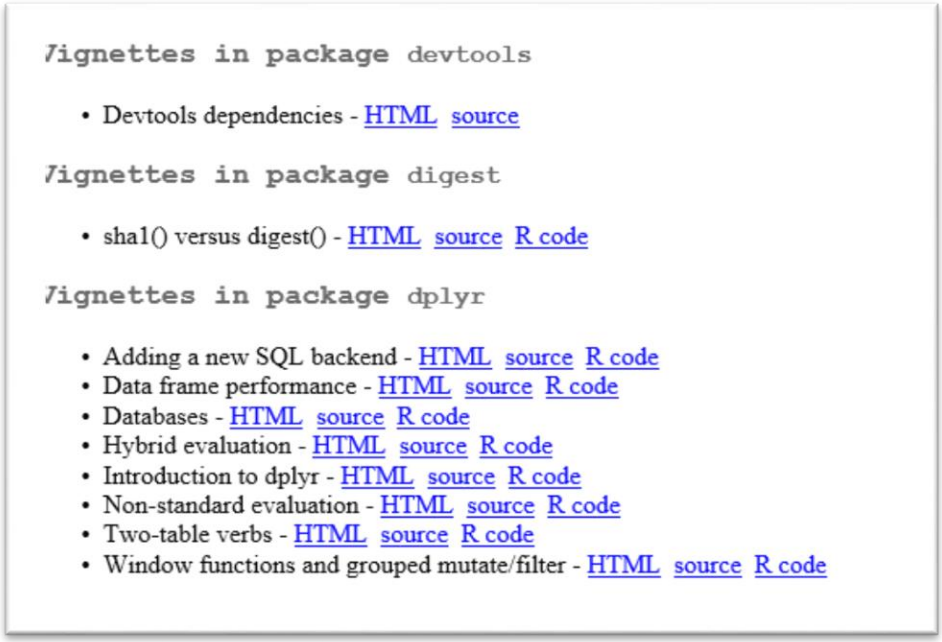
Sometimes these methods don't provide any results and the search must be expanded from just the names of functions and packages. The `help.search()` function scans the entire documentation pages for all packages installed in your library. The input argument for the search functions just like the apropos() function and accepts both character strings and regular expressions. The `??` operator is a shortcut for `help.search()` just like the `?` operator is a shortcut for help().

If you prefer a point and click approach to finding help, `help.start()` will open a root index help page and allow you to click links to navigate through the installed help documentation.



Vignettes and Code Demonstrations:

The help documentation for many packages include vignettes, which are short tutorials about using the features of the package. The main help page for each package will list the available vignettes or they can be viewed with the `browseVignettes()` function which will open a webpage of all available vignettes on your computer.



Vignettes in package devtools

- Devtools dependencies - HTML source

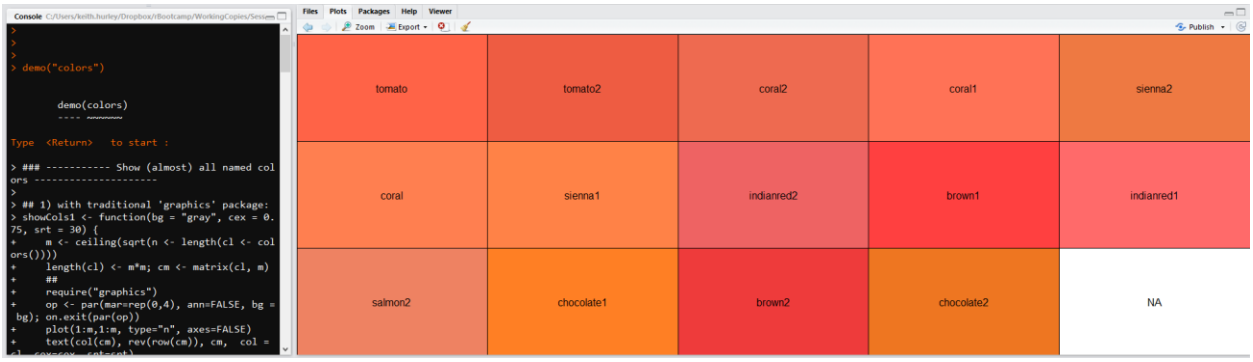
Vignettes in package digest

- sha1() versus digest() - HTML source R code

Vignettes in package dplyr

- Adding a new SQL backend - HTML source R code
- Data frame performance - HTML source R code
- Databases - HTML source R code
- Hybrid evaluation - HTML source R code
- Introduction to dplyr - HTML source R code
- Non-standard evaluation - HTML source R code
- Two-table verbs - HTML source R code
- Window functions and grouped mutate/filter - HTML source R code

In much the same way, the demos() function will display a list of all package demonstrations loaded on your computer. Running the demos() function with a demo name will load and run the demo in R.



R Help on the Internet

CRAN Homepage

The homepage for R can be found out http://cran.r-project.org. This page is the gateway into R including all the contributed packages in the CRAN archive. If you wish to view the package documentation for a given package, it can be found under “Packages” and then by clicking on “Table of available packages”. Clicking on the “Reference Manual” link will open a pdf of the packages help documentation.

dplyr: A Grammar of Data Manipulation

A fast, consistent tool for working with data frame like objects, both in memory and out of memory.

Version:	0.7.0
Depends:	R (≥ 3.1.2)
Imports:	assertthat, bindrcpp, glue, magrittr, methods, pkgconfig, rlang (≥ 0.1), R6, Rcpp (≥ 0.12.6), tibble (
LinkingTo:	Rcpp (≥ 0.12.0), BH (≥ 1.58.0-1), bindrcpp, plogr
Suggests:	bit64, covr, dbplyr, dtplyr, DBI, ggplot2, hms, knitr, Lahman (≥ 3.0-1), mgcv, microbenchmark, ny
Published:	2017-06-09
Author:	Hadley Wickham [aut, cre], Romain Francois [aut], Lionel Henry [aut], Kirill Müller [aut], RStudi
Maintainer:	Hadley Wickham <hadley at rstudio.com>
BugReports:	https://github.com/tidyverse/dplyr/issues
License:	MIT + file LICENSE
URL:	http://dplyr.tidyverse.org, https://github.com/tidyverse/dplyr
NeedsCompilation:	yes
Materials:	README NEWS
CRAN checks:	dplyr results

Downloads:

Reference manual:	dplyr.pdf
Vignettes:	dplyr compatibility Introduction to dplyr Programming with dplyr Two-table verbs Window functions
Package source:	dplyr_0.7.0.tar.gz
Windows binaries:	r-devel: dplyr_0.7.0.zip, r-release: dplyr_0.7.0.zip, r-oldrel: dplyr_0.7.0.zip
OS X El Capitan binaries:	r-release: dplyr_0.5.0.tgz
OS X Mavericks binaries:	r-oldrel: dplyr_0.7.0.tgz
Old sources:	dplyr archive

CRAN Task Views

Task Views are also available on the CRAN website and summarize R resources on CRAN grouped into particular areas of application. These task views can help sort through the thousands of CRAN packages if you're not familiar with the packages for given tasks.

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & V
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing

CRAN Task View: Cluster Analysis & Finite Mixture Models

Maintainer: Friedrich Leisch and Bettina Gruen
Contact: Bettina.Gruen at jku.at
Version: 2017-05-03
URL: https://CRAN.R-project.org/view=Cluster

This CRAN Task View contains a list of packages that can be used for finding groups in data and modeling unobserved cross-sectional heterogeneity. Many packages provide functionality for more than one of the topics listed below, the section headings are mainly meant as quick starting points rather than an ultimate categorization. Except for packages stats and cluster (which ship with base R and hence are part of every R installation), each package is listed only once.

Most of the packages listed in this CRAN Task View, but not all are distributed under the GPL. Please have a look at the DESCRIPTION file of each package to check under which license it is distributed.

Hierarchical Clustering:

- Functions `hclust()` from package stats and `agnes()` from cluster are the primary functions for agglomerative hierarchical clustering, function `diana()` can be used for divisive hierarchical clustering. Faster alternatives to `hclust()` are provided by the packages fastcluster and flashClust.
- Function `dendrogram()` from stats and associated methods can be used for improved visualization for cluster dendrograms.
- The dendextend package provides functions for easy visualization (coloring labels and branches, etc.), manipulation (rotating, pruning, etc.) and comparison of dendrograms (tangelgrams with heuristics for optimal branch rotations, and tree correlation measures with bootstrap and permutation tests for significance).
- Package dynamicTreeCut contains methods for detection of clusters in hierarchical clustering dendrograms.
- Package genie implements a fast hierarchical clustering algorithm with a linkage criterion which is a variant of the single linkage method combining it with the Gini inequality measure to robustify the linkage method while retaining computational efficiency to allow for the use of larger data sets.
- hybridHclust implements hybrid hierarchical clustering via mutual clusters.
- Package idendr0 allows to interactively explore hierarchical clustering dendrograms and the clustered data. The data can be visualized (and interacted with) in a built-in heat map, but also in GGobi dynamic interactive graphics (provided by rggobi), or base R plots.
- Package isopam uses an algorithm which is based on the classification of ordination scores from isometric feature mapping. The classification is performed either as a hierarchical, divisive method or as non-hierarchical partitioning.
- The package protoclust implements a form of hierarchical clustering that associates a prototypical element with each interior node of the dendrogram. Using the package's `plot()` function, one can produce dendrograms that are prototype-labeled and are therefore easier to interpret.
- pvclust is a package for assessing the uncertainty in hierarchical cluster analysis. It provides approximately unbiased p-values as well as bootstrap p-values

Searching for Help Online

For most R users, google is a way of life. In fact, there are times I wonder of most R-users could function without access to google. R has a tremendous user base which contributes to a wealth of online information. There are few problems or difficulties that cannot be resolved with some time spent searching online. Of course, the trick to doing so efficiently is twofold. First is deciding what should be used for search terms. And second is understanding which of the returned sites are the most likely to point you to a solution.

Search Terms

Obviously, including R in your search terms for google is a no-brainer. It is also a no-brainer to realize that almost every webpage includes an "r" in it somewhere. If you are signed into a google account, as you browse the AI will start to understand that when you use "R" you likely mean the r programming language and the results will improve. What other search terms are helpful?

If you are looking for a particular package or finding a package to accomplish a task, include "CRAN". Anything about setting up your environment should include "RStudio". Including the general activity, function calls, and package names like "filter", "linear regression", or ""ggplot" in the terms will help lead to better answers. If you are looking for the way the code should look, use "syntax". If you are receiving error messages, include the general text and error numbers of the message but leave out any of your own variable names, object names, and line numbers that appear in the error message. In general, the more terms you use the more likely your results will be relevant. Some examples:

Bad	Good
how do I change the look of my R chart	R ggplot change axis label font
how do I write a linear model in R	R linear model lm equation syntax
in R, how do I move data around	R data wrangle dplyr switch rows to columns
I can't get rbind to work	Error rbind numbers of columns of argument do not match

Favorable websites

Even if your provide the world's best terms for the question you are asking, you'll still need to choose which results to open. The following websites are commonly returned as results in google R searches and are good choices to pick when reviewing your search results. Some combination of these sites were present in the results of the previous four example queries.

- stackoverflow.com
- stackexchange.com
- tidyverse.org
- rstudio.com
- r-bloggers.com
- cookbook-r.com
- www.sthda.com
- stat.ethz.ch
- www.statmethods.net
- rpubs.com

Stack overflow is a particularly good source of information. The stack overflow site is essentially a web forum created for computer programmers. The posts are grouped by language (R, C#, SAS, etc) and the purpose of the forum is to allow users to ask real-world programming questions and let other programmers help explain solutions. The valuable part of the forum is that other users are able to give approval (upvotes) for answers that they agree with. In addition, the original poster can mark one response as the correct answer that worked for them and this reply is then displayed immediately below the question post with a check mark. What makes the whole site work so well is that it is heavily policed by users according to a strict set of rules like "no duplicate posts containing an already answered question". These rules help keep the posts on topic and reduce the garbage present in most forums. The site also uses a social-media style reward system to encourage users to participate. An excellent tour and explanation of how stack overflow works can be found at https://stackoverflow.com/tour. When stack overflow pages appear in your search results they are often a great place to find answers.

Asking for Help

If you are having trouble answering a question or solving a problem by yourself, then it is time to turn to others. Be aware, people do not like to have their time wasted and when they are volunteering their time to help others, they can get a little harsh. Whether you are asking your local R guru down the hall or posting for help from others on the internet, there are both some hints to increase your odds and cautions to use in your approach.

To improve the odds of getting help:

- Phrase the question clearly – if people don't understand the question, they can't give a good answer.
- Provide code – when you are solving code problems, you need to see the code that isn't working.
- Provide a reproducible example - If you're trying to solve a problem, include a small, self-contained, reproducible example of the problem that others can execute. Two sources for advice on creating reproducible example can be found at https://stackoverflow.com/help/mcve and http://stackoverflow.com/questions/5963269/how-to-make-a-great-r-reproducible-example.
- Do NOT include a bunch of extraneous information that doesn't contribute to understanding the problem, the internet really doesn't care about the who or why, just about the what and the how.
- Be patient - these folks are volunteers and like to feel appreciated. They also have day jobs and other lives. Be thankful for what they take time to offer
- NEVER speak ill of posters – As soon as anything displaying an ill-temper or snarkyness is posted by a user, others will no longer seem interested in helping them
- Do NOT sound like a student – users volunteering their time have no interest in helping others with their homework. Make sure that your initial post does not even have a hint of being a class assignment.

Caution: There Be Dragons Here

When seeking help online, be away that what is contained in the code and data that you are posting, even though seemingly insignificant, can be used for nefarious purposes. NEVER include any username or password information that will be displayed or stored on a public website. NEVER provide any information that would allow other users to access your operational or production data. NEVER provide any data that has personal information of yourself or others included. In fact, if possible many users even change the names of their variables and tables so as not to provide information that may help compromise databases. Follow the advice given in the links above to create isolated, independent, reproducible examples of your problem and then use the answers given to make changes to your real code.

Stack Overflow

This is the preferred place to get answers to your R questions and problems. Make sure to follow the advice above and to thoroughly search and make sure an answer for your problem does not already exist on Stack Overflow. In addition, the more you participate in the site responding to others and upvoting answers, the more you will receive quality help when you post a question.

Google Groups

Google groups are a general forum-style help site. Posts are both searchable and loosely grouped by topics. This is probably the least prickly and most forgiving of the interactive support sites. Google groups can be found at http://groups.google.com.

RStudio Support Community

Rstudio operates a community support forum and provides a place to ask questions about R and particularly about using R studio. Their website can be found at https://support.rstudio.com/hc/en-us/community/topics/.

R Email Lists

The R Project maintains a number of subscription-based email lists (https://www.r-project.org/mail.html) for posing and answering questions about R, including the general R-help (https://stat.ethz.ch/mailman/listinfo/r-help) email list; lists for announcements about R (https://stat.ethz.ch/mailman/listinfo/r-announce) and R packages (https://stat.ethz.ch/mailman/listinfo/r-packages); and a variety of more specialized lists. Before posing a question on one of these lists, please read the R mailing list instructions (https://www.r-project.org/mail.html) and the posting guide (https://www.r-project.org/posting-guide.html).