

# Collecting R Package Usage Statistics

by Zhiyong Zhang

**Abstract** Usage statistics about a package can help its developers, and ultimately its users, in many ways. However, such information is rarely available. An R package **rpkgstats** is developed in the hope to stimulate more discussion on how to better collect and utilize package usage information. The package **rpkgstats** makes easily possible rating a package and viewing package usage information within R.

As of April 2013, there are almost 4,500 R packages available on CRAN and the number is growing. Contribution of R packages from all around the world has kept R more sophisticated than the rest of statistical software and has also largely advanced statistical data analysis. Usage statistics on a package can help the developers understand how a package is used, track potential bugs, and improve the package in a future release, to say the least. In some cases, such information can also be used to make high-stake decision on the developers, e.g., by their supervisors.

As a developer of several R packages myself, however, I found that usage information of R packages was rare. There are certainly good reasons for the lack of usage information. CRAN has many mirrors that appear and disappear over time. Therefore, it is generally difficult to track the usage statistics. For example, although the site <http://neolab.stat.ucla.edu/cranstats/> provides statistics about package usage, the information is limited to the packages downloaded from the UCLA CRAN mirror. Furthermore, it only provides basic information on the downloads of a given package. The site <http://crantastic.org/> has been recently started by Dr. Hadley Wickham to collect more information on R usage. However, the users of R seemed to lack the enthusiasm to use the site, I think, largely because of the lack of a direct connection between R and the website.

In this paper, I propose a simple method that can be used to more actively collect information on the usage of R packages. This method, utilizing an R package **rpkgstats** and a web server, allows users to provide usage information of a package and makes possible the interaction between users and developers within R. The usage statistics can be viewed within R directly. The method can be easily integrated to CRAN or crantastic. The package can also be used in other scenarios as we will point out.

In the following, I will first demonstrate how to use the R package **rpkgstats**. Then I will explain how the package works. Finally, I will discuss possible ways to improve the package in the future.

## Use of rpkgstats

The package **rpkgstats** is currently available on r-forge and can be installed using `install.packages('rpkgstats', repos='http://r-forge.r-project.org')`. After installing the package, a user can start to provide download information, rate, and comment on any R package currently installed by the user. The information provided by a user will be sent to the maintainer of the R package automatically.<sup>1</sup> A user can also ask a question related to a package. If the user chooses to provide his/her email address, any reply to that question will be sent to the user, too.

Before rating the package, it is always convenient to set up some common information using the function `pkginfo`. For example,

```
pkginfo(package="rsem", email="myemail@xxx.com", name="Zhiyong Zhang", cc="r-help@r-project.org")
```

saves the information in the global settings that can be accessed using `options()$pkginfo` within the R session. This means that a user will provide information on the package **rsem** and the email address and name of the user are also provided. Note that one can cc the information to provided email address, too. After setting this up, one can use the package without repeatedly providing such information.

A user can use the `download()` function to let the maintainer of a package know that she/he has downloaded the package. Note that if the function `pkginfo` is not used, one should use `download(package="rsem", email="myemail@xxx.com", name="Zhiyong Zhang")`. A user can also let the maintainer of a package know that she/he likes the package using `like()`. Furthermore, a rating on the package can be provided using `rate(5)`. A user can rate a package with scores 1, 2, 3, 4, 5 indicating Bad, OK, Good, Very Good, Excellent ratings through the argument `rating`. Feedback can be provided on a package using the `feedback` function as shown below,

```
feedback(comment="I found the package was useful for analyzing my non-normal missing data.")
```

<sup>1</sup>The maintainer of a package can opt out/in the option to receive the feedback information anytime.

Currently, a comment is limited to 1,000 characters.

All information provided by a user is saved on a web server at <http://rstats.psychstat.org>. The usage information of a package can be accessed within R using the function below,

```
view(comment=FALSE, ncomment=1:5, package)
```

Note that a user does not have to install a package to view its usage information. By default, the function only provides the information on download, like, and rating. If a user wants to view the comments of a package, he/she needs to set `comment = TRUE`. By default, only the 5 latest comments will be retrieved. However, one can change the argument `ncomment` to view more comments or select a specific subset of comments.

The following example displays the usage information of the R package **rsem**. Basically, the output says that the package has been downloaded 6 times. This is not the actually download time but the number of times that users chose to report their downloads. One user voted liking the package and the overall rating for this package is 4.83 (between Very Good and Excellent). There are also 59 pieces of feedbacks on the package. In the comment part, the number at the beginning is the id of the feedback in our database. Following the id is a number in the parentheses which indicates the total number of replies to the feedback. Note that the example is made by the author, not really from end users, to illustrate how to use the package **rpkgstats**. If one has not set up the `pkginfo` function, the code `view(comment=T, package="rsem")` needs to use to generate the same output.

```
> view(comment=T)
Download: 6
Like: 1
Rating: 4.83
```

There are 59 comments in total. The number in parentheses is # of replies.

```
81 (1) how to use rsem?
80 (0) This is a test
79 (0) A user just rated your package.
78 (0) A user liked your package. Congratulations!
77 (0) A user just downloaded your package. Congratulations!
```

Sometimes, one may want to ask a question about a package. This can also be done within R using our package. For example, the following code will send the question to the maintainer of the R package **rsem**.

```
ask("Is there a reference to the package?", email="user@xxx.com", name="User")
```

Note that if a user provides an email address, any reply to this question will be sent to it. The question will be assigned a unique id for reference as shown in the `view()` output above. Assuming the id is 20 here.

If the maintainer of the package or any other users want to answer a question, the function `reply` can be used. For example,

```
reply(id=20, "Yes, you can download it at WWW")
```

will send the user who asked the question the information provided. A user who answered the question can also choose to provide his/her email address and name.

A user can monitor the reply to a question anytime within R using the function `viewreply`. For example `viewreply(id=20)` lists all replies to the question with the id 20.

## How does rstats work

The package **rpkgstats** utilizes the R package **RCurl** (Lang, 2013) to communicate between R and a web server (<http://rstats.psychstat.org>). Each R function will call a PHP function on the web server to process usage information.<sup>2</sup> For example, The function `rate` calls the PHP script file `rate.php` on the web server to save usage information of a package provided by a user. The whole process takes place within R without actually opening a web browser. However, because the information needs to be uploaded to a web server, Internet connection is necessary. To retrieve information regarding a package, the R function `view` calls the script file `view.php` on the web server. The retrieved information is then organized in the way as displayed in the previous example.

## Uniqueness of usage information

In order to avoid repeated submission of the same information within R, e.g., accidentally using the loop, we try to generate a unique ID for each user. To generate such an ID, information from functions `Sys.getenv` and

<sup>2</sup>The server side PHP code can be browsed at <http://rstats.psychstat.org/phpcode>

`Sys.info` is used. In addition, the output from either `ipconfig` or `ifconfig` is used. Specifically, the output is converted to a string and then converted to a SHA-256 hash value through the package **digest** (Eddelbuettel, 2013). Therefore, although a user is assigned a unique id, the id does not contain useful information about the identification of a user. On our webserver, the information is further converted to md5 hash value and saved. If within a short period of time, a user with the same md5 hash value rates a package more than once, a warning information will be returned. However, it is possible that two users can have the same md5 hash value based on the method used here. But at the same time, we do not want to collect more information in order to protect the privacy of users.

## Future development

Many times, it will be a good idea to know which version of a package is used on what kind of operating systems. Such information is saved if a user chooses to provide it. However, currently, we have not utilized the information in reporting the usage statistics. The main reason is that we do not have enough information to process such statistics at the current stage.

Ideally, R base can incorporate certain method in evaluating the usage of a package. For example, if the `rate` function can be incorporated into the function `install.packages`, the download information can be very accurate and one can also get more information such as which repository a package is downloaded. If the download function is incorporated into the function `library`, one can get the information each time a package is used. Certainly, this may raise concerns on privacy.

The developers of R packages can also incorporate **rpkgstats** in their own packages. For example, it is often useful if the developers understand the possible error information of a package. In addition to display the error information to end users, one can also let the users choose to send the error information to the developers directly as a comment. The code below gives a simple example so that package developers can adapt for their use.

```
example <- try(replaced by any analysis)
if (class(example) != "try-error"){
  ...
  cat("Do you want to send the error information to the maintainer of the package? Y or N \n")
  flush.console()
  yesno <- readline()
  if (yesno != 'N') feedback('error information here', package='rsem')
  ...
}
```

## Closing remarks

Admittedly, the package **rpkgstats** and the described method for collecting usage information are still in its early developmental stage. However, it clearly demonstrates the possibility to better collect and utilize R package usage statistics. I hope this will stimulate more discussion among R developers and users to develop better methods in the future.

Any comments and suggestions can be sent to me at:

Zhiyong Zhang  
Department of Psychology  
University of Notre Dame  
USA  
[rstats@psychstat.org](mailto:rstats@psychstat.org)

## Bibliography

- D. Eddelbuettel. *digest: Create cryptographic hash digests of R objects*, 2013. URL <http://CRAN.R-project.org/package=digest>. R package version 0.6.3. [p3]
- D. T. Lang. *RCurl: General network (HTTP/FTP/...) client interface for R*, 2013. URL <http://CRAN.R-project.org/package=RCurl>. R package version 1.95-4.1. [p2]