Searching help pages of R packages

by Spencer Graves, Sundar Dorai-Raj, and Romain François

The sos package provides a means to quickly and flexibly search the help pages of contributed packages, finding functions and datasets in seconds or minutes that could not be found in hours or days by any other means we know.

The main capability of this package is the findFn function, which scans the "function" entries in Jonathan

Baron's "R site search"

need a bibliography entry for Baron's "R site search"

data base and returns the matches in a data.frame of class findFn. Baron's is one of five search capabilities currently identified under "search" from the main "www.r-project.org" web site. It includes options to search both R mailing list archives plus the help pages of R packages contributed to CRAN (the Comprehensive R Archive Network) plus a few others.

The print method for objects of class findFn displays the results as a table in a web browser with links to the individual help pages, sorted by package displaying the one with the most matches first. This is different from the RSiteSearch function, as findFn returns the results in R as a data.frame, which can be further manipulated, and the ultimate display in a web browser is a table, unlike the list produced by RSiteSearch.

Other sos functions provide summaries with one line for each package, support the union and intersection of findFn objects, and write the results to an Excel file with three sheets: (1) PackageSum2, which provides an enhanced summary of the packages with matches, (2) the findFn table itself, and (3) the call used to produce it.

Three examples are considered below: First we find a dataset containing a variable Petal.Length, used without indicating the source by Chambers

... how to cite? w/o page numbers cite **?**. citet **?**.

citep (**?**).

with page numbers cite (**?**, pp. 282-283). citet **?**, pp. 282-283.

citep (**?**, pp. 282-283).

Second, we study R capabilities for splines, including looking for a function named spline. Third, we search for contributed R packages with capabilities for solving differential equations.

Finding a Variable in a Data Set: Petal.Length

(**?**, pp. 282-283) Chambers (2009, p. 282-283) uses a variable Petal.Length from a famous Fisher data set but without naming the dataset nor indicating where it can be found nor even if it exists in any contributed R package. The sample code he provides does not work by itself. To get his code to work to produce his Figure 7.2, we must first obtain a copy of this famous data set in a format compatible with his code.

How to add Bibliography ? Chambers, John (2009) Software for Data Analysis (Springer)

To look for this data set, one might first try the help.search function. Unfortunately, this function returns nothing in this case:

```
> library(sos)
> help.search('Petal.Length')
No help files found ...
```

When this failed, many users might then try RSiteSearch('Petal.Length'). This produced 80 matches when it was tried one day (and 62 matches a few months later). RSiteSearch('Petal.Length', 'function') will identify only the help pages. We can get something similar and for many purpose more useful as follows:

```
> library(sos)
> PL <- findFn('Petal.Length')
```

PL is a data.frame of class findFn identifying all the help pages in Jonathan Baron's data base matching the search term. An alias for "findFn" is "???", and this same search can be performed as follows:

```
> PL <- ???Petal.Length
```

This data.frame has columns Count, MaxScore, TotalScore, Package, Function, Date, Score, Description, and Link. Function is the name of the help page, not the name of the function, as multiple functions may be documented on a single help page, and some help pages document other things such as data sets. Score is the index of the strength of the match used by Baron's search engine to decide which items to display first. Package is the name of the package containing Function. Count gives the total number of matches found in this findFn call. By default, the findFn object is sorted to present first the most important Package first (sorting first by Count, MaxScore, TotalScore, and Package in case multiple packages have the same number of matches to the search string), then placing the Function with the highest Score first within each Package.

The summary method for such an object prints a table giving for each Package the Count (number of matches), MaxScore (max of Score), TotalScore (sum of Score), and Date, sorted like a Pareto chart to place the Package with the most help pages first:

```
> summary(PL)
```

```
Total number of matches: 27
Downloaded 27 links in 14 packages.
Packages with at least 1 match using search
                      pattern 'Petal.Length':
Package Count MaxScore TotalScore       Date
yaImpute    8        1          8 2009-08-16
<...>
datasets    1        2          2 2009-07-09
<...>
```

One of the listed packages is datasets. Since it is part of the default R distribution, we decide to look there first. We can select that row of PL just like we would select a row from any other data.frame:

```
> PL[PL$Package=='datasets', 'Function']
[1] iris
```

Problem solved in less than a minute! Any other method known to the present authors would have taken substantially more time.

Finding Packages with Spline Capabilities

Three years ago, the lead author of this article decided he needed to learn more about splines. A literature search began as follows:

```
RSiteSearch('spline')
```

(using the RSiteSearch function in the utils package). While preparing this manuscript, this command identified 1526 documents one day. That is too many. It can be restricted to functions as follows:

```
RSiteSearch('spline', 'fun')
```

This identified only 739 one day (631 a few months earlier). That's an improvement over 1526 but is still too many. To get a quick overview of these 739, we can proceed as follows:

```
splinePacs <- findFn('spline')
```

This downloaded a summary of the 400 highest-scoring help pages in the 'RSiteSearch' data base in roughly 5-15 seconds, depending on the speed of the Internet connection. To get all 739 matches, increase the maxPages argument from its default 20:

```
splineAll <- findFn('spline', maxPages=999)
```

The print method for a findFn object displays the result as a table in a web browser.

If we want to find a function named spline, we can proceed as follows:

```
selSpl <- (splineAll[,'Function']=='spline')
splineAll[selSpl, ]
```

This has 0 rows, because there is no help page named spline. This does not mean that no function with that exact name exists, only that no help page has that name. To find a function with that exact name, try findFn('spline('). This produced one match for a function named 'regspline'.

To look for functions whose name includes the characters 'spline', we can use grepFn:

```
grepFn('spline', splineAll, ignore.case=TRUE)
```

This returned a findFn object identifying 78 help pages. The print method for an object of class findFn presents the result in a web browser. In this case, the sixth row is 'lspline' in the 'assist' package, which has a Score of 1. It is the fifth row in this table, because it is in the assist package, which had a total of 34 help pages matching the search term, and this was the only one whose name matched the grepFn pattern.

To try to evaluate further the splineAll findFn object, we must first acknowledge that a table with 739 rows is too large to digest easily.

summary(splineAll) would tell us that the 739 help pages came from 191 different packages and display the first minPackages = 12. (If other packages had the same number of matches as the twelfth package, they would also appear in the summary.)

A more complete view can be obtained in MS Excel format using the writeFindFn2xls function:

```
writeFindFn2xls(splineAll)
```

If either the WriteXLS package and compatible Perl code are properly installed or if you are running Windows with the RODBC package, this produces a '*.xls' Excel file with three sheets:

The PackageSum2 sheet contains information on locally installed packages not available summary.

The findFn sheet contains the search results.

The call sheet gives the call to findFn that generated these search results.

If WriteXLS cannot produce an Excel file with your installation, it will write three *.csv files. (NOTE: Users who do not have MS Excel may like to know that Open Office Calc can open a standard '*.xls' file and can similarly create such files.

citation for Open Office Calc?)

The PackageSum2 sheet (or file) is created by the PackagesSum2 function, which adds information from installed packages not obtained on all functions by findFn. This includes names of author and maintainer, the date packaged, the number of help pages in the package, and the name(s) of any vignettes. This can be quite valuable in prioritizing packages for further study.

For packages not already installed, the standard install.packages function can be used. To simplify this task, the findFn object can be passed to the installPackages function to check all packages for which the number of matches exceeds a second argument minCount and install any of those that are not already installed; the default minCount is the square root of the largest Count. The results from PackageSum2 and the PackageSum2 sheet of writeFindFn2xls will typically contain more information after running installPackages than before.

This analysis quickly gave us a very informative overview of spline capabilities in contributed R packages in a way that can help establish priorities for further study of the different packages and functions. An analysis of this nature almost four years ago led the lead author to the fda package and its companion books, which further led to a collaboration that has joint presentations at three different conferences and a joint book

(?)

.

Combining Search Results to Find Functions to Solve Differential Equations

The lead author of this article recently gave an invited presentation on "Fitting Nonlinear Differential Equations to Data in R"

?

. A key part of preparing for that presentation was a search of contributed R code, which proceeded roughly as follows:

```
de <- findFn('differential equation')
des <- findFn('differential equations')
de. <- de | des
```

The object de has 53 rows, while des has 105. If this search engine were simply searching for character strings, the first would be larger than the second rather than the other way around. The last object de. is the union of the other two; "|" is an alias for unionFindFn. The de. object has 124 rows, which suggests that the corresponding intersection must have (53+105-124) = 34, which is confirmed, e.g., via nrow(de & des).

To make everthing in de. locally available, we can use installPackages(de., minCount=1). This installed all referenced packages except 'rmutil' and a dependency 'Biobase', which were not available on CRAN but are included in Jonathan Baron's "R site search" data base.

Next, writeFindFn2xls(de.) produced a file de..xls in the working directory (identifiable via getwd()).

The PackageSum2 page of that Excel file provided a quick summary of packages with matches, sorted to put the package with the most matches first. In this case, this first package was deSolve, which provides, "General solvers for initial value problems of ordinary differential equations (ODE), partial differential equations (PDE) and differential algebraic equations (DAE)". This is clearly quite relevant to the subject. The second package was PKfit, which is "A Data Analysis Tool for Pharmacokinetics". This may be too specialized for general use. I therefore would not want to study this first unless my primary interest here was in pharmacokinetic models.

By studying this summary page in this way, I was able to decide relatively quickly which packages I should consider first. In making this decision, I gave more weight to packages with one or more vignettes and less weight on those where the 'Packaged' date was old, indicating that the code was not being actively maintained and updated. I also checked the conference information to make sure I didn't embarrass myself by overlooking a package authored or maintained by another invited speaker.

Discussion

In sum, we have found findFn in the sos package to be a very quick and efficient method for finding things in contributed packages. The grepFn function helps quickly look for functions (or help pages) with particular names. The unionFindFn and intersectFindFn (especially via their "|" and "&" aliases) can be quite useful where a single search term seems inadequate and multiple searches can be combined to produce something closer to what is desired. An example of this was provided with searching for both "differential equation" and "differential equations".

Finally, the "PackageSum2" sheet of an excel file produced by writeFindFn2xls is quite valuable for understanding the general capabilities available for a particular topic. This could be of great value for authors to find what is already available so they don't duplicate something that already exists and so their new contributions are compatible to the maximum extent feasible with other packages.

The findFn capability also makes it much easier to avoid "the researcher's nightmare" of being told after substantial work that someone else has already done essentially the same thing.

Acknowledgments

The RSiteSearch capabilities here extend the power of the RSiteSearch search engine maintained by Jonathan Baron. Without Prof. Baron's support, it would not have been feasible to develop the features described here. Duncan Murdoch, Marc Schwarz, Dirk Eddelbuettel and Gabor Grothendiek and anonymous referees contributed suggestions for improving, but of course can not be blamed for any

of its deficiencies. The collaboration required to produce the current sos package was greatly facilitated by R-Forge

cite ... ???

The sos package is part of the `RSiteSearch` project hosted there. This project also includes code for a Firefox extension to simplify the process of finding information about R.

Spencer Graves
President and Chief Operating Officer
Structure Inspection and Monitoring
San Jose, CA
email: `spencer.graves@prodsyse.com`

Sundar Dorai-Raj
Google
Mountain View, CA
email: `sdorairaj@google.com`

Romain François

email: `romain.francois@dbmail.com`