rcall v. 3.0 package vignette

E. F. Haghish

June 3, 2021

Contents

Part I
`rcall` package commands

Version: 3.0.0

cite: Haghish, E. F. (2019). Seamless interactive language interfacing between R and Stata. The Stata Journal, 19(1), 61-82.

1 rcall

seamless interactive **R** in Stata. The command automatically returns rclass R objects with *integer*, *numeric*, *character*, *logical*, *matrix*, *data.frame*, *list*, and *NULL* classes to Stata. It also allows passing Stata variable, dataset, macro, scalar, and matrix to R as well as load a dataframe from R to Stata automatically, which provides an automated reciprocal communication between Stata and R. in addition to robust automated data communication between Stata and R, **rcall** also includes several modes for integrating R into Stata, including:

1. **interactive**: executing R code within Stata do-file editor (allowing reproducible data analysis practice)
2. **console**: simulating R console within Stata console for interactive exploratory analysis
3. **vanilla**: embedding R base function and R packages within Stata programs defensively

for more information and examples visit rcall homepage and its GitHub repository. note that **rcall** is only hosted on GitHub and must be installed using the github command.

1.1 Syntax

To call R from Stata use the following syntax

```
rcall [mode] [:] [R-command]
```

the package also includes a few subcommands to facilitate integrating R in Stata

```
rcall [subcommand]
```

the following functions can be used to communicate data from Stata to R:

Function	Description
st.scalar(name)	passes a scalar to R
st.matrix(name)	passes a matrix to R
st.var(varname)	passes a numeric or string variable to R
st.data(filename)	passes Stata data to R. without _filename_, the currently loaded data is used.
st.load(dataframe)	loads data from R dataframe to Stata

Programmers can use **rcall_check** command to evaluate the required version of R, R packages, or **rcall** itself:

```
rcall_check [pkgname>=ver] [pkgname>=ver] [...] , rversion(ver) rcallversion(str)
```

2 Modes

The *mode* changes the behavior of the package and it can be **vanilla** or **interactive**. The default mode is **interactive**, which is used if no other mode is specified. Finally, when the [R-command] is not specified (i.e. only **rcall** is typed), the **console** mode will be executed which simulates R console within Stata results window for interactive use. In all of these modes, **rcall** returns *rclass* objects from R to Stata. These modes are summarized below:

```
Mode                    Description
-----------------------------------------------------------------------------------------------------------------------------------
      __[vanilla](http://www.haghish.com/packages/Rcall.php#vanilla_mode)__
                              Calls R non-interactively. This mode is advised for programmers who wish to embed R in Stata packages

      __[interactive](http://www.haghish.com/packages/Rcall.php#interactive_mode)__
                              when the mode is not specified, R is called interactively which memorizes the actions, objects available in the

      __[console](http://www.haghish.com/packages/Rcall.php#console_mode)__
                              when the R command is not specified, R is called interactively and in addition, R console is simulated within t
                              results back interactively. Similarly, the results are returned to Stata in the background and can be accessed
-----------------------------------------------------------------------------------------------------------------------------------
```

3 Subcommands

rcall allows a few subcommands which provide several features to facilitate working with the package interactivey. The subcommands are summarized in the table below:

```
Subcommand              Description
-----------------------------------------------------------------------------------------------------------------------------------
      [setpath](http://www.haghish.com/packages/Rcall.php#setpath_subcommand) "path/to/R"
                              permanently defines the path to executable R on the machine.
      [clear](http://www.haghish.com/packages/Rcall.php#clear_subcommand)
                              erases the R memory and history in the interactive mode.
      [warnings](http://www.haghish.com/packages/Rcall.php#warnings_subcommand)
                              shows the warnings returned from R
      [describe](http://www.haghish.com/packages/Rcall.php#describe_subcommand)
                              returns the R version and paths to R, RProfile, and Rhistory
      [history](http://www.haghish.com/packages/Rcall.php#history_subcommand)
                              opens __Rhistory.do__ in do-file editor which stores the history of the interactive R session.
      [site](http://www.haghish.com/packages/Rcall.php#site_subcommand)
                              opens __rprofile.site__ in do-file editor which is used for customizing R when is called from Stata.
-----------------------------------------------------------------------------------------------------------------------------------
```

4 Description

R statistical language is a free software and programming langage for statistical computing and graphics. The rcall package combines the power of R with Stata, allowing the Stata users to call R interactively within Stata, embed it in Stata programs, and communicate data and analysis results between R and Stata simultaniously.

In other words, anytime an R code is executed, the R objects are available for further manipulation in Stata. R objects with *numeric*, *integer*, *character*, *logical*, *matrix*, *list*, and *NULL* classes are automatically returned to Stata as rclass.

R objects with *data.frame* class can be automatically loaded from R to Stata using the **st.load()** function (see below).

rcall uses the **try** function to evaluate the R code and returns **r(rc)** scalar which is an indicator for errors occuring in R. if **r(rc)** equals zero, R has successfully executed the code. Otherwise, if

r(rc) equals one an error has occured and **rcall** will return the error message and break the execution.

4.1 Communication from R to Stata

Stata automatically receives R objects as rclass anytime the rcall is executed. If R is running interactively (i.e. without **vanilla** subcommand), the previous objects still remain accessable to Stata, unless they are changed or erased from R. Moreover, the packages that you load from Stata in R remain loaded until you detach them.

Accessing R objects in Stata is simultanious which makes working with rcall convenient. For example a *numeric*, or *string* vector which is defined in R, can be accessed in Stata as simple as calling the name of that object withing rclass i.e. **r(*objectname*)**.

A *numeric* object example:

```
. rcall clear            //clear the R interactive session
. rcall: a <- 100
. display r(a)
100
```

Without the **vanilla** subcommand or until **rcall clear** command is used again, the defined object remains in the memory of R and consequently, returned to Stata anytime R is called.

```
. rcall: a
[1] 100
```

A *string* object example:

```
. rcall clear            //clear the R interactive session
. rcall: str <- "Hello World"
. display r(str)
Hello World

. rcall: str <- c("Hello", "World")
. display r(str)
"Hello"  "World"
```

A *vector* example:

```
. rcall: v <- c(1,2,3,4,5)
. display r(v)
1 2 3 4 5
```

A *matrix* example:

```
. rcall: A = matrix(1:6, nrow=2, byrow = TRUE)
. mat list r(A)
r(A)[2,3]
    c1  c2  c3
r1   1   2   3
r2   4   5   6
```

A *list* example:

```
. rcall: mylist <- list(a=c(1:10))
. display r(mylist_a)
1 2 3 4 5 6 7 8 9 10
```

A *logical* example:

```
. rcall: l <- T
. display r(l)
TRUE
```

A *NULL* example:

```
. rcall: n <- NULL
. display r(n)
NULL
```

Regarding communicating R data set to Stata automatically, see the **st.load(*dataframe*)** function below.

4.2 Communication from Stata to R

For an ideal reciprocation between Stata and R, Stata should also easily communicate variables to R. Local and global macros can be passed within R code, since Stata automatically interprets them while it passes the code to rcall command, as shown in the example below:

```
. global a 99
. rcall: (a <- thi is a string)
[1] 99
```

In order to pass a scalar from Stata to R, you can use the **st.scalar()** function as shown below:

```
. scalar a = 50
. rcall: (a <- st.scalar(a))
[1] 50
```

Similarly, Stata matrices can be seamlessly passed to R using the **st.matrix()** function as shown below:

```
. matrix A = (1,2\3,4)
. matrix B = (96,96\96,96)
. rcall: C <- st.matrix(A) + st.matrix(B)
. rcall: C
     [,1] [,2]
[1,]   97   98
[2,]   99  100
```

And of course, you can access the matrix from R in Stata as well:

```
. mat list r(C)
r(C)[2,2]
     c1   c2
r1   97   98
r2   99  100
```

Passing variables from Stata to R is convenient, using the **st.var(*varname*)** function. Therefore, any analysis can be executed in R simply by passing the variables required for the analysis from Stata to R:

```
. sysuse auto, clear
. rcall: dep <- st.var(price)
. rcall: pre <- st.var(mpg)
. rcall: lm(dep~pre)

Call:
lm(formula = dep ~ pre)

Coefficients:
(Intercept)          pre
    11267.3       -238.3
```

The rcall package also allows to pass Stata data to R within **st.data(*filename*)** function. This function relies on the **readstata13** package in R to load Stata data sets, without converting them to CSV or alike. The **readstata13** package is faster and more acurate then **foreign** and **haven** packages and read Stata 13 and 14 datasets. This R package can be installed within Stata as follows:

```
. rcall: install.packages("readstata13", repos="http://cran.uk.r-project.org")
```

Specify the relative or absolute path to the data set to transporting data from Stata to R. For example:

```
. rcall: data <- st.data(/Applications/Stata/ado/base/a/auto.dta)
. rcall: dim(data)
```

If the *filename* is not specified, the function passes the currently loaded data to R.

```
. sysuse auto, clear
. rcall: data <- st.data()
. rcall: dim(data)
[1] 74 12
```

Finally, the data can be imported from R to Stata automatically, using the **st.load(*dataframe*)** function. This function will automatically save a Stata data set from R and load it in Stata by clearing the current data set, if there is any. Naturally, you can have more control over converting variable types if you write a proper code in R for exporting Stata data sets. Nevertheless, the function should work just fine in most occasions:

```
    . clear
    . rcall: st.load(cars)
    . list in 1/2
    +--------------+
    | speed   dist |
    |--------------|
 1. |     4      2 |
 2. |     4     10 |
    +--------------+
```

4.3 Remarks

You should be careful with using Stata symbols in R. For example, the **$** sign in Stata is preserved for global macros. To use this sign in R, you should place a backslash before it to pass it to R. For example:

```
. rcall: head(cars$speed)
```

Also, the object name in R can include a dot, for example:

```
. rcall: a.name <- "anything"
```

The rcall package returns scalars and locals which can only include underscore in the names (e.g. a_name). rcall automatically converts dots to underscore in the name. In the example above, if you type return list in Stata, you would get a macro as follos:

```
. return list
r(a_name) : "anything"
```

To maximize the speed of calling R from Stata, detach the packages that are no longer needed and also, drop all the objects that are of no use for you. The more objects you keep in R memory, the more time needed to automatically communicate those objects between R and Stata.

4.4 Example

Visit rcall homepage for more examples and documentation.

4.5 Author

E. F. Haghish
Department of Psychology
University of Oslo
haghish@uio.no
    rcall Homepage
Package Updates on Twitter

This help file was dynamically produced by MarkDoc Literate Programming package

5 rcall_check

examines the required version of R and R packages

5.1 Syntax

rcall_check [*pkgname>=version*] [[...]] [, *options*]

option	*Description*
rversion(*str*)	specify the minimum required R version
rcallversion(*str*)	specify the minimum required **rcall** version

5.2 Description

rcall_check can be used to check that R is accessible via **rcall**, check for the required R packages, and specify a minimum acceptable versions for R, **rcall** , and all the required R packages.

As showed in the syntax, all of the arguments are optional. If **rcall_check** is executed without any argument or option, it simply checks whether R is accessible via **rcall** and returns **r(rc)** and **r(version)** which is the version of the R that is used by the package. If R is not reachable, an error is returned accordingly.

5.3 Example(s)

checking that R is accessuble via rcall

```
. rcall_check
```

check that the minimum rcall version 1.3.3, ggplot2 version 2.1.0, and R version of 3.1.0 are installed

```
. rcall_check ggplot2>=2.1.0 , r(3.1.0) rcall(1.3.3)
```

5.4 Stored results

5.4.1 Scalars

r(rc): indicates whether R was accessible via **rcall** package

r(version): returns R version accessed by **rcall**

Version: 1.0.0

6 matexport

export matrices from Stata as a data set

6.1 Syntax

matexport , rnames(name) filename(filename) version(13)

the command exports a data set from a given matrix with the given name and includes the matrix row names in a column named

6.2 Citation

The program is based on a command written by **Nicholas J. Cox**, in a package called **dm79**, from which I read and borrowed a command named **svmat2**. I have used the version 1.2.2

Since the package is hosted on SSC and cannot be specified as a dependency, I had to embed it in this package with a different name.

6.3 Author

E. F. Haghish Department of Psychology University of Oslo haghish@hotmail.com
rcall Homepage Package Updates on Twitter

This help file was dynamically produced by MarkDoc Literate Programming package